HYPERTTS: Parameter Efficient Adaptation in Text to Speech using Hypernetworks

Yingting Li[1*], **Rishabh Bhardwaj**[2*], **Ambuj Mehrish**[2*], **Bo Cheng**[1], **Soujanya Poria**[2]
[1]Beijing University of Posts and Telecommunications, China
[2]Singapore University of Technology and Design, Singapore
cindyyting@bupt.edu.cn, rishabh_bhardwaj@mymail.sutd.edu.sg, ambuj_mehrish@sutd.edu.sg
chengbo@bupt.edu.cn, sporia@sutd.edu.sg

Abstract

Neural speech synthesis, or text-to-speech (TTS), aims to transform a signal from the text domain to the speech domain. While developing TTS architectures that train and test on the same set of speakers has seen significant improvements, out-of-domain speaker performance still faces enormous limitations. Domain adaptation on a new set of speakers can be achieved by fine-tuning the whole model for each new domain, thus making it parameter-inefficient. This problem can be solved by Adapters that provide a parameter-efficient alternative to domain adaptation. Although famous in NLP, speech synthesis has not seen much improvement from Adapters. In this work, we present **HYPERTTS**, which comprises a small learnable network, "hypernetwork", that generates parameters of the Adapter blocks, allowing us to condition Adapters on speaker representations and making them dynamic. Extensive evaluations of two domain adaptation settings demonstrate its effectiveness in achieving state-of-the-art performance in the parameter-efficient regime. We also compare different variants of HYPERTTS, comparing them with baselines in different studies. Promising results on the dynamic adaptation of adapter parameters using hypernetworks open up new avenues for domain-generic multi-speaker TTS systems. The audio samples and code are available at https://github.com/declare-lab/HyperTTS.

Keywords: Text to Speech, Speaker Adaptation, Hypernetwork, Parameter Efficient Adaptation

1. Introduction

Neural text-to-speech (TTS) synthesis has transformed our interactions with digital content by converting text into natural-sounding speech. Current TTS systems are often limited to predefined speaker styles or specific sets of speaker IDs (Ren et al., 2019a), reducing their utility in multi-speaker environments with unseen speakers. To make TTS scalable and economical, parameter-efficient adaptation of such systems to new speakers is an important, but highly challenging problem (Li et al., 2023b).

Zero-shot and few-shot speaker adaptation techniques (Shen et al., 2023; Li et al., 2023a; Casanova et al., 2021; Cooper et al., 2020; Casanova et al., 2022; Shen et al., 2023) have gained prominence in the domain of TTS, aiming at accommodating new speakers and styles with limited speaker-specific data. While these methods excel in scenarios with constrained data, it's important to note that when sufficient data is available, fine-tuning the model offers distinct advantages. Fine-tuning allows for highly personalized and tailored speech synthesis, precise control over the alignment of synthesized speech with the speaker's characteristics, and the production of higher-quality, more natural-sounding speech.

In this paper, we assume sufficient availability of data from the adaptation domain. When adapting a multi-speaker TTS model (**backbone model**) to a target domain, the traditional approach involves complete fine-tuning of the entire backbone (Figure 1-*Fine-tuning*). However, this approach is resource-intensive, requiring separate copies of model parameters for each new target domain. To make the adaptation scalable, recent research has introduced parameter-efficient domain adaptation methods using Adapters, as seen

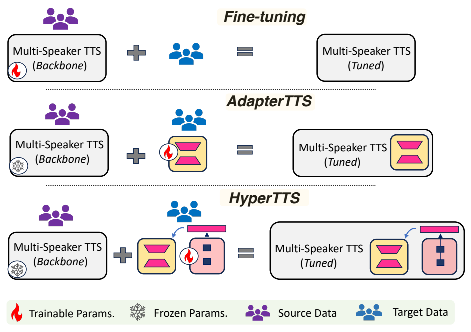

Fine-tuning
Multi-Speaker TTS (Backbone)
Multi-Speaker TTS (Tuned)
AdapterTTS
Multi-Speaker TTS (Backbone)
Multi-Speaker TTS (Tuned)
HyperTTS
Multi-Speaker TTS (Backbone)
Multi-Speaker TTS (Tuned)
Trainable Params.
Frozen Params.
Source Data
Target Data

Figure 1: Comparison of our approach against baselines: *Fine-tuning* tunes the backbone model parameters on the adaptation dataset. *AdapterTTS* inserts learnable modules into the backbone. *HyperTTS* (ours) converts the static adapter modules to dynamic by speaker-conditional sampling using a (learnable) hypernetwork. Both AdapterTTS and HyperTTS keep the backbone model parameters frozen and thus parameter efficient.

[*] Equal Contribution.

in NLP (Houlsby et al., 2019) and speech (Li et al., 2023b). Adapters incorporate small blocks of learnable dense layers into each block of the backbone model, with the aim of learning additional parameters while keeping the main model parameters fixed (Figure 1-*AdapterTTS*). Despite the advantages demonstrated by adapters in various NLP tasks, their direct application in adapting a TTS backbone to a target domain has shown limited improvements (Li et al., 2023b)

Since learning a generic TTS system that works well across different speaker styles is a more difficult problem than learning one network per speaker (Ren et al., 2019a, 2021), we hypothesize the same is the case with adapters. Forcing a static set of adapter parameters to perform well across multiple speakers of the adaptation domain can be challenging and potentially infeasible due to underparameterization (Mehrish et al., 2023a; Biadsy et al., 2022).

In this paper, we present **HYPERTTS**, a pioneering approach for the parameter-efficient adaptation of TTS models to new speakers. This method conditions adapters on speaker embeddings, expanding the learnable parameter space through a "hypernetwork". The main highlights of HyperTTS are:

1. *Dynamic Adapters*: Instead of keeping the adapters static, for each speaker in the adaptation domain, HYPERTTS learns speaker-adaptative adapters. Adapter conditioning on speaker representations is observed to unlock adapter capabilities and make them performant which was a challenge with static adapters (Li et al., 2023b).

2. *Parameter Sampling*: A large set of speakers makes it infeasible to keep the space of adapter parameters discrete. To facilitate this, we employ parameter sampling from a continuous distribution defined by a learnable hypernetwork.

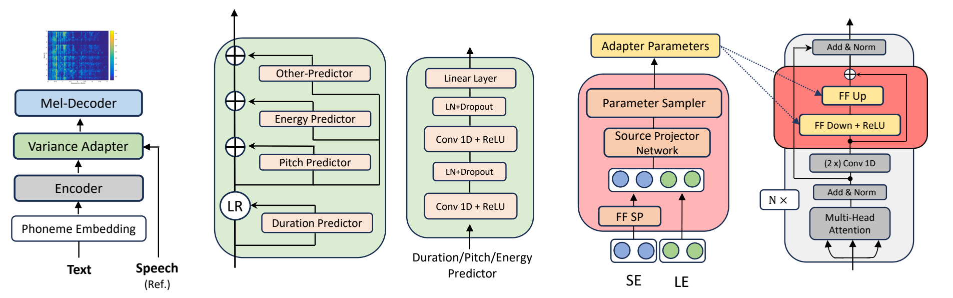
3. *Parameter Efficiency*: Compared to parameter-expensive fine-tuning, it achieves competitive results with less than 1% of the backbone parameters, making it highly practical and resource-friendly for scalable applications.

We perform a comprehensive set of experiments to showcase HYPERTTS's effectiveness (see Figure 1) compared to traditional methods like static bottleneck adapters (AdapterTTS) and full model fine-tuning (TTS-FT). Our experiments cover datasets from diverse environmental conditions, such as LibriTTS and VCTK, representing various accents from different regions. Results highlight HYPERTTS's parameter-efficient performance advantages over the baselines across both objective and subjective metrics. Notably, HYPERTTS can even surpass fine-tuning in performance with only a 20% increase in parameters (Table 6-HYPERTTS$_{e/v/d}$). A key strength of HYPERTTS lies in its remarkable parameter efficiency: it achieves results within 1 point of fine-tuning while using less than 1% of the parameter count in the backbone. This practical and resource-friendly approach enables real-world applications.

2. Related Works

Text-to-speech models. The rise of deep learning has transformed TTS technology, with neural network-based architectures like Tacotron (Wang et al., 2017; Shen et al., 2017), FastSpeech2 (Ren et al., 2021), and Transformer-TTS (Li et al., 2019) leading the way. These models represent significant progress in TTS, leveraging deep learning techniques. Autoregressive TTS models (Wang et al., 2017; Valle et al., 2020; Ren et al., 2021, 2019a; Kim et al., 2020; Łańcucki, 2021), while effective, face limitations in maintaining alignment in long utterances and exhibit slower training and inference speeds with longer sequences. In contrast, non-autoregressive (parallel) models separate phoneme duration estimation from decoding, reducing latency and enhancing training efficiency. These models typically rely on external aligners or pre-trained autoregressive models for phoneme duration. To achieve training efficiency and support end-to-end TTS, this paper focuses on a non-autoregressive TTS model with an alignment framework based on the RAD-TTS (Shih et al., 2021) alignment learning objective, as proposed by Badlani et al. (2022). Recently, several speech models have been compared to GPT in natural language processing, with a focus on in-context learning for speech. Notably, VALL-E (Wang et al., 2023) and SPEAR-TTS (Kharitonov et al., 2023) leverage emerging codecs to learn discrete speech tokens and employ a vocoder-like decodec to convert these tokens into waveforms. Meanwhile, Voicebox, inspired by flow-matching and aligned with the Fastspeech framework, utilizes continuous features like Mel spectrogram and HiFi-GAN.

Speaker Adaptation in TTS. Speaker adaptation is a crucial aspect of TTS systems, aiming to personalize the synthesized speech by modifying the voice characteristics to match those of a specific target speaker. Over the years, various techniques and approaches have been proposed to address the challenges associated with speaker adaptation in TTS (Jia et al., 2018; Chen et al.; Min et al., 2021; Hsieh et al., 2022; Gabryś et al.,



(a) Architecture (b) Variance Adapter (c) HyperX (d) Encoder/Decoder

Figure 2: An overview of the HYPERTTS. SE and LE denote speaker embedding and layer embedding.

2022). Furthermore, several studies have focused on exploring parameter-efficient methods for adapting TTS to new sets of speakers, addressing the need for effective adaptation in diverse speaker scenarios. These approaches aim to accommodate a wide range of linguistic variations (Pamisetty et al., 2023; Do et al., 2022), including diverse accents (Yang et al., 2023), speakers (Luo et al., 2021; Miao et al., 2021; Mehrish et al., 2023a), and low-resource scenarios introduced by the target domain (Azizah and Jatmiko, 2022; Mehrish et al., 2023a; Lux and Vu, 2022), while maintaining the number of trainable parameters. HYPERTTS primarily focuses on contributing in the line of parameter-efficient domain adaptation of the backbone TTS model to a target set of speakers.

Dynamic Parameters. Parameter generation, although not popular in speech, has been used in various forms in other domains, such as Klein et al. (2015); Riegler et al. (2015) in NLP and Ha et al. (2017) in computer vision. Specific to adapters, Bhardwaj et al. (2022); Chen et al. (2020) make prompt tokens dynamic by conditioning their values on input text using a parameter prompt generator network, (Üstün et al., 2022; Mahabadi et al., 2021) used hypernetworks for generating adapter down and up-projection weights. Shared hypernetworks obviate the need to maintain a separate set of parameters for each task (or new setting) and generate weights for each block of the backbone network (Mahabadi et al., 2021). To the best of our knowledge, this is the first work that studies the utility of a parameter generator in the domain of speech (Mehrish et al., 2023b).

3. Methodology

The TTS backbone architecture comprises a text (phoneme) encoder, Variance Adapter, and Mel-Decoder. We pre-train the backbone on LibriTTS, a large multi-speaker dataset for which we provide details in Section 4.3.

3.1. Encoder

Given a phoneme sequence $(p_1, \ldots, p_n)$ obtained from text, we first map it to vector embedding mixed with sinusoidal positional encoding. The Encoder is composed of four Feed-Forward Transformer (FFT) blocks, with each block comprised of two multi-head attention modules and two 1D-convolutions. Shown in Figure 2-(d), we adopt Encoder's FFT from FastSpeech (Ren et al., 2019a) which adapted the text-specific transformer block from Vaswani et al. (2017) for text-to-speech. Each head performs a self-attention to contextualize each phoneme with relevant global phoneme sequence information. Followed by contextualization with multiple heads, the encoder uses two 1D convolutions over the phoneme sequence to capture local phoneme information. This is because adjacent phonemes and mel-spectrogram features are more closely related in speech, thus, kernels are intuitively an effective alternative[1]. The output of the first convolution is ReLU activated.

3.2. Variance Adapter

Variance Adapter (VA) (Figure 2-(b)) transforms the phoneme embeddings of length n into mel-spectrogram embeddings of length m where m is typically larger than n. Thus, each phoneme at the input tends to map to one or multiple mel-frames. To solve this length mismatch, VA comprises a duration predictor that predicts how many mel-frames are required for each phoneme. Following this, VA also dedicates itself to predicting pitch and energy for each element of the length-regulated phoneme sequence. Each predictor is associated with a corresponding loss function.

Duration Predictor. Let $\mathcal{H} = (h_1, \ldots, h_n)$ denote the phoneme sequence input of VA. Dura-

[1]We do not claim convolutions are better replacements of dense layers. The model characteristics can change with large parameter speech models, thus leaving it for future research.

tion predictor takes this as the input and predicts $\mathcal{D} = (d_1, \ldots, d_n)$ where d_i is a positive integer that tells how many mel-frames the phoneme will be mapped to. Before feeding to Pitch and Energy, the phoneme sequence is expanded using predicted phoneme durations $\mathcal{D}$. The length-regulated phoneme sequence will appear $(h_1..[d_1\text{-times}]..h_1, \ldots, h_n..[d_n\text{-times}]..h_n])$.

Duration predictor warrants ground truth durations, i.e., alignment between phoneme sequences and mel-frames. Following Badlani et al. (2022), we train a phoneme-to-mel unsupervised alignment predictor end-to-end with the backbone TTS architecture, obviating the need for external aligners such as MFA (McAuliffe et al., 2017) and non-autoregressive aligner models (Peng et al., 2020; Ren et al., 2019a; Łańcucki, 2021). Similar to Shih et al. (2021), we compute the soft alignment distribution based on the learned pairwise affinity between all text tokens and mel-frames. Given a soft alignment map, the Viterbi algorithm is used to find the monotonic path with the highest likelihood in order to convert soft alignments to hard alignments. We relegate specifics of the duration predictor to Badlani et al. (2022); Shih et al. (2021).

Pitch Predictor. We use continuous wavelet transform (CWT) for pitch contour prediction, following the approach in Ren et al. (2021). During training, the pitch predictor targets a pitch spectrogram obtained from the continuous pitch series decomposition. During inference, the pitch predictor's spectrogram output is converted back to pitch contours using inverse CWT (iCWT). This pitch predictor comprises a two-layer 1D-convolutional network with ReLU activation, followed by layer normalization and dropout. The pitch spectrogram is obtained via a linear layer. To handle discontinuous pitch contours, we employ linear interpolation for unvoiced frames, apply a logarithmic transformation, and normalize to have zero mean and unit variance. Mean and variance predictions are extracted from the output of the 1D-convolutional layer across the time dimension, and values are obtained from separate linear layers. The pitch predictor's parameters are learned by minimizing the mean squared error (MSE) between the spectrogram, mean, and variance values of the ground truth and predictions

Energy Predictor. The energy predictor estimates the original energy values for each Short-Time Fourier Transform (STFT) frame, for which it computes the L2-norm of the frame's amplitude. To optimize efficiency, the calculated energy is quantized into 256 evenly distributed values. These quantized values are then encoded into an energy embedding and added to the expanded hidden sequence, similar to the pitch information.

3.3. Mel-Decoder and Postnet.

The decoder converts the variance adaptor's hidden sequence into a mel-spectrogram. It shares the same architecture as the encoder but with six FFT blocks. To improve mel-spectrogram quality, a Postnet is used at the mel-decoder's output, reducing artifacts and distortions in speech.

3.4. Hypernetwork

The adapter performs the following operation:

$$h = h + ReLU(hW_d)W_u \tag{1}$$

where h is the hidden representation in the TTS network, W_d and W_u are down-projection $(d_h \rightarrow d_r)$ and up-projection matrices $(d_r \rightarrow d_h)$ with d_r as bottleneck dimension. The projection matrices are attached to each layer of the encoder, decoder, and variance adapter after the convolutional layers (shown in Figure 2-d). Although adapters have shown significant advantages in NLP and several speech tasks (Houlsby et al., 2019; Li et al., 2023b), they are observed to be less useful in performing TTS adaptation (Li et al., 2023b). We posit that forcing a static set of adapter parameters to perform well across multiple speakers of the adaptation domain can be challenging.

HYPERTTS is aimed to make adapters significantly more effective by conditioning them on speaker embeddings, thus enhancing the (effective) learnable parameter space of adapters. For this purpose, it employs a hypernetwork. A hypernetwork is typically a small neural network that generates weights for a larger main network performing the usual learning task (here it is text-to-speech). We propose to enhance the effectiveness of adapters by learning to adapt their parameters with the change in speaker:

$$W_d = f_d(v_s, v_l) \tag{2}$$
$$W_u = f_u(v_s, v_l) \tag{3}$$

Here v_s and v_l denote speaker embedding of dimension d_1 and layer embedding of dimension d_l, respectively. f_d and f_u denote parameter generators which are part of the hypernetwork and are learnable modules. Notably, to keep the network small in size, we leverage a shared hypernetwork to generate parameters for adapters in every layer of a given module of the TTS backbone. We only train the hypernetwork while keeping the TTS backbone frozen. Since the hypernetwork is significantly smaller than the backbone, we refer to the adaption as parameter efficient. In this work, the hypernetwork is smaller than 1% of the TTS backbone size (in the number of parameters).

Implementation: With reference to Figure 2-c, we project d_1-dimensional speaker embedding (SE) onto a d_2-dimensional space, where $d_2 < d_1$, using a speaker projector (SP) which is a feed-forward network with bias. To this, we concatenate a d_l-dimensional layer embedding (LE). Layer embeddings are a learnable look-up table that maps a layer-id to a vector. The concatenated vector is passed through a source projector network that maps a $(d_2 + d_l)$-dimensional vector into a d_s-dimensional space. For adapter down and up projection, we sample weights from the hypernetwork through the source projector network using dedicated dense (Parameter Sampler) layers.

4. Experiments

In this section, we elaborate on the comparable baselines, target domain multi-speaker datasets, TTS backbone model configurations, and evaluation metrics used.

4.1. Baseline models

We assess the quality and similarity of speech samples generated by HYPERTTS by comparing them to other methods. These methods include:

TTS-0 TTS-0 represents the zero-shot performance of the TTS model, wherein it is pre-trained on source data (LTS) and subsequently evaluated on target data without any fine-tuning, which is kept as a baseline to define lower-bound on the performance of the parameter-efficient training (HYPERTTS and AdapterTTS).

Reference and Reference (Voc.) Reference refers to ground truth speech. To obtain Reference (voc.) we transform the reference speech into mel-spectrograms and subsequently reconstruct the speech signals using HiFi-GAN (Kong et al., 2020).

TTS-FT (full fine-tuning) TTS-FT denotes the model obtained after fine-tuning all parameters of the backbone model on the target dataset, which is kept as a baseline to define upper-bound on the performance of the parameter-efficient training (HyperTTS and AdapterTTS).

AdapterTTS It inserts bottleneck adapter modules, a down-projection and up-projection layer, in each layer of the backbone model's encoder. We follow the configuration proposed by Pfeiffer et al. (2021). AdapterTTS learns only adapter parameters, keeping the backbone parameters frozen. We examine various configurations of AdapterTTS: AdapterTTS$_e$, AdapterTTS$_v$, AdapterTTS$_d$ denoting bottleneck adapter block inserted in the encoder, VA, and decoder. The combination of all is represented by AdapterTTS$_{e/v/d}$.

HYPERTTS For a comprehensive evaluation, we examine various variants of HYPERTTS: HYPERTTS$_e$, HYPERTTS$_v$, HYPERTTS$_d$ denoting bottleneck adapter block inserted in the encoder, VA, and decoder. The combination of all is represented by HYPERTTS$_{e/v/d}$. Hypernetwork is not shared across the different modules of the architecture, thus, the number of parameters are in order HYPERTTS$_{e/v/d}$ >HYPERTTS$_d$ ≈ HYPERTTS$_v$ ≈ HYPERTTS$_e$.

4.2. Datasets

We primarily base our experiments on English language datasets. For the training and adaptation, we employed three distinct datasets: train-clean-100, the dev-clean and test-clean subset of LibriTTS (Zen et al., 2019), and VCTK (Veaux et al., 2013). In the subsequent discussion, we will refer to the train-clean-100 subset as LTS and the (dev-clean, test-clean) subset as LTS2. TTS backbone is pre-trained on the LTS dataset. For adaptation, we consider VCTK and LTS2. Each of the datasets is divided into train and validation subsets. To evaluate the effectiveness of the model, we conducted performance assessments using dedicated validation subsets from both the VCTK and LTS2.

4.3. Model Configuration

Backbone Model Pre-training. The encoder consists of four layers, and the decoder comprises six layers, both with a hidden state dimension of 256. In Figure 2-(a), the variance adapter adds the speaker embedding to the text representation. Speaker embeddings are computed using the speaker verification model trained with the generalized end-to-end (GE2E) loss, incorporating data from LibriSpeech (train-other-500)(Panayotov et al., 2015), Voxceleb1, and Voxceleb2 (Nagrani et al., 2017). For this work, we set $d_h = 256$, $d_r = 32$, $d_1 = 256$, $d_2 = 64$, $d_l = 64$, and $d_s = 32$.

In the pre-training of the multi-speaker TTS backbone model on the LTS dataset, speech samples are downsampled to a 16 kHz sampling rate. The model is trained using the Adam optimizer. Unsupervised duration modeling, which includes the pitch predictor, duration predictor, and energy predictor in the variance adapter, begins at 50K steps to enhance model convergence. To ensure stable training, a learning rate warm-up is implemented for the first 4K steps, followed by annealing at steps 300K, 400K, and 500K. After training the backbone model for 600K steps, adaptation is carried out on either the VCTK or LTS2 datasets. Backbone fine-tuning, Hypernetworks in HYPERTTS, and AdapterTTS are adjusted for the next 300K steps. The Hypernetwork is trained for 300K steps using the Adam optimizer with a constant learning rate of 0.0001.

4.4. Evaluation Metrics

In this section, we discuss various metrics we use to compare our model against the baselines.

Objective Metrics: We assess timbre and prosody similarity between synthesized and reference audio using two metrics: cosine similarity (COS) and F0 Frame Error (FFE). COS provides insights into speaker similarity by measuring the average cosine similarity between embeddings from the synthesized and ground truth data. FFE focuses on fundamental frequency (F0) information, considering voicing decision error and F0 error metrics. Mel cepstral distortion (MCD) is a widely used objective metric that quantifies perceptual differences between synthesized and original speech. It measures the divergence between the Mel-scale Frequency Cepstral Coefficients (MFCC) of the synthesized speech and the original speech. Additionally, we calculate Word Error Rate (WER) to gauge the intelligibility of the generated speech. In our experiments, we use enterprise-grade speech-to-text (STT) pre-trained silero models (Team, 2021) to compute WER.

Subjective Metrics: In addition to objective evaluations, we conducted a comprehensive series of subjective listening tests with six participants to assess the naturalness of the synthesized audio. These tests offer valuable insights into human perception and subjective preferences. Participants listened to a random selection of 5 sets of 20 sound samples, including four distinct methods (TTS-FT, AdapterTTS$_e$, HYPERTTS$_e$, and HYPERTTS$_d$), along with a reference sample. They rated each sample on a five-point Likert Scale (Joshi et al., 2015), ranging from 1 (Poor) to 5 (Excellent). To ensure an unbiased evaluation, each row in the test contained samples with identical text content, eliminating potential bias from text variations and allowing for a fair assessment of the samples.

To assess speaker individuality, we conducted an XAB test (Mizuno and Abe, 1995) to evaluate speaker similarity. The reference speech of the target speaker was denoted as X, while the synthesized speech produced by the AdapterTTS, and $HyperTTS_d$ models were presented to listeners as options A and B in random order. A specific comparison was made between AdapterTTS and $HyperTTS_d$. A total of 6 listeners with backgrounds in NLP and speech participated in the experiment, during which they were presented with 30 synthesized speech samples (including 15 reference samples).

5. Results and Discussions

In Table 1, we observe full fine-tuning (TTS-FT) of the backbone TTS model on VCTK (TTS-0) improves the COS score by ≈7 points on its test set, and reduce frame error (FFE) of basemodel by ≈4.5 points. This forms our baseline in a non-parameter efficient regime. First, we carry out extensive experiments on AdapterTTS by inserting it in the encoder, VA, decoder, encoder+decoder, and all three encoder+decoder+VA with approximately 0.2%, 0.1%, 0.3%, 0.5%, and 0.56% trainable parameters. We observe that compared with AdapterTTS$_v$ and AdapterTTS$_d$, AdapterTTS$_e$ shows advantages in three of four metrics. Overall, the performance of AdapterTTS across its variants stays significantly lower than fine-tuning performance, making it a less interesting choice for adaptation. While performing TTS adaptation on LTS2 (Table 2), we draw a similar inference where AdapterTTS$_e$ comes out to be hardly advantageous when compared with TTS-FT baselined across the objective metric. Moving further, we experiment on AdapterTTS$_e$ as AdapterTTS$_{e/v/d}$ shows relatively poor performance on WER while the performance of AdapterTTS$_{e/d}$ is lower by 1 point on speaker similarity COS score, which is one of the critical attributes of multi-speaker TTS adaptation.

Next, we change the adapter parameters from static learnable to dynamic by sampling them using a hypernetwork, conditioning the sampling process on speaker style and layer-id. In Table 1, we observe HYPERTTS significantly improves the backbone performance on two out of four metrics while staying in a parameter-efficient regime. HYPERTTS$_d$ i.e., hypernetwork for adapter block in decoder performs best on two out of four metrics across its low parameter setting (<0.5% added parameters) when comparing against HYPERTTS$_e$ and HYPERTTS$_v$. We posit that adapting HYPERTTS$_e$ is not effective as it is dedicated to encoding phoneme embedding while adapting HYPERTTS$_v$ in VA is observed to be noisy, observed through an increase in MCD score. Trading off with a few parameters (≈1% added parameters) in HYPERTTS$_{e/d}$ and HYPERTTS$_{e/v/d}$, we can achieve performance close to the full fine-tuning baseline within a margin of ≈1% in COS, FFE, WER, and MCD. Except for WER where decoder adaptation is close to AdapterTTS, HYPERTTS consistently outperforms AdapterTTS and achieves performance closer to the fine-tuning baseline. A similar trend is seen for LTS2 adaptation (Table 2) where HYPER-

LTS → VCTK					
Model	**COS ↑**	**FFE ↓**	**WER ↓**	**MCD ↓**	**Params**
Reference	$100.000_{(\pm 0.000)}$	$00.00_{(\pm 0.00)}$	0.2055	–	–
Reference (Voc.)	$95.027_{(\pm 0.001)}$	$22.10_{(\pm 0.03)}$	0.2074	–	–
TTS-0	$73.794_{(\pm 0.004)}$	$39.19_{(\pm 0.02)}$	0.2035	5.9232	–
TTS-FT	$80.443_{(\pm 0.003)}$	$34.63_{(\pm 0.02)}$	0.2027	5.2387	35.7M (100%)
AdapterTTS$_e$	$73.769_{(\pm 0.004)}$	$38.73_{(\pm 0.02)}$	**0.2075**	5.9002	66.6K (0.186%)
AdapterTTS$_v$	$73.131_{(\pm 0.004)}$	$42.87_{(\pm 0.02)}$	0.2258	6.2733	33.3K (0.095%)
AdapterTTS$_d$	$76.180_{(\pm 0.004)}$	$39.14_{(\pm 0.03)}$	0.2101	6.0092	100K (0.280%)
AdapterTTS$_{e/d}$	$72.703_{(\pm 0.004)}$	$37.99_{(\pm 0.02)}$	0.2141	5.8804	166.7K (0.466%)
AdapterTTS$_{e/v/d}$	$77.298_{(\pm 0.006)}$	$35.53_{(\pm 0.03)}$	0.2234	**5.2971**	200K (0.559%)
HyperTTS$_e$	$75.432_{(\pm 0.004)}$	$36.07_{(\pm 0.02)}$	0.2367	5.3930	151.1K (0.423%)
HyperTTS$_v$	$73.731_{(\pm 0.004)}$	$38.47_{(\pm 0.02)}$	0.2367	5.9137	150.9K (0.422%)
HyperTTS$_d$	$77.590_{(\pm 0.004)}$	$38.55_{(\pm 0.02)}$	0.2090	5.9641	151.2K (0.423%)
HyperTTS$_{e/d}$	$79.232_{(\pm 0.003)}$	$35.02_{(\pm 0.02)}$	0.2168	5.3650	302.3K (0.846%)
HyperTTS$_{e/v/d}$	**79.464**$_{(\pm 0.003)}$	**34.47**$_{(\pm 0.02)}$	0.2340	5.3293	453.3K (1.269%)

Table 1: Domain adaptation performance on VCTK. TTS-0 denotes the zero-shot performance of the backbone TTS model evaluated on the VCTK test set. TTS-FT is a fine-tuned backbone model on the VCTK train set and evaluated on its test set. Where subscript $m \in \{e, v, d\}$ in HYPERTTS$_m$ or AdapterTTS$_m$ denotes hypernetwork-adapter or adapter inserted to the encoder, variance adapter, and decoder of the backbone model, respectively.

LTS → LTS2					
Model	**COS ↑**	**FFE ↓**	**WER ↓**	**MCD ↓**	**Params**
Reference	$100.000_{(\pm 0.000)}$	$00.00_{(\pm 0.00)}$	0.2046	–	–
Reference (Voc.)	$96.919_{(\pm 0.000)}$	$19.72_{(\pm 0.02)}$	0.2089	–	–
TTS-0	$78.784_{(\pm 0.005)}$	$43.31_{(\pm 0.02)}$	0.2129	7.7039	–
TTS-FT	$82.351_{(\pm 0.004)}$	$41.26_{(\pm 0.02)}$	0.2135	7.5843	35.7M (100%)
AdapterTTS	$77.989_{(\pm 0.005)}$	$42.28_{(\pm 0.02)}$	**0.2143**	7.6581	66.6K (0.186%)
HYPERTTS$_e$	$78.302_{(\pm 0.006)}$	$41.38_{(\pm 0.02)}$	0.2232	**7.4746**	151.1K (0.423%)
HYPERTTS$_v$	$78.853_{(\pm 0.005)}$	$43.19_{(\pm 0.02)}$	0.2180	7.7055	150.9K (0.422%)
HYPERTTS$_d$	$81.021_{(\pm 0.005)}$	$41.77_{(\pm 0.02)}$	0.2159	7.5942	151.2K (0.423%)
HYPERTTS$_{e/d}$	$81.360_{(\pm 0.005)}$	$41.95_{(\pm 0.02)}$	0.2180	7.5865	302.3K (0.846%)
HYPERTTS$_{e/v/d}$	**81.742**$_{(\pm 0.005)}$	**41.03**$_{(\pm 0.02)}$	0.2337	7.5876	453.3K (1.269%)

Table 2: Speaker adaptation on LTS2 is assessed. "TTS-0" represents the base TTS model's zero-shot performance on the LTS2 test set. "TTS-FT" is the fine-tuned base model on the LTS2 training set, evaluated on its test set. The subscript $m \in e, v, d$ in HYPERTTSm or AdapterTTSm denotes the integration of hypernetwork-adapter or adapter in the encoder, variance adapter, or decoder of the base model, respectively.

Model	**COS ↑**	**FFE ↓**	**WER ↓**	**MCD ↓**	**Params**
HYPERTTS$_d$(2)	$75.89_{(\pm 0.0040)}$	$38.99_{(\pm 0.0211)}$	0.2096	6.0016	50K (0.14%)
HYPERTTS$_d$(8)	$77.59_{(\pm 0.0036)}$	$38.55_{(\pm 0.0208)}$	**0.2090**	**5.9641**	151K (0.42%)
HYPERTTS$_d$(32)	$79.38_{(\pm 0.0033)}$	**38.05**$_{(\pm 0.0210)}$	0.2152	6.0024	554K (1.55%)
HYPERTTS$_d$(128)	**80.26**$_{(\pm 0.0033)}$	$38.15_{(\pm 0.0210)}$	0.2186	5.9820	2.17M (6.06%)

Table 3: Varying number of parameters of hypernetwork in decoder with VCTK as the target domain. HYPERTTS$_d(n)$ denotes hypernetwork with n-dimensional source projection.

Model	**MOS ↑**	**Params**
Reference	$4.62_{(\pm 0.02)}$	-
TTS-FT	$3.70_{(\pm 0.16)}$	35.7M (100%)
AdapterTTS$_e$	$3.47_{(\pm 0.07)}$	66.6K (0.186%)
HyperTTS$_e$	$3.43_{(\pm 0.10)}$	151.1K (0.423%)
HyperTTS$_d$	$3.64_{(\pm 0.06)}$	150.9K (0.422%)

Table 4: MOS comparison among Reference, TTS-FT, AdapterTTS$_e$, HYPERTTS$_e$ and HYPERTTS$_d$ for samples randomly selected from VCTK validation set.

TTS$_d$ increases the COS score by over 2 points, however, scores did not change significantly on other metrics. We posit that this is potentially due to the undertraining of the network. Similar to the findings of VCTK, we observe while adapting the backbone to LTS2. AdapterTTS worsens in the speaker similarity COS score and MCD although slightly improving on FFE and WER errors. In the class of models with less than 0.5% added parameters, hypernetwork in decoder HYPERTTS$_d$ achieves performance close to the fine-tuning baseline TTS-FT on all the metrics. It is noteworthy that HYPERTTS and adapters while achieving close to the TTS-FT performance, do not inherently suffer from catastrophic forgetting, thus, providing advantages of both regimes. After fine-tuning, TTS-FT performance on the pre-training dataset VCTK becomes worse COS $83.10 \rightarrow 77.09$ and FFE $39.01 \rightarrow 39.88$.

5.1. Subjective Evaluation

We conducted a subjective evaluation, comparing the naturalness and quality of synthesized speech to a reference sample. Results of the side-by-side subjective test are shown in Table 4. Notably, HYPERTTS$_d$ achieved an MOS of 3.6412, surpassing AdapterTTS with an MOS of 3.4727. TTS-FT obtained a MOS of 3.7021 with 35.7 million parameters, while HYPERTTS$_d$ only used 0.422% of the parameters in TTS-FT, as seen in Table 4. These results highlight HYPERTTS$_d$'s superior performance in both subjective evaluation scores and parameter efficiency when compared to AdapterTTS and TTS-FT. We also conducted a paired t-test to assess the statistical significance of our findings. The results indicate a highly significant difference in mean opinion scores between the Reference and all models (p < 0.0001). There is a statistically significant contrast in mean opinion scores between AdapterTTSe ($p \sim 0.0415$) and HYPERTTSe ($p \sim 0.0345$) when compared to TTS-FT. However, in the case of HYPERTTSd ($p \sim 0.56$), we lack sufficient grounds to reject the null hypothesis (p < 0.05), suggesting that TTS-FT and HYPERTTSd likely share identical expected values.

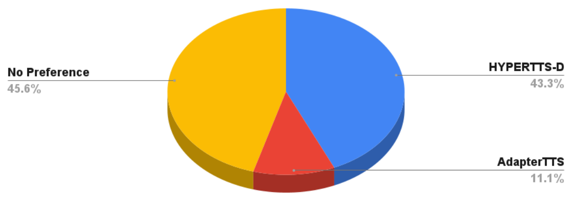


Figure 3: XAB speaker similarity test results between AdapterTTS, and $HyperTTS_d$.

The speaker similarity results depicted in Figure 3 indicate a preference for samples generated by $HyperTTS_d$, with a percentage of 43.3% favoring it over AdapterTTS.

5.2. Impact of Parameter Efficiency

The parameter efficiency of hypernetwork and adapters depends on the projection dimension. We investigate this trade-off by varying the size of the source projector network in 3 variants: HYPERTTS$_e$, HYPERTTS$_d$, and HYPERTTS$e/v/d$, as shown in Figure 2-c. The results of these experiments are detailed in Tables 3, 5 and 6. AdapterTTS, with approximately 66K parameters, has fewer parameters than smaller versions of HYPERTTS. When adapting to VCTK, increasing the down-projection dimension from 32 to 128 results in adapter parameters growing to around 263K. In this case, the COS decreases, while FFE scores increase to 73.58 and 39.46, respectively. This suggests that AdapterTTS does not benefit from parameter scaling in the same way as HYPERTTS.

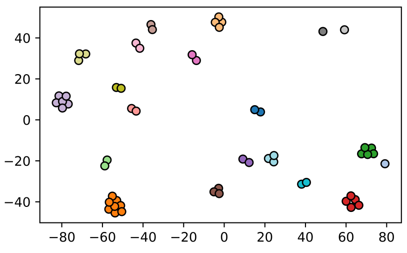


Figure 4: t-SNE of hypernetwork generated parameters for 20 randomly sampled speakers from VCTK test set. The same color marks represent reference speech from the same speaker.

5.3. Output of Hypernetwork

In Figure 4, in the continuous space of parameters, we observe that different reference speech from the same speaker clustered together while different speakers are distant apart. Thus, hypernetwork behaves as a dynamic adapter, conditioning its parameter on the speaker style. In contrary to this, a static adapter will learn a single set of parameters (one mark) in the possible space of parameters.

5.4. Other Discussions

Layernorms (standard and conditional). Similar to Mahabadi et al. (2021), we also experiment with inserting a conditional layer norm in the adapter network output where, however, results did not show any significant performance difference both in HYPERTTS and AdapterTTS.

Low-Rank Adaptation. We refrain from employing hypernetwork for generating LoRA (Hu et al.) parameters instead of the bottleneck adapter. One primary reason is the higher number of parameters involved in LoRA due to the adaptation of query, key, and value in each backbone layer. We leave further exploration in this direction as a future work.

6. Conclusion

In this paper, we present **HYPERTTS**, an approach that enhances the effectiveness of adapters by conditioning them on speaker embeddings. Utilizing a "hypernetwork" to customize adapter block

Model	COS ↑	FFE ↓	WER ↓	MCD ↓	Params
HyperTTS$_e$(2)	$74.64_{(\pm 0.0038)}$	$37.27_{(\pm 0.0178)}$	**0.2170**	5.4041	50K (0.14%)
HyperTTS$_e$(8)	$75.43_{(\pm 0.0035)}$	$36.07_{(\pm 0.0193)}$	0.2367	5.3930	151K (0.42%)
HyperTTS$_e$(32)	$\mathbf{76.02}_{(\pm 0.0035)}$	$\mathbf{35.17}_{(\pm 0.0190)}$	0.2449	**5.3779**	554K (1.5%)
HyperTTS$_e$(128)	$75.97_{(\pm 0.0036)}$	$35.93_{(\pm 0.0193)}$	0.2612	5.4210	2.17M (6.06%)

Table 5: Varying number of parameters of hypernetwork in encoder with VCTK as the target domain.

Model	COS ↑	FFE ↓	WER ↓	MCD ↓	Params
HyperTTS$_{e/v/d}$(2)	$77.26_{(\pm 0.0034)}$	$35.10_{(\pm 0.0180)}$	**0.2314**	5.3436	150K (0.42%)
HyperTTS$_{e/v/d}$(8)	$79.46_{(\pm 0.0030)}$	$34.47_{(\pm 0.0198)}$	0.2340	5.3293	453K (1.27%)
HyperTTS$_{e/v/d}$(32)	$80.67_{(\pm 0.0027)}$	$33.85_{(\pm 0.0209)}$	0.2513	**5.2761**	1.66M (4.65%)
HyperTTS$_{e/v/d}$(128)	$\mathbf{81.49}_{(\pm 0.0031)}$	$\mathbf{33.58}_{(\pm 0.0223)}$	0.2622	5.2879	6.50M (18.19%)

Table 6: Varying number of parameters of hypernetwork in the encoder, decoder, and VA with VCTK as the target domain.

weights for the TTS backbone network, we significantly expand the adapter parameter space. This dynamic method replaces the conventional static adapter parameter set, enabling input-conditioned parameter sampling. Additionally, the hypernetwork's continuous parameter space theoretically allows the generation of adapter parameters for numerous speakers without increasing hypernetwork parameters. This makes HyperTTS an excellent choice for multi-speaker TTS adaptation, surpassing traditional adapter limitations.

Limitations: While hypernetworks exhibit promising enhancements in both adaptation domains, there are training challenges to address. Time and resource constraints may have led to potential underfitting, negatively impacting performance. Additionally, hypernetworks tend to overfit the backbone model on the adaptation domain, warranting further research to enhance their generalizability. Notably, the relatively higher number of parameters in hypernetworks poses potential inefficiency for low-resource training.

7. Acknowledgments

We thank the anonymous reviewers for their constructive feedback. This work was supported in part by the National Key Research and Development Program of China under Grant 2022YFF0902701, in part by the National Natural Science Foundation of China under Grants U21A20468, 62372058, U22A201339, in part by the Fundamental Research Funds for the Central Universities under Grant 2020XD-A07-1, in part by BUPT Excellent Ph.D. Students Foundation CX2021229.

This work is also generously supported by the AcRF MoE Tier-2 grant (Project no. T2MOE2008, and Grantor reference no. MOE-T2EP20220-0017) titled: "CSK-NLP: Leveraging Commonsense Knowledge for NLP", and the SRG grant id: T1SRIS19149 titled "An Affective Multimodal Dialogue System".

8. Bibliographical References

Kurniawati Azizah and Wisnu Jatmiko. 2022. Transfer learning, style control, and speaker reconstruction loss for zero-shot multilingual multi-speaker text-to-speech on low-resource languages. *IEEE Access*, 10:5895–5911.

Rohan Badlani, Adrian Łańcucki, Kevin J Shih, Rafael Valle, Wei Ping, and Bryan Catanzaro. 2022. One tts alignment to rule them all. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6092–6096. IEEE.

Rishabh Bhardwaj, Amrita Saha, Steven C.H. Hoi, and Soujanya Poria. 2022. Vector-quantized input-contextualized soft prompts for natural language understanding. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 6776–6791, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Fadi Biadsy, Youzheng Chen, Xia Zhang, Oleg Rybakov, Andrew Rosenberg, and Pedro J. Moreno. 2022. A scalable model specialization framework for training and inference using submodels and its application to speech model personalization. In *Interspeech 2022, 23rd Annual Conference of the International Speech Communication Association, Incheon, Korea, 18-22 September 2022*, pages 5125–5129. ISCA.

Edresson Casanova, Christopher Shulby, Eren Gölge, Nicolas Michael Müller, Frederico Santos de Oliveira, Arnaldo Candido Jr., Anderson da Silva Soares, Sandra Maria Aluísio, and Moacir Antonelli Ponti. 2021. Sc-glowtts: An efficient zero-shot multi-speaker text-to-speech model. In *Interspeech 2021, 22nd Annual Conference of the International Speech Communication Association, Brno, Czechia, 30 August - 3 September 2021*, pages 3645–3649. ISCA.

Edresson Casanova, Julian Weber, Christopher D Shulby, Arnaldo Candido Junior, Eren Gölge, and Moacir A Ponti. 2022. Yourtts: Towards zero-shot multi-speaker tts and zero-shot voice conversion for everyone. In *International Conference on Machine Learning*, pages 2709–2720. PMLR.

Mingjian Chen, Xu Tan, Bohan Li, Yanqing Liu, Tao Qin, Tie-Yan Liu, et al. Adaspeech: Adaptive text to speech for custom voice. In *International Conference on Learning Representations*.

Mingjian Chen, Xu Tan, Yi Ren, Jin Xu, Hao Sun, Sheng Zhao, and Tao Qin. 2020. Multispeech: Multi-speaker text to speech with transformer. In *Interspeech*.

Erica Cooper, Cheng-I Lai, Yusuke Yasuda, Fuming Fang, Xin Wang, Nanxin Chen, and Junichi Yamagishi. 2020. Zero-shot multi-speaker text-to-speech with state-of-the-art neural speaker embeddings. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6184–6188. IEEE.

Phat Do, Matt Coler, Jelske Dijkstra, and Esther Klabbers. 2022. Text-to-speech for under-resourced languages: Phoneme mapping and source language selection in transfer learning. In *Proceedings of the 1st Annual Meeting of the ELRA/ISCA Special Interest Group on Under-Resourced Languages*, pages 16–22.

Adam Gabryś, Goeric Huybrechts, Manuel Sam Ribeiro, Chung-Ming Chien, Julian Roth, Giulia Comini, Roberto Barra-Chicote, Bartek Perz, and Jaime Lorenzo-Trueba. 2022. Voice filter: Few-shot text-to-speech speaker adaptation using voice conversion as a post-processing module. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7902–7906. IEEE.

David Ha, Andrew M. Dai, and Quoc V. Le. 2017. Hypernetworks. *ArXiv*, abs/1609.09106.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for nlp. In *International Conference on Machine Learning*, pages 2790–2799. PMLR.

Cheng-Ping Hsieh, Subhankar Ghosh, and Boris Ginsburg. 2022. Adapter-based extension of multi-speaker text-to-speech model for new speakers. *ArXiv*, abs/2211.00585.

Edward J Hu, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, et al. Lora: Low-rank adaptation of large language models. In *International Conference on Learning Representations*.

Ye Jia, Yu Zhang, Ron Weiss, Quan Wang, Jonathan Shen, Fei Ren, Patrick Nguyen, Ruoming Pang, Ignacio Lopez Moreno, Yonghui Wu, et al. 2018. Transfer learning from speaker verification to multispeaker text-to-speech synthesis. *Advances in neural information processing systems*, 31.

Ankur Joshi, Saket Kale, Satish Chandel, and D Kumar Pal. 2015. Likert scale: Explored and explained. *British journal of applied science & technology*, 7(4):396.

Eugene Kharitonov, Damien Vincent, Zalán Borsos, Raphaël Marinier, Sertan Girgin, Olivier Pietquin, Matt Sharifi, Marco Tagliasacchi, and Neil Zeghidour. 2023. Speak, read and prompt: High-fidelity text-to-speech with minimal supervision.

Jaehyeon Kim, Sungwon Kim, Jungil Kong, and Sungroh Yoon. 2020. Glow-tts: A generative flow for text-to-speech via monotonic alignment search.

B. Klein, Lior Wolf, and Y. Afek. 2015. A dynamic convolutional layer for short rangeweather prediction. *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4840–4848.

Jungil Kong, Jaehyeon Kim, and Jaekyoung Bae. 2020. Hifi-gan: Generative adversarial networks for efficient and high fidelity speech synthesis. *Advances in Neural Information Processing Systems*, 33:17022–17033.

Naihan Li, Shujie Liu, Yanqing Liu, Sheng Zhao, and Ming Liu. 2019. Neural speech synthesis with transformer network. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 6706–6713.

Yinghao Aaron Li, Cong Han, Vinay S Raghavan, Gavin Mischler, and Nima Mesgarani. 2023a. Styletts 2: Towards human-level text-to-speech through style diffusion and adversarial training with large speech language models. *arXiv preprint arXiv:2306.07691*.

Yingting Li, Ambuj Mehrish, Rishabh Bhardwaj, Navonil Majumder, Bo Cheng, Shuai Zhao, Amir Zadeh, Rada Mihalcea, and Soujanya Poria. 2023b. Evaluating parameter-efficient transfer learning approaches on sure benchmark for speech understanding. In *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5. IEEE.

Renqian Luo, Xu Tan, Rui Wang, Tao Qin, Jinzhu Li, Sheng Zhao, Enhong Chen, and Tie-Yan Liu. 2021. Lightspeech: Lightweight and fast text to speech with neural architecture search. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5699–5703. IEEE.

Florian Lux and Thang Vu. 2022. Language-agnostic meta-learning for low-resource text-to-speech with articulatory features. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6858–6868.

Rabeeh Karimi Mahabadi, Sebastian Ruder, Mostafa Dehghani, and James Henderson. 2021. Parameter-efficient multi-task fine-tuning for transformers via shared hypernetworks. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 565–576.

Michael McAuliffe, Michaela Socolof, Sarah Mihuc, M. Wagner, and Morgan Sonderegger. 2017. Montreal forced aligner: Trainable text-speech alignment using kaldi. In *INTERSPEECH*.

Ambuj Mehrish, Abhinav Ramesh Kashyap, Li Yingting, Navonil Majumder, and Soujanya Poria. 2023a. Adaptermix: Exploring the efficacy of mixture of adapters for low-resource tts adaptation. *arXiv preprint arXiv:2305.18028*.

Ambuj Mehrish, Navonil Majumder, Rishabh Bharadwaj, Rada Mihalcea, and Soujanya Poria. 2023b. A review of deep learning techniques for speech processing. *Information Fusion*, page 101869.

Chenfeng Miao, Liang Shuang, Zhengchen Liu, Chen Minchuan, Jun Ma, Shaojun Wang, and Jing Xiao. 2021. Efficienttts: An efficient and high-quality text-to-speech architecture. In *International Conference on Machine Learning*, pages 7700–7709. PMLR.

Dongchan Min, Dong Bok Lee, Eunho Yang, and Sung Ju Hwang. 2021. Meta-stylespeech: Multi-speaker adaptive text-to-speech generation. In *International Conference on Machine Learning*, pages 7748–7759. PMLR.

Hideyuki Mizuno and Masanobu Abe. 1995. Voice conversion algorithm based on piecewise linear conversion rules of formant frequency and spectrum tilt. *Speech communication*, 16(2):153–164.

Arsha Nagrani, Joon Son Chung, and Andrew Zisserman. 2017. Voxceleb: a large-scale speaker identification dataset. *Telephony*, 3:33–039.

Giridhar Pamisetty, S Chaitanya Varun, and K Sri Rama Murty. 2023. Lightweight prosody-tts for multi-lingual multi-speaker scenario. In *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–2. IEEE.

Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur. 2015. Librispeech: an asr corpus based on public domain audio books. In *2015 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 5206–5210. IEEE.

Kainan Peng, Wei Ping, Zhao Song, and Kexin Zhao. 2020. Non-autoregressive neural text-to-speech. In *International Conference on Machine Learning*, pages 7586–7598. PMLR.

Jonas Pfeiffer, Aishwarya Kamath, Andreas Rücklé, Kyunghyun Cho, and Iryna Gurevych. 2021. Adapterfusion: Non-destructive task composition for transfer learning. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 487–503.

Yi Ren, Chenxu Hu, Xu Tan, Tao Qin, Sheng Zhao, Zhou Zhao, and Tie-Yan Liu. 2021. Fastspeech 2: Fast and high-quality end-to-end text to speech. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Yi Ren, Yangjun Ruan, Xu Tan, Tao Qin, Sheng Zhao, Zhou Zhao, and Tie-Yan Liu. 2019a. Fastspeech: Fast, robust and controllable text to speech. *Advances in neural information processing systems*, 32.

Yi Ren, Yangjun Ruan, Xu Tan, Tao Qin, Sheng Zhao, Zhou Zhao, and Tie-Yan Liu. 2019b. Fastspeech: Fast, robust and controllable text to speech. In *Advances in Neural Information Processing Systems*, volume 32, pages 3171–3180. Curran Associates, Inc.

Gernot Riegler, S. Schulter, M. Rüther, and H. Bischof. 2015. Conditioned regression models for non-blind single image super-resolution.

2015 IEEE International Conference on Computer Vision (ICCV), pages 522–530.

Jonathan Shen, Ruoming Pang, Ron J. Weiss, Mike Schuster, Navdeep Jaitly, Zongheng Yang, Zhifeng Chen, Yu Zhang, Yuxuan Wang, R. J. Skerry-Ryan, Rif A. Saurous, Yannis Agiomyrgiannakis, and Yonghui Wu. 2017. Natural TTS synthesis by conditioning wavenet on mel spectrogram predictions. *CoRR*, abs/1712.05884.

Kai Shen, Zeqian Ju, Xu Tan, Yanqing Liu, Yichong Leng, Lei He, Tao Qin, Sheng Zhao, and Jiang Bian. 2023. Naturalspeech 2: Latent diffusion models are natural and zero-shot speech and singing synthesizers. *arXiv preprint arXiv:2304.09116*.

Kevin J. Shih, Rafael Valle, Rohan Badlani, Adrian Lancucki, Wei Ping, and Bryan Catanzaro. 2021. RAD-TTS: Parallel flow-based TTS with robust alignment learning and diverse synthesis. In *ICML Workshop on Invertible Neural Networks, Normalizing Flows, and Explicit Likelihood Models*.

Silero Team. 2021. Silero models: pre-trained enterprise-grade stt / tts models and benchmarks. `https://github.com/snakers4/silero-models`.

Ahmet Üstün, Arianna Bisazza, Gosse Bouma, Gertjan van Noord, and Sebastian Ruder. 2022. Hyper-x: A unified hypernetwork for multi-task multilingual transfer. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 7934–7949.

Rafael Valle, Kevin Shih, Ryan Prenger, and Bryan Catanzaro. 2020. Flowtron: an autoregressive flow-based generative network for text-to-speech synthesis.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Christophe Veaux, Junichi Yamagishi, and Simon King. 2013. The voice bank corpus: Design, collection and data analysis of a large regional accent speech database. In *2013 international conference oriental COCOSDA held jointly with 2013 conference on Asian spoken language research and evaluation (O-COCOSDA/CASLRE)*, pages 1–4. IEEE.

Chengyi Wang, Sanyuan Chen, Yu Wu, Ziqiang Zhang, Long Zhou, Shujie Liu, Zhuo Chen, Yanqing Liu, Huaming Wang, Jinyu Li, Lei He, Sheng Zhao, and Furu Wei. 2023. Neural codec language models are zero-shot text to speech synthesizers.

Yuxuan Wang, R. J. Skerry-Ryan, Daisy Stanton, Yonghui Wu, Ron J. Weiss, Navdeep Jaitly, Zongheng Yang, Ying Xiao, Zhifeng Chen, Samy Bengio, Quoc V. Le, Yannis Agiomyrgiannakis, Rob Clark, and Rif A. Saurous. 2017. Tacotron: A fully end-to-end text-to-speech synthesis model. *CoRR*, abs/1703.10135.

Li-Jen Yang, Chao-Han Huck Yang, and Jen-Tzung Chien. 2023. Parameter-efficient learning for text-to-speech accent adaptation. *arXiv preprint arXiv:2305.11320*.

Heiga Zen, Viet Dang, Rob Clark, Yu Zhang, Ron J Weiss, Ye Jia, Zhifeng Chen, and Yonghui Wu. 2019. Libritts: A corpus derived from librispeech for text-to-speech. *arXiv preprint arXiv:1904.02882*.

Adrian Łańcucki. 2021. Fastpitch: Parallel text-to-speech with pitch prediction. In *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6588–6592.