# ENHANCING SPEECH-TO-SPEECH TRANSLATION WITH MULTIPLE TTS TARGETS

*Jiatong Shi*[1]*, Yun Tang*[2]*, Ann Lee*[2]*, Hirofumi Inaguma*[2]*, Changhan Wang*[2]*, Juan Pino*[2]*, Shinji Watanabe*[1]

[1] Carnegie Mellon University [2] Meta AI

{jiatongs,swatanbe}@cs.cmu.edu, {yuntang,annl}@meta.com

## ABSTRACT

It has been known that direct speech-to-speech translation (S2ST) models usually suffer from the data scarcity issue because of the limited existing parallel materials for both source and target speech. Therefore to train a direct S2ST system, previous works usually utilize text-to-speech (TTS) systems to generate samples in the target language by augmenting the data from speech-to-text translation (S2TT). However, there is a limited investigation into how the synthesized target speech would affect the S2ST models. In this work, we analyze the effect of changing synthesized target speech for direct S2ST models. We find that simply combining the target speech from different TTS systems can potentially improve the S2ST performances. Following that, we also propose a multi-task framework that jointly optimizes the S2ST system with multiple targets from different TTS systems. Extensive experiments demonstrate that our proposed framework achieves consistent improvements (2.8 BLEU) over the baselines on the Fisher Spanish-English dataset.

***Index Terms***— speech-to-speech translation, text-to-speech augmentation, discrete units

## 1. INTRODUCTION

Speech-to-speech translation (S2ST) focuses on translating speech from a source language into the speech of a target language [1]. Conventional cascaded S2ST models decompose the task into three components, including automatic speech recognition (ASR), machine translation (MT), and text-to-speech (TTS) [2, 3]. Alternatively, some previous works adopt end-to-end speech-to-text translation (S2TT) instead of ASR and MT. However, it would introduce high computational costs and inference latency for further application. To mitigate the issue, recent literature focuses on building direct S2ST models without three standalone modules [4–12].

The training of direct S2ST models needs inevitably large amounts of parallel S2ST corpora, which are far more difficult to obtain than conventional cascaded methods [6]. To mitigate the issue and enable the training for S2ST models, previous works incorporated TTS systems to form the dataset for S2ST [4–11]. Nearly all the published datasets on S2ST are extended from speech-to-text corpora where the target speech for S2ST is synthesized by TTS systems [13–15]. When synthesizing the target speech for S2ST, researchers in previous works usually select a specific TTS system. For instances, in [14], they utilized a variant of Non-attentive Tacotron (NAT) [16], while in [15], they adopted Fastspeech2 [17]. To the best of our knowledge, there is no investigation into how different synthesized target speech would affect the S2ST modeling.

To fill the research gap aforementioned, this paper focuses on the effect of different synthesized speech from various TTS systems. We find simply using training data from multiple TTS systems can improve the performance of S2ST. To further utilize the shared knowledge across multiple TTS systems, we further propose a framework that jointly optimized the S2ST systems with multiple targets from different TTS systems. Results show that our proposed method could significantly improve the S2ST performances over baseline models. To be specific, our proposed framework shows a 2.8 BLEU score improvement over the best baseline system with a single TTS target on the Fisher Spanish-English dataset [18]. The contribution of this work can be summarized as follows:

- We first investigate the effect of different TTS systems for target synthesized speech for S2ST.
- We propose a multi-task framework that combines knowledge from different TTS data, which shows reasonable improvements according to our experiments.

## 2. METHODOLOGY

In this section, we first review the background of this research, including the S2ST system with discrete units and various TTS systems used in this work. Then, we introduce our proposed framework for combining knowledge from target speech from different TTS systems.

### 2.1. Background

**S2ST with discrete units**: Speech self-supervised learning (SSL) models have shown outstanding performances on various tasks [19, 20]. Notably, they are also applicable to synthesis tasks [21–23]. To apply SSL representations, a common strategy is to discretize them into speech units through clustering approaches [21, 24]. Previous works have shown that the discrete units can disentangle linguistic content from other acoustic properties (e.g., speaker identity or prosody information), resulting in easier learning of linguistic information directly from speech [21]. Due to this reason, Lee et al. proposed a direct S2ST model, which uses discrete speech units as the prediction target of the system [7]. Their experiments also demonstrated their superiority over the translatotron-based methods [4, 5] and comparable performances to the cascaded S2ST systems [2, 3, 25]. The discrete units in their system are generated from the K-Means clustering over the representation from a pre-trained HuBERT model [26].

**TTS systems**: As mentioned in Sec. 1, previous studies usually employ TTS systems to generate the target speech for S2ST. The translatotron series of works mainly adopt auto-regressive (AR) TTS systems (i.e., NAT) [4, 5, 14], while there are also other studies that apply non-auto-regressive (NAR) TTS such as Fastspeech2 [15]. All these models are text2Mel models, where they convert the text to Mel spectrogram, so they need additional vocoders to get the waveform of speech. The choices of vocoders also vary, including non-parametric Griffin-Lim and neural vocoders.

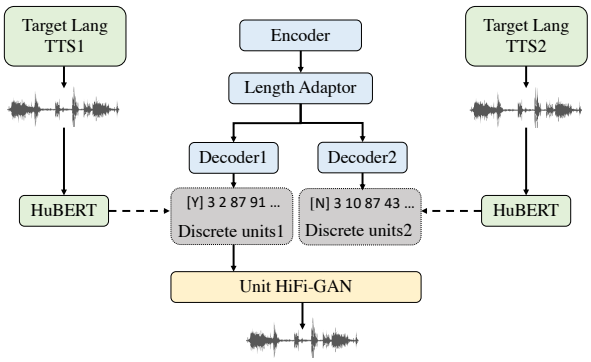


**Fig. 1**. The framework of our proposed S2ST model using multiple TTS targets. The blue blocks represent the S2ST modeling; the green blocks are modules used to generate target discrete units; the gray blocks are the targets of the S2ST model, while the first token is for predicting which TTS target is used for inference; the yellow block is for model inference. Details are explained in Sec. 2.2.

In this work, we select three TTS models: Tacotron2 (TT2) [27], Fastspeech2 (FS2) [17], and VITS [28]. Tacotron2 is a classical AR TTS text2Mel model, while Fastspeech2 is a typical NAR TTS text2Mel model. VITS, different from others (text2Mel + vocoder), directly models the process from text to waveform (text2wav), which does not need additional vocoders. For text2Mel models (i.e., TT2 and FS2), we adopt three different vocoders for investigation: Parallel WaveGAN (PWG) [29], Hifi-GAN (HFG) [30], and StyleMelGAN (SMG) [31]. We also investigate the effect of duration control for NAR models by tuning the speed factor in the inference.

For the first set of our experiments, we evaluate the S2ST model with different target TTS speech. Then, we further conduct experiments on combining synthesized speech from different TTS systems. For this study, we focus only on single-speaker TTS systems.

**Overall Workflow**: Based on the previous work discussed, the overall workflow for constructing an S2ST system is outlined below: (1) Target speech synthesis: Target speech is synthesized using a TTS model, which can be either an acoustic model and vocoder or a direct text2wav model. (2) Discrete unit extraction: The synthesized target speech is converted into discrete units using a HuBERT model by clustering. (3) S2ST system training: The S2ST model is trained using source speech as input and target discrete units as output. (4) Inference: During inference, the S2ST model converts the source speech into a sequence of discrete units. Then, a unit-based vocoder is applied to generate the final waveform speech.

### 2.2. The Proposed Framework

As in [21, 24], speech discrete units from speech SSL representations can potentially disentangle linguistic, prosodic, and speaker-related information. However, at the same time, it is still noisy to use. For example, in [12], the authors have shown that the same sentence spoken by different speakers, could result in different speech discrete unit sequences. A similar phenomenon may happen when the same sentence is generated by different TTS systems. To verify our hypothesis, we measure the Pearson correlation coefficients of the HuBERT units' distribution between different TTS systems trained on LJSpeech [32].[1] As shown in Fig. 2, it clearly indicates that different TTS systems are still different though with the same linguistic source and trained on the same corpus. The data includes

[1]The configuration of HuBERT and TTS systems are discussed in Sec. 3.2 in detail.

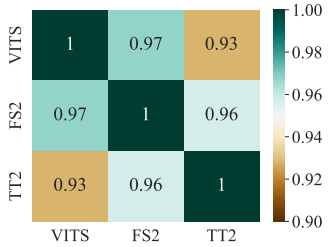


**Fig. 2**. The Pearson correlation coefficients between different TTS systems: the unit distribution is collected with the development set of the Fisher Spanish-English dataset [18]. For the wave generation from FS2 and TT2, we both utilize the Hifi-GAN vocoder.

the development set of the Fisher Spanish-English corpus [18]. On the other hand, given the same utterance in the target language, the synthesized speech should have the same linguistic content. Therefore, it is reasonable to assume that the extracted units could have a similar consensus shared across, given the same text is employed to generate speech from different TTS systems.

Following the assumption discussed above, we propose the framework as shown in Fig. 1. The framework is based on the model proposed in [7] but is additionally designed to capture the high-level consensus over linguistic information across different TTS systems. To be specific, we add separate decoder branches for speech discrete units generated from different TTS systems. For simplicity, in Fig. 1, we show the case with two targets, but it could be easily extended into three or more targets because of its parallel property.

Instead of directly predicting the units in parallel, we also append a special token at the start of **target** unit sequences as shown in the gray blocks in Fig. 1. The special token is defined as an indicator of the quality of synthesized speech, which we can use to select better output during model inference. Practically, we first compute the character error rates (CER) at the sentence level for each utterance from different TTS systems. Then, at the training stage, we assign token [Y] to the TTS system with the best CER among candidate target speech discrete units and token [N] to other systems.[2] For inference, we compute the probability of the first predicted special token [Y] from all the decoders and select the one with the highest probability to continue generating the sequences. Noted that since the special token is at the start of the sequence, the inference process does not need to auto-regressively generate future tokens if it already has a lower probability than other branches. Therefore, compared to the base system without multiple TTS targets, there is not much additional searching burden when doing inference.

Due to the noise present in discrete units, a similar approach was explored in [12]. The authors proposed a speaker normalization method to normalize the units of different speakers to a reference speaker. However, in the case of using different TTS systems, it can be challenging to determine which system should be used as the reference, as they are all synthesized using the same text.

## 3. EXPERIMENTS

### 3.1. Datasets

For S2ST, we use the Fisher Spanish-English dataset [18], which is also widely used in the previous S2ST works [7,9]. The English TTS systems are applied to synthesize target speech from the English text for training and validation. For the training of TTS systems, we use LJSpeech, a 24-hour single-speaker corpus [32].

[2]For even cases, we assign [Y] to all systems with the best CER.

### 3.2. Experimental Settings

#### *3.2.1. Model architectures*

**Speech-to-unit translation model (S2UT)**: We follow the updated version of S2UT model described in [11], which is an extension of [7]. For speech discrete unit generation, we adopt the pre-trained multilingual HuBERT, K-Means clustering, and the unit-based HFG vocoder released in [11, 12]. The vocabulary size of the discrete unit is 1,000, corresponding to the number of clusters in the K-Means model. The generated discrete units are reduced by duplication-pooling, during the training of S2UT models. On the other hand, the reduced units are recovered to their original lengths through a duration predictor that is jointly trained with unit-based HFG. The same of [11], the encoder is initialized from a Conformer-based wav2vec 2.0, while the decoder is initialized from the mBART decoder [33] released in [11]. To keep the consistency over different settings, we do not tune the hyper-parameters, but use the settings as [11] for different systems.

**TTS models**: As mentioned in Sec. 2.1, we utilize three different TTS systems (i.e., TT2, FS2, and VITS) and three different vocoders (i.e., PWG, HFG, and SWG). To keep the reproducibility of the experiments, all the models are from public-available checkpoints in ESPnet-TTS [34, 35], an open-source framework for TTS.

#### *3.2.2. Training and decoding*

For the training of S2UT models, we use the AdamW optimizer [36] with a learning rate of 0.0005. The scheduler is applied with a warmup policy that starts the learning rate from 1e-7 and reaches the maximum learning rate at 20k steps. We accumulate the gradients for every 120 steps to simulate a large batch size, which has shown to be effective for S2ST learning. During the training, we follow the "LNA-D" policy introduced in [11], which does not fine-tune all the parameters in the pre-trained mBART decoder but only the Layer-Norm and self-attention parameters. For the decoding of S2UT, we apply beam search with a beam size of 10.

For NAR TTS models, the duration can be tuned with a speed factor, resulting in the duration control to the generation. In our experiments, apart from the TTS models we discussed in Sec. 2.1, we also apply different speed factors including 0.95, 1.0, and 1.05.

#### *3.2.3. Experiments design*

The experiments generally include three folds:

**Single TTS systems**: To compare the effect of different TTS systems, we directly train the S2UT model with discrete units generated from a single TTS system.

**Simple combination of TTS systems**: As discussed in Sec. 2.2, multiple TTS systems could potentially improve the S2ST, by normalizing the noise from unit sequences. To systematically investigate the effects of different systems, we carry out experiments based on the modeling properties of different TTS systems, including AR versus NAR, different vocoders, different speed factors, and text2Mel versus text2wav.

**Multi-task framework for multiple TTS targets**: We follow the design of the experiments in the simple combination of TTS systems, but change from the data combination into the multi-task way of training, as introduced in Sec. 2.2. We use the wav2vec 2.0-based ASR model to compute the CER mentioned in Sec. 2.2. Due to the requirements of large GPU memory to train models with more than three additional branches, we limit the number of TTS systems to less than four.

**Table 1**. S2ST Performances on different TTS systems. The "Data ID" column stands for the target speech units generated from the TTS system. The CER and BLEU are calculated as discussed in Sec. 3.2.4. The acoustic models (AM) and vocoders are introduced in Sec. 2.1.

| Data ID | AM | Vocoder | CER(↓) | BLEU(↑) |
|---|---|---|---|---|
| **A** | TT2 | PWG | 9.1 | 37.3 |
| **B** | | HFG | 8.9 | 37.3 |
| **C** | | SWG | 8.7 | 37.3 |
| Avg. | TT2 | / | 8.9 | 37.3 |
| **D** | FS2 | PWG | 9.4 | 37.5 |
| **E** | | HFG | **8.3** | 37.6 |
| **F** | | SWG | 8.7 | 37.5 |
| Avg. | FS2 | / | 8.8 | 37.5 |
| **G** | VITS | | 8.4 | **38.3** |

#### *3.2.4. Evaluation metric*

The TTS quality is first evaluated by inputting the synthesized speech to the ASR model and calculating the character error rate (CER) between the ASR prediction and the reference text. For the ASR model, we employ the open-source ASR model that is trained over wav2vec 2.0[3]. For evaluation of the translation quality, we first utilize the same ASR model to get the transcription of the S2ST system (i.e., predicted units + code-HFG vocoder) and then compute BLEU score with the reference text using SacreBLEU [37]. Noted that the reference text is also tokenized and converted to lowercase without punctuation for BLEU calculation.

### 3.3. Results and Discussion

We conduct experiments with target speech discrete units generated from a single TTS system. The best system was obtained from synthesized speech using VITS (data **G**), and the results are shown in Table 1. We observe that there was no significant effect on S2ST performance when different vocoders were applied to models **A**-**C** and models **D**-**F**. However, the difference in acoustic models could affect the S2ST results, with the best system using target speech synthesized from VITS and the worst from TT2. We also find that the CER in each TTS acoustic model roughly correlated with their S2ST performance.

An interesting finding from Table 1 is that systems with different vocoders achieved similar performances despite having different CERs from the ASR model. We assume that this is due to the HuBERT units helping to normalize the vocoder differences in speech synthesis. To verify this hypothesis, we conduct a Pearson coefficient analysis over HuBERT unit distribution on FS2 with different vocoders and found that the Pearson scores were all above 0.98.

Furthermore, we find that VITS usually had a higher CER than TT2 and FS2 for spoken words (e.g., "HMM", "HUM"). This could be due to the data domain mismatch between the read speech used for TTS training (i.e., LJSpeech) and ASR training (i.e., Librispeech) and the conversational speech used for S2ST training (i.e., Fisher).

Table 2 presents the effect of different speed factors on NAR TTS systems. The results show that a smaller speed factor can improve the performance of the S2ST system. However, when mea-

[3] https://huggingface.co/facebook/wav2vec2-large-960h-lv60-self

**Table 2**. S2ST Performances on different speed factors. The Fastspeech2 (FS) model is combined with the HFG vocoder for TTS as default. The TTS models are introduced in Sec. 2.1.

| Data ID | TTS | Speed Factor | CER(↓) | BLEU(↑) |
|---|---|---|---|---|
| **H** | FS2 | 0.95 | 8.7 | 38.0 |
| **E** | | 1.0 | 8.3 | 37.5 |
| **I** | | 1.05 | 9.5 | 37.4 |
| **J** | VITS | 0.95 | 8.6 | **38.7** |
| **G** | | 1.0 | 8.4 | 38.3 |
| **K** | | 1.05 | 8.5 | 38.3 |

**Table 3**. S2ST Performances on multiple TTS targets. We follow the categories listed in Sec. 3.2.3 to conduct experiments The models with ✓in the "Multi-task" column are trained with the framework proposed in Sec. 2.2.

| Category | Data | Multi-task | BLEU(↑) |
|---|---|---|---|
| Best Single TTS system | **J** | / | 38.7 |
| TT2 + FS2 | **B** + **E** | ✗ | 37.0 |
| | | ✓ | 37.6 |
| TT2 + Vocoders | **A** + **B** + **C** | ✗ | 37.3 |
| | | ✓ | 37.3 |
| FS2 + Vocoders | **D** + **E** + **F** | ✗ | 37.7 |
| | | ✓ | 37.9 |
| FS2 + Speed Factors | **H** + **E** + **I** | ✗ | 38.8 |
| | | ✓ | 39.7 |
| VITS + Speed Factors | **J** + **G** + **K** | ✗ | 39.7 |
| | | ✓ | **41.5** |
| VITS + TT2 | **B** + **G** | ✗ | 37.2 |
| | | ✓ | 38.4 |
| VITS + FS2 | **E** + **G** | ✗ | 39.6 |
| | | ✓ | 40.5 |
| VITS + TT2 + FS2 | **B** + **E** + **G** | ✗ | 38.5 |
| | | ✓ | 40.1 |

suring the CER of the synthesized speech, using the default speed factor of 1.0 is more favorable. Combining the results from Table 1 and Table 2, we find that VITS with a speed factor of 0.95 yields the best-synthesized data for building the unit-based S2ST system. Therefore, we report this number as a reference in the following combination experiments.

Table 3 shows the results with the combination of different TTS systems **A**-**K**, investigated in previous experiments. The experiments with "✗" are the approaches that simply combine the data from different TTS systems for training, while the experiments with "✓" are based on our proposed method in Sec. 2.2. When comparing models between Table 1 and Table 3, the results show that there are usually some improvements in S2ST by simply merging the data from different TTS systems. For example, we get 39.6 BLEU by combining data **E** and **G** from Table 3, while from Table 1, when training with only **B** or **G**, we get 37.6 and 38.3 BLEU, respectively. Noted that there are also cases that the simple combination of data does not improve the S2ST performances (e.g., **A** + **B** + **C** versus **A**, **B**, and **C**). Meanwhile, compared to models without multi-targets training, we would get even better results by adopting the framework proposed in Sec. 2.2.

**Table 4**. S2ST Performances from different branches of the proposed framework. The BLEU Diff. is the absolute difference between the system trained on corresponding single TTS systems. The details are discussed in Sec. 3.3.

| Category | Branch(es) | BLEU(↑) | BLEU Diff. |
|---|---|---|---|
| VITS + TT2 + FS2 | **B** | 37.9 | +0.6 |
| | **E** | 38.8 | +1.2 |
| | **G** | 38.9 | +0.6 |
| | **B** + **E** + **G** | 40.1 | / |
| VITS + FS2 | **E** | 38.8 | +1.2 |
| | **G** | 39.0 | +0.7 |
| | **E** + **G** | 40.5 | / |
| VITS + Speed Factors | **J** | 39.5 | +0.8 |
| | **G** | 39.1 | +0.8 |
| | **K** | 39.0 | +0.7 |
| | **J** + **G** + **K** | 41.5 | / |

As introduced in Sec. 2.2, given an utterance during inference, the proposed framework starts with predicting the first token (e.g., [Y] or [N]) of each decoder branch. Then, it selects the decoder branch with the highest probability of token [Y] as the decoder branch for this utterance. In Table 4, we conduct an ablation study about the inference based on the special token. To be specific, we report the S2ST performances of each decoder branch from the proposed framework and compare the results with S2ST systems trained only on the corresponding TTS targets. Three S2ST systems with the top three performances are chosen in our comparison. The results show that the S2ST performances get improved for all the branches when compared to systems trained on a single TTS system (e.g., models trained with **E** or **G** from Table 1, and models trained with **J**, **G**, **K** from Table 2). Meanwhile, it also shows the effectiveness of the proposed inference procedure because the combination of branches with our proposed method still outperforms the single-branch performances.

## 4. CONCLUSION

This work first investigates the effect of using different targets from different TTS systems. Experiments show that simply combining the target speech from TTS systems could help the learning of S2ST, especially the speech-to-unit model (S2UT). Following the findings, we propose a new framework to integrate multiple TTS targets into the S2ST modeling. Experiments demonstrate that our proposed framework can consistently improve the performances of the best baseline S2ST by 2.8 BLEU.

## 5. ACKNOWLEDGEMENT

This work was supported by a Meta AI SRA grant. Jiatong Shi and Shinji Watanabe are funded in part of the Bridges system [38], which is supported by NSF award number ACI-1445606, at the Pittsburgh Supercomputing Center (PSC).

## 6. REFERENCES


[1] Enrique Vidal, "Finite-state speech-to-speech translation," in *ICASSP*, 1997.

[2] Shigeki Matsuda, Xinhui Hu, Yoshinori Shiga, et al., "Multilingual speech-to-speech translation system: Voicetra," in *ICMDM*, 2013.

[3] Quoc Truong Do, Tomoki Toda, Graham Neubig, , et al., "Preserving word-level emphasis in speech-to-speech translation," *TASLP*, 2016.

[4] Ye Jia, Ron J Weiss, Fadi Biadsy, et al., "Direct speech-to-speech translation with a sequence-to-sequence model," *Interspeech*, 2019.

[5] Ye Jia, Michelle Tadmor Ramanovich, Tal Remez, , et al., "Translatotron 2: High-quality direct speech-to-speech translation with voice preservation," in *ICML*, 2022.

[6] Ye Jia, Yifan Ding, Ankur Bapna, et al., "Leveraging unsupervised and weakly-supervised data to improve direct speech-to-speech translation," in *Interspeech*, 2022, pp. 1721–1725.

[7] Ann Lee, Peng-Jen Chen, Changhan Wang, et al., "Direct speech-to-speech translation with discrete units," in *ACL*, 2022.

[8] Takatomo Kano, Sakriani Sakti, and Satoshi Nakamura, "Transformer-based direct speech-to-speech translation with transcoder," in *SLT*, 2021.

[9] Chen Zhang, Xu Tan, Yi Ren, et al., "Uwspeech: Speech to speech translation for unwritten languages," in *AAAI*, 2021.

[10] Xutai Ma, Hongyu Gong, Danni Liu, et al., "Direct simultaneous speech to speech translation," *arXiv preprint arXiv:2110.08250*, 2021.

[11] Sravya Popuri, Peng-Jen Chen, Changhan Wang, et al., "Enhanced Direct Speech-to-Speech Translation Using Self-supervised Pre-training and Data Augmentation," in *Interspeech*, 2022, pp. 5195–5199.

[12] Ann Lee, Hongyu Gong, Paul-Ambroise Duquenne, et al., "Textless speech-to-speech translation on real data," in *NAACL*, 2022, pp. 860–872.

[13] Genichiro Kikui, Eiichiro Sumita, Toshiyuki Takezawa, et al., "Creating corpora for speech-to-speech translation," in *Eurospeech*, 2003.

[14] Ye Jia, Michelle Tadmor Ramanovich, et al., "CVSS corpus and massively multilingual speech-to-speech translation," in *LREC*, 2022, pp. 6691–6703.

[15] Pedro Jeuris and Jan Niehues, "Libris2s: A german-english speech-to-speech translation corpus," in *LREC*, 2022, pp. 928–935.

[16] Jonathan Shen, Ye Jia, Mike Chrzanowski, et al., "Non-attentive Tacotron: Robust and controllable neural TTS synthesis including unsupervised duration modeling," *arXiv preprint arXiv:2010.04301*, 2020.

[17] Yi Ren, Chenxu Hu, Xu Tan, et al., "Fastspeech 2: Fast and high-quality end-to-end text to speech," in *ICLR*, 2020.

[18] Matt Post, Gaurav Kumar, Adam Lopez, et al., "Fisher and CALLHOME spanish–english speech translation," *LDC2014T23. Web Download. Philadelphia: Linguistic Data Consortium*, 2014.

[19] Shu-wen Yang, Po-Han Chi, Yung-Sung Chuang, et al., "SUPERB: Speech processing universal performance benchmark," in *Proc. Interspeech 2021*, 2021, pp. 1194–1198.

[20] Hsiang-Sheng Tsai, Heng-Jui Chang, Wen-Chin Huang, et al., "SUPERB-SG: Enhanced speech processing universal performance benchmark for semantic and generative capabilities," in *ACL*, 2022, pp. 8479–8492.

[21] Adam Polyak, Yossi Adi, Jade Copet, et al., "Speech resynthesis from discrete disentangled self-supervised representations," in *Interspeech*, 2021.

[22] Wen-Chin Huang, Shu-Wen Yang, Tomoki Hayashi, et al., "S3PRL-VC: Open-source voice conversion framework with self-supervised speech representations," in *ICASSP*, 2022.

[23] Tomoki Hayashi and Shinji Watanabe, "Discretalk: Text-to-speech as a machine translation problem," *arXiv preprint arXiv:2005.05525*, 2020.

[24] Kushal Lakhotia, Eugene Kharitonov, Wei-Ning Hsu, et al., "On generative spoken language modeling from raw audio," *TACL*, 2021.

[25] PD Aguero, Jordi Adell, and Antonio Bonafonte, "Prosody generation for speech-to-speech translation," in *ICASSP*, 2006.

[26] Wei-Ning Hsu, Benjamin Bolte, Yao-Hung Hubert Tsai, et al., "HuBERT: Self-supervised speech representation learning by masked prediction of hidden units," *TASLP*, 2021.

[27] Jonathan Shen, Ruoming Pang, Ron J Weiss, et al., "Natural TTS synthesis by conditioning Wavenet on Mel spectrogram predictions," in *ICASSP*, 2018.

[28] Jaehyeon Kim, Jungil Kong, and Juhee Son, "Conditional variational autoencoder with adversarial learning for end-to-end text-to-speech," in *ICML*, 2021.

[29] Ryuichi Yamamoto, Eunwoo Song, and Jae-Min Kim, "Parallel WaveGAN: A fast waveform generation model based on generative adversarial networks with multi-resolution spectrogram," in *ICASSP*, 2020.

[30] Jungil Kong, Jaehyeon Kim, and Jaekyoung Bae, "HiFi-GAN: Generative adversarial networks for efficient and high fidelity speech synthesis," *NeurIPS*, 2020.

[31] Ahmed Mustafa, Nicola Pia, and Guillaume Fuchs, "Stylemelgan: An efficient high-fidelity adversarial vocoder with temporal adaptive normalization," in *ICASSP*, 2021.

[32] Keith Ito and Linda Johnson, "The LJ speech dataset," 2017.

[33] Yinhan Liu, Jiatao Gu, Naman Goyal, et al., "Multilingual denoising pre-training for neural machine translation," *TACL*, 2020.

[34] Tomoki Hayashi, Ryuichi Yamamoto, Katsuki Inoue, et al., "ESPnet-TTS: Unified, reproducible, and integratable open source end-to-end text-to-speech toolkit," in *ICASSP*, 2020.

[35] Tomoki Hayashi, Ryuichi Yamamoto, Takenori Yoshimura, et al., "ESPnet2-TTS: Extending the edge of TTS research," *arXiv preprint arXiv:2110.07840*, 2021.

[36] Ilya Loshchilov and Frank Hutter, "Decoupled weight decay regularization," in *ICLR*, 2018.

[37] Matt Post, "A call for clarity in reporting BLEU scores," in *Conference on Machine Translation*, 2018.

[38] Nicholas A Nystrom, Michael J Levine, Ralph Z Roskies, et al., "Bridges: a uniquely flexible hpc resource for new communities and data analytics," in *XSEDE*, 2015.