# LRSpeech: Extremely Low-Resource Speech Synthesis and Recognition

Jin Xu[1], Xu Tan[2], Yi Ren[3], Tao Qin[2], Jian Li[1], Sheng Zhao[4], Tie-Yan Liu[2]
[1]Institute for Interdisciplinary Information Sciences, Tsinghua University, China, [3]Zhejiang University, China
[2]Microsoft Research Asia, [4]Microsoft Azure Speech
j-xu18@mails.tsinghua.edu.cn,xuta@microsoft.com,rayeren@zju.edu.cn
lijian83@mail.tsinghua.edu.cn,{taoqin,sheng.zhao,tyliu}@microsoft.com

## ABSTRACT

Speech synthesis (text to speech, TTS) and recognition (automatic speech recognition, ASR) are important speech tasks, and require a large amount of text and speech pairs for model training. However, there are more than 6,000 languages in the world and most languages are lack of speech training data, which poses significant challenges when building TTS and ASR systems for extremely low-resource languages. In this paper, we develop LRSpeech, a TTS and ASR system under the extremely low-resource setting, which can support rare languages with low data cost. LRSpeech consists of three key techniques: 1) pre-training on rich-resource languages and fine-tuning on low-resource languages; 2) dual transformation between TTS and ASR to iteratively boost the accuracy of each other; 3) knowledge distillation to customize the TTS model on a high-quality target-speaker voice and improve the ASR model on multiple voices. We conduct experiments on an experimental language (English) and a truly low-resource language (Lithuanian) to verify the effectiveness of LRSpeech. Experimental results show that LRSpeech 1) achieves high quality for TTS in terms of both intelligibility (more than 98% intelligibility rate) and naturalness (above 3.5 mean opinion score (MOS)) of the synthesized speech, which satisfy the requirements for industrial deployment, 2) achieves promising recognition accuracy for ASR, and 3) last but not least, uses extremely low-resource training data. We also conduct comprehensive analyses on LRSpeech with different amounts of data resources, and provide valuable insights and guidances for industrial deployment. We are currently deploying LRSpeech into a commercialized cloud speech service to support TTS on more rare languages.

**ACM Reference Format:**
Jin Xu[1], Xu Tan[2], Yi Ren[3], Tao Qin[2], Jian Li[1], Sheng Zhao[4], Tie-Yan Liu[2]. 2020. LRSpeech: Extremely Low-Resource Speech Synthesis and Recognition. In *Proceedings of the 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '20), August 23–27, 2020, Virtual Event, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3394486.3403331

[1]This work was conducted at Microsoft. Correspondence to: Tao Qin <taoqin@microsoft.com>.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
*KDD '20, August 23–27, 2020, Virtual Event, CA, USA*
© 2020 Association for Computing Machinery.
ACM ISBN 978-1-4503-7998-4/20/08…$15.00
https://doi.org/10.1145/3394486.3403331

## 1 INTRODUCTION

Speech synthesis (text to speech, TTS) [28, 30, 35, 41] and speech recognition (automatic speech recognition, ASR) [6, 10, 11] are two key tasks in speech domain, and attract a lot of attention in both the research and industry community. However, popular commercialized speech services (e.g., Microsoft Azure, Google Cloud, Nuance, etc.) only support dozens of languages for TTS and ASR, while there are more than 6,000 languages in the world [21]. Most languages are lack of speech training data, which makes it difficult to support TTS and ASR for these rare languages, as large-amount and high-cost speech training data are required to ensure good accuracy for industrial deployment.

We describe the typical training data to build TTS and ASR systems as follows:

- TTS aims to synthesize intelligible and natural speech from text sequences, and usually needs single-speaker high-quality recordings that are collected in professional recording studio. To improve the pronunciation accuracy, TTS also requires a pronunciation lexicon to convert the character sequence into phoneme sequence as the model input (e.g., "speech" is converted into "s p iy ch"), which is called as grapheme-to-phoneme conversion [36]. Additionally, TTS models use text normalization rules to convert the irregular word into the normalized type that is easier to pronounce (e.g., "Sep 7th" is converted into "September seventh").
- ASR aims to generate correct transcripts (text) from speech sequences, and usually requires speech data from multiple speakers in order to generalize to unseen speakers during inference. The multi-speaker speech data in ASR do not need to be as high-quality as that in TTS, but the data amount is usually an order of magnitude bigger. We call the speech data for ASR as multi-speaker low-quality data[1]. Optionally, ASR can first recognize the speech into phoneme sequence, and further convert it into character sequence with the pronunciation lexicon as in TTS.
- Besides paired speech and text data, TTS and ASR models can also leverage unpaired speech and text data to further improve the performance.

### 1.1 Related Work

According to the data resource used, previous works on TTS and ASR can be categorized into rich-resource, low-resource and unsupervised settings.

As shown in Table 1, we list the data resources and the corresponding related works in each setting :

[1]The low quality here does not mean the quality of ASR data is very bad, but is just relatively low compared to the high-quality TTS recordings.

| Setting | | Rich-Resource | Low-Resource | Extremely Low-Resource | Unsupervised |
|---|---|---|---|---|---|
| Data | pronunciation lexicon | ✓ | ✓ | × | × |
| | paired data (single-speaker, high-quality) | dozens of hours | dozens of minutes | several minutes | × |
| | paired data (multi-speaker, low-quality) | hundreds of hours | dozens of hours | several hours | × |
| | unpaired speech (single-speaker, high-quality) | ✓ | dozens of hours | × | × |
| | unpaired speech (multi-speaker, low-quality) | ✓ | ✓ | dozens of hours | ✓ |
| | unpaired text | ✓ | ✓ | ✓ | ✓ |
| Related Work | TTS | [22, 28, 30, 35] | [2, 12, 23, 31] | Our Work | / |
| | ASR | [6, 10, 11] | [16, 32, 33, 39] | | [8, 24, 45] |

**Table 1: The data resource to build TTS and ASR systems and the corresponding related works in rich-resource, low-resource, extremely low-resource and unsupervised settings.**

- In the rich-resource setting, both TTS [22, 28, 30, 35] and ASR [6, 10, 11] require a large amount of paired speech and text data to achieve high accuracy: TTS usually needs dozens of hours of single-speaker high-quality recordings, while ASR requires at least hundreds of hours multiple-speaker low-quality data. Besides, TTS in the rich-resource setting also leverages pronunciation lexicon for accurate pronunciation. Optionally, unpaired speech and text data can be leveraged.
- In the low-resource setting, the single-speaker high-quality paired data are reduced to dozens of minutes in TTS [2, 12, 23, 31] while the multi-speaker low-quality paired data is reduced to dozens of hours in ASR [16, 32, 33, 39], compared to that in the rich-resource setting. Additionally, they leverage unpaired speech and text data to ensure the performance.
- In the unsupervised setting, only unpaired speech and text data are leverage to build ASR models [8, 24, 45].

As can be seen, a large amount of data resources are leveraged in the rich-resource setting to ensure the accuracy for industrial deployment. Considering nearly all low-resource languages are lack of training data and there are more than 6,000 languages in the world, it will be a huge cost for training data collection. Although data resource can be reduced in the low-resource setting, it still requires 1) a certain amount of paired speech and text (dozens of minutes for TTS and dozens of hours for ASR), 2) a pronunciation lexicon, and 3) a large amount of single-speaker high-quality unpaired speech data that still incur high data collection cost[2]. What is more, the accuracy of the TTS and ASR models in the low-resource setting is not high enough. The purely unsupervised methods for ASR suffer from low accuracy and cannot meet the requirement of industrial deployment.

## 1.2 Our Method

In this paper, we develop LRSpeech, a TTS and ASR system under the extremely low-resource setting, which supports rare languages with low data collection cost. LRSpeech aims for industrial deployment under two constraints: 1) extremely low data collection cost, and 2) high accuracy to satisfy the deployment requirement. For the first constraint, as the extremely low-resource setting shown in Table 1, LRSpeech explores the limits of data requirements by 1) using single-speaker high-quality paired data as few as possible (several minutes), 2) using a few multi-speaker low-quality paired data (several hours), 3) using slightly more multi-speaker low-quality unpaired speech data (dozens of hours), 4) not using single-speaker high-quality unpaired data, and 5) not using the pronunciation lexicon but directly taking character as the input of TTS and the output of ASR.

For the second constraint, LRSpeech leverages several key techniques including transfer learning from rich-resource languages, iterative accuracy boosting between TTS and ASR through dual transformation, and knowledge distillation to further refine TTS and ASR models for better accuracy. Specifically, LRSpeech consists of a three-stage pipeline:

- We first pre-train both TTS and ASR models on rich-resource languages with plenty of paired data, which can learn the alignment capability between speech and text and benefit the alignment learning on low-resource languages.
- We further leverage dual transformation between TTS and ASR to iteratively boost the accuracy of each other with unpaired speech and text data.
- Furthermore, we leverage knowledge distillation with unpaired speech and text data to customize the TTS model on a high-quality target-speaker voice and improve the ASR model on multiple voices.

## 1.3 Data Cost and Accuracy

Next, we introduce the extremely low data cost while promising accuracy achieved by LRSpeech.

According to [4, 14, 15, 38, 43], the pronunciation lexicon, single-speaker high-quality paired data and single-speaker high-quality unpaired speech data require much higher collection cost than other data such as multi-speaker low-quality unpaired speech data and unpaired text, since they can be crawled from the web. Accordingly, compared to the low-resource setting in Table 1, LRSpeech 1) removes the pronunciation lexicon, 2) reduces the single-speaker high-quality paired data by an order of magnitude, 3) removes single-speaker high-quality unpaired speech data, 4) also reduces multi-speaker low-quality paired data by an order of magnitude, 5) similarly leverages multi-speaker low-quality unpaired speech, and 6) additionally leverage paired data from rich-resource languages which incur no additional cost since they are already available in the commercialized speech service. Therefore, LRSpeech can greatly reduce the data collection cost for TTS and ASR.

[2] Although we can crawl the multi-speaker low-quality unpaired speech data from the web, it is hard to crawl the single-speaker high-quality unpaired speech data. Therefore, it has the same collection cost (recorded by human) with the single-speaker high-quality paired data.

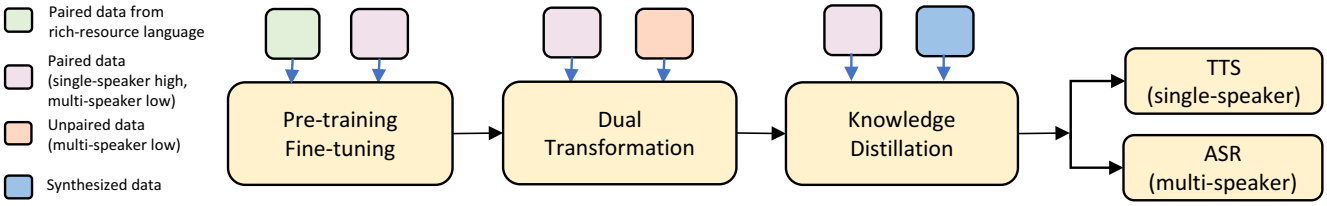



**Figure 1: The three-stage pipeline of LRSpeech.**

To verify the effectiveness of LRSpeech under the extremely low-resource setting, we first conduct comprehensive experimental studies on English and then verify on the truly low-resource language: Lithuanian, which is for product deployment. For TTS, LRSpeech achieves 98.08% intelligibility rate, 3.57 MOS score, with 0.48 gap to the ground-truth recordings, satisfying the online deployment requirements[3]. For ASR, LRSpeech achieves 28.82% WER and 14.65% CER, demonstrating great potential under the extremely low-resource setting. Furthermore, we also conduct ablation studies to verify the effectiveness of each component in LRSpeech, and analyze the accuracy of LRSpeech under different data settings, which provide valuable insights for industrial deployment. Finally, we apply LRSpeech to Lithuanian and also meets the online requirement for TTS and achieves promising results on ASR. We are currently deploying LRSpeech to a commercialized speech service to support TTS for rare languages.

## 2 LRSPEECH

In this section, we introduce the details of LRSpeech for extremely low-resource speech synthesis and recognition. We first give an overview of LRSpeech, and then introduce the formulation of TTS and ASR. We further introduce each component of LRSpeech respectively, and finally describe the model structure of LRSpeech.

### 2.1 Pipeline Overview

To ensure the accuracy of TTS and ASR models under extremely low-resource scenarios, we design a three-stage pipeline for LRSpeech as shown in Figure 1:

- Pre-training and fine-tuning. We pre-train both TTS and ASR models on rich-resource languages and then fine-tune them on low-resource languages. Leveraging rich-resource languages in LRSpeech are based on two considerations: 1) a large amount of paired data on rich-resource languages are already available in the commercialized speech service, and 2) the alignment capability between speech and text in rich-resource languages can benefit the alignment learning in low-resource languages, due to the pronunciation similarity between human languages [42].
- Dual transformation. Considering the dual nature between TTS and ASR, we further leverage dual transformation [31] to boost the accuracy of each other with unpaired speech and text data.
- Knowledge distillation. To further improve the accuracy of TTS and ASR and facilitate online deployment, we leverage knowledge distillation [18, 37] to synthesize paired data to train better TTS and ASR models.

[3]According to the requirements of a commercialized cloud speech service, the intelligibility rate should be higher than 98% and the MOS score should be higher than 3.5 while the MOS gap to the ground-truth recordings should be less than 0.5.

### 2.2 Formulation of TTS and ASR

TTS and ASR are usually formulated as sequence to sequence problems [6, 41]. Denote the text and speech sequence pair $(x, y) \in D$, where $D$ is the paired text and speech corpus. Each element in the text sequence $x$ represents a phoneme or character, while each element in the speech sequence $y$ represents a frame of speech. To learn the TTS model $\theta$, a mean square error loss is used:

$$\mathcal{L}(\theta; D) = -\Sigma_{(x,y)\in D}(y - f(x;\theta))^2. \quad (1)$$

To learn the ASR model $\phi$, a negative log likelihood loss is used:

$$\mathcal{L}(\phi; D) = -\Sigma_{(y,x)\in D} \log P(x|y;\phi). \quad (2)$$

TTS and ASR models can be developed based on an encoder-attention-decoder framework [3, 25, 40], where the encoder transforms the source sequence into a set of hidden representations, and the decoder generates the target sequence autoregressively based on the source hidden representations obtained through an attention mechanism [3].

We make some notations for the data used in LRSpeech. Denote $D_{\text{rich_tts}}$ as the high-quality TTS paired data in rich-resource languages, $D_{\text{rich_asr}}$ as the low-quality ASR paired data in rich-resource languages, $D_h$ as the single-speaker high-quality paired data for target speaker, and $D_l$ as the multi-speaker low-quality paired data. Denote $X^u$ as unpaired text data while $Y^u$ as multi-speaker low-quality unpaired speech data.

Next, we introduce each component of the LRSpeech pipeline in the following subsections.

### 2.3 Pre-Training and Fine-Tuning

The key to the conversion between text and speech is to learn the alignment between the character/phoneme representations (text) and the acoustic features (speech). Since people coming from different nations and speaking different languages share similar vocal organs and thus similar pronunciations, the ability of alignment learning in one language can help the alignment in another language [19, 42]. This motivates us to transfer the TTS and ASR models that trained in rich-resource languages into low-resource languages, considering there are plenty of paired speech and text data for both TTS and ASR in rich-resource languages.

*Pre-Training.* We pre-train the TTS model $\theta$ with data corpus $D_{\text{rich_tts}}$ following Equation 1 and pre-train the ASR model $\phi$ with $D_{\text{rich_asr}}$ following Equation 2.

*Fine-Tuning.* Considering the rich-resource and low-resource languages have different phoneme/character vocabularies and speakers, we initialize the TTS and ASR models on low-resource language with all the pre-trained parameters except the phoneme/character and speaker embeddings in TTS and the phoneme/character embeddings in ASR[4] respectively. We then fine-tune the TTS model $\theta$ and ASR model $\phi$ both with the concatenation corpus of $D_h$ and $D_l$ following Equation 1 and Equation 2 respectively. During fine-tuning, we first fine-tune the character embeddings and speaker embeddings following the practice in [1, 7], and then fine-tune all parameters. It can help prevent the TTS and ASR models from overfitting on the limited paired data in a low-resource language.

## 2.4 Dual Transformation between TTS and ASR

TTS and ASR are two dual tasks and their dual nature can be explored to boost the accuracy of each other, especially in the low-resource scenarios. Therefore, we leverage dual transformation [31] between TTS and ASR to improve the ability to transform between text and speech. Dual transformation shares similar ideas with back-translation [34] in machine translation and cycle-consistency [46] in image translation, which are effective ways to leverage unlabeled data in speech, text and image domains respectively. Dual transformation works as follows:

- For each unpaired text sequence $x \in X^u$, we transform it into speech sequence using the TTS model $\theta$, and construct a pseudo corpus $D(X^u)$ to train the ASR model $\phi$ following Equation 2.
- For each unpaired speech sequence $y \in Y^u$, we transform it into text sequence using the ASR model $\phi$, and construct a pseudo corpus $D(Y^u)$ to train the TTS model $\theta$ following Equation 1.

During training, we run the dual transformation process on the fly, which means the pseudo corpus are updated in each iteration and the model can benefit from the newest data generated by each other. Next, we introduce some specific designs in dual transformation to support multi-speaker TTS and ASR.

*Multi-Speaker TTS Synthesis.* Different from [23, 31] that only support a single speaker in both TTS and ASR model, we support multi-speaker TTS and ASR in the dual transformation stage. Specifically, we randomly choose a speaker ID and synthesize speech of this speaker given a text sequence, which can benefit the training of the multi-speaker ASR model. Furthermore, the ASR model transforms multi-speaker speech into text, which can help the training of the multi-speaker TTS model.

*Levering Unpaired Speech of Unseen Speakers.* Since multiple-speaker low-quality unpaired speech data are much easier to obtain than high-quality single-speaker unpaired speech data, enabling the TTS and ASR models to utilize unseen speakers' unpaired speech in dual transformation can make our system more robust and scalable. Compared to ASR, it is more challenging for TTS to synthesize voice on unseen speakers. To this end, we split dual transformation into two phases: 1) In the first phase, we only use the unpaired speech whose speakers are seen before in the training data. 2) In the second phase, we also add the unpaired speech whose speakers are unseen in the training data. As the ASR model can naturally support unseen speakers, the pseudo paired data can be used to train and enable the TTS model with the capability to synthesize speech of new speakers.

[4]ASR model does not need speaker embeddings, and the target embeddings and the softmax matrix are usually shared in many sequence generation tasks for better accuracy [29].

## 2.5 Customization on TTS and ASR through Knowledge Distillation

The TTS and ASR models we currently have are far from ready for online deployment after dual transformation. There are several issues we need to address: 1) While the TTS model can support multiple speakers, the speech quality of our target speaker is not good enough and needs further improvement; 2) The synthesized speech by the TTS models still have word skipping and repeating issues; 3) The accuracy of the ASR model needs to be further improved. Therefore, we further leverage knowledge distillation [18, 37], which generates target sequences given source sequences as input to construct a pseudo corpus, to customize the TTS and ASR models for better accuracy.

*2.5.1 Knowledge Distillation for TTS.* The knowledge distillation process for TTS consists of three steps:

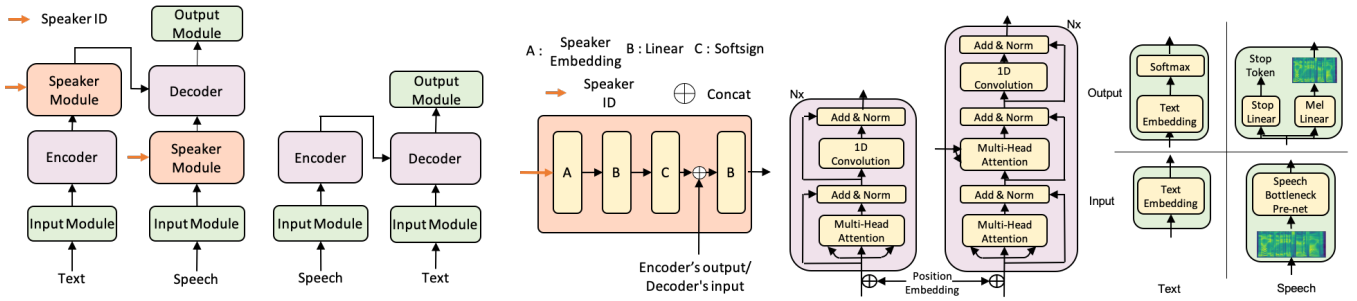
- For each unpaired text sequence $x \in X^u$, we synthesize the corresponding speech of the target speaker using the TTS model $\theta$, and construct a single-speaker pseudo corpus $D(X^u)$.
- Filter the pseudo corpus $D(X^u)$ whose synthesized speech has word skipping and repeating issues.
- Use the filtered corpus $D(X^u)$ to train a new TTS model dedicated to the target speaker following Equation 1.

In the first step, the speech in the pseudo corpus $D(X^u)$ are single-speaker, which is different from the multi-speaker pseudo corpus $D(X^u)$ in Section 2.4. The TTS model (obtained by dual transformation) in the first step has word skipping and repeating issues. Therefore, in the second step, we filter the synthesized speech which has word skipping and repeating issues, and thus the distilled model can be trained on accurate text and speech pairs. In this way, the word skipping and repeating problem can be largely reduced. We filter the synthesized speech based on two metrics: word coverage ratio (WCR) and attention diagonal ratio (ADR).

*Word Coverage Ratio.* We observe that word skipping happens when a word has small or no attention weights from the target mel-spectrograms. Therefore, we propose word coverage ratio (WCR):

$$WCR = \min_{i \in [1,N]} \{ \max_{t \in [1,T_i]} \max_{s \in [1,S]} A_{t,s} \}, \tag{3}$$

where $N$ is the number of words in a sentence, $T_i$ is the number of characters in the $i$-th word, $S$ is the number of frames of the target mel-spectrograms, and $A_{t,s}$ denotes the element in the $t$-th row and $s$-th column of the attention weight matrix $A$. We get the attention weight matrix $A$ from the encoder-decoder attention weights in the TTS model and calculate the mean over different layers and attention heads. A high WCR indicates all words in a sentence have high attention weights from target speech frames, and thus is less likely to cause word skipping.




(a) TTS model (b) ASR model (c) Speaker module (d) Encoder (left) and Decoder (right) (e) Input/Output module for speech/text

**Figure 2: The Transformer based TTS and ASR models in LRSpeech.**

*Attention Diagonal Ratio.* As demonstrated by previous works [30, 41], the attention alignments between text and speech are monotonic and diagonal. When the synthesized speech has word skipping and repeating issues, or is totally crashed, the attention alignments will deviate from the diagonal. We define the attention diagonal ratio (ADR) as:

$$ADR = \frac{\sum_{t=1}^{T} \sum_{s=kt-b}^{kt+b} A_{t,s}}{\sum_{t=1}^{T} \sum_{s=1}^{S} A_{t,s}}, \tag{4}$$

where $T$ and $S$ are the number of characters and speech frames in a text and speech pair, $k = \frac{S}{T}$, and $b$ is a hyperparameter to determine the width of diagonal. ADR measures how much attention lies in the diagonal area with a width of $b$. A higher ADR indicates that the synthesized speech has good attention alignment with text and thus has less word skipping, repeating or crashing issues.

*2.5.2 Knowledge Distillation for ASR.* Since the unpaired text and low-quality multi-speaker unpaired speech are both available for ASR , we leverage both the ASR and TTS models to synthesize data during the knowledge distillation for ASR:

- For each unpaired speech $y \in Y^u$, we generate the corresponding text using the ASR model $\phi$, and construct a pseudo corpus $D(Y^u)$.
- For each unpaired text $x \in X^u$, we synthesize the corresponding speech of multiple speakers using the TTS model $\theta$, and construct a pseudo corpus $D(X^u)$.
- We combine the above pseudo corpus $D(Y^u)$ and $D(X^u)$, as well as the single-speaker high-quality paired data $D_h$ and multi-speaker low-quality paired data $D_l$ to train a new ASR model following Equation 2.

Similar to the knowledge distillation for TTS, we also leverage a large amount of unpaired text to synthesize speech. To further improve the ASR accuracy, we use SpecAugment [27] to add noise in the input speech which acts like data augmentation.

## 2.6 Model Structure of LRSpeech

In this section, we introduce the model structure of LRSpeech, as shown in Figure 2.

*Transformer Model.* Both the TTS and ASR models adopt the Transformer based encoder-attention-decoder structure [40]. One difference from the original Transformer model is that we replace the feed-forward network with a one-dimensional convolution network following [31], in order to better capture the dependencies in a long speech sequence.

*Input/Output Module.* To enable the Transformer model to support ASR and TTS, we need different input and output modules for speech and text [31]. For the TTS model: 1) The input module of the encoder is a character/phoneme embedding lookup table, which converts character/phoneme ID into embedding; 2) The input module of the decoder is a speech pre-net, which consists of multiple dense layers to transform each speech frame non-linearly; 3) The output module of the decoder consists of a linear layer to convert hidden representations into mel-spectrograms, and a stop linear layer with a sigmoid function to predict whether current step should stop or not. For the ASR model: 1) The input module of the encoder consists of multiple convolutional layers, which reduce the length of the speech sequence; 2) The input module of the decoder is a character/phoneme embedding lookup table; 3) The output module of the decoder consists of a linear layer and a softmax function, where the linear layer shares the same weights with the character/phoneme embedding lookup table in the decoder input module.

*Speaker Module.* The multi-speaker TTS model relies on a speaker embedding module to differentiate multiple speakers. We add a speaker embedding vector both in the encoder output and decoder input (after the decoder input module). As shown in Figure 2 (c), we convert the speaker ID into a speaker embedding vector using an embedding lookup table, and then add a linear transformation with a softsign function $x = x/(1+|x|)$. We further concatenate the obtained vector with the encoder output or decoder input, and use another linear layer to reduce the hidden dimension to the original hidden of the encoder output or decoder input.

# 3 EXPERIMENTS AND RESULTS

In this section, we conduct experiments to evaluate LRSpeech for extremely low-resource TTS and ASR. We first describe the experiment settings, show the results of our method, and conduct some analyses of LRSpeech.

| Notation | Quality | Type | Dataset | #Samples |
|---|---|---|---|---|
| $D_h$ | High | Paired | LJSpeech [17] | 50 (5 minutes) |
| $D_l$ | Low | Paired | LibriSpeech [26] | 1000 (3.5 hours) |
| $Y^u_{\text{seen}}$ | Low | Unpaired | LibriSpeech | 2000 (7 hours) |
| $Y^u_{\text{unseen}}$ | Low | Unpaired | LibriSpeech | 5000 (14 hours) |
| $X^u$ | / | Unpaired | news-crawl | 20000 |

**Table 2: The data used in the low-resource language: English. $D_h$ represents target-speaker high-quality paired data. $D_l$ represents multi-speaker low-quality paired data (50 speakers). $Y^u_{\text{seen}}$ represents multi-speaker low-quality unpaired speech data (50 speakers), where speakers are seen in the paired training data. $Y^u_{\text{unseen}}$ represents multi-speaker low-quality unpaired speech data (50 speakers), where speakers are unseen in the paired training data. $X_u$ represents unpaired text data.**

## 3.1 Experimental Setup

*3.1.1 Datasets.* We describe the datasets used in rich-resource and low-resource languages respectively:

- We select Mandarin Chinese as the rich-resource language. The TTS corpus $D_{\text{rich_tts}}$ contains 10000 paired speech and text data (12 hours) of a single speaker from Data Baker[5]. The ASR corpus $D_{\text{rich_asr}}$ is from AIShell [5], which contains about 120000 paired speech and text data (178 hours) from 400 Mandarin Chinese speakers.
- We select English as a low-resource language for experimental development. The details of the data resources used are shown in Table 2. More information about these datasets are shown in Section A.1 and Table 6.

*3.1.2 Training and Evaluation.* We use a 6-layer encoder and a 6-layer decoder for both the TTS and ASR models. The hidden size, character embedding size, and speaker embedding size are all set to 384, and the number of attention heads is set to 4. During dual transformation, we up-sample the paired data to make its size roughly the same with the unpaired data. During knowledge distillation, we filter the synthesized speech with WCR less than 0.7 and ADR less than 0.7. The width of diagonal ($b$) in ADR is 10. More model training details are introduced in Section A.2.

The TTS model uses Parallel WaveGAN [44] as the vocoder to synthesize speech. To train Parallel WaveGAN, we combine the speech data in the Mandarin Chinese TTS corpus $D_{\text{rich_tts}}$ with the speech data in the English target-speaker high-quality corpus $D_h$. We up-sample the speech data in $D_h$ to make it roughly the same with the speech data in $D_{\text{rich_tts}}$.

For evaluation, we use MOS (mean opinion score) and IR (intelligibility rate) for TTS, and WER (word error rate) and CER (character error rate) for ASR. For TTS, we select English text sentences from the news-crawl[6] dataset to synthesize speech for evaluation. We randomly select 200 sentences for IR test and 20 sentences for MOS test, following the practice in [30, 41][7]. Each speech is listened by at least 5 testers for IR test and 20 testers for MOS test, who are all native English speakers. For ASR, we measure the WER and CER score on the LibriSpeech "test-clean" set. The test sentences and speech for TTS and ASR do not appear in the training corpus.

[5]https://www.data-baker.com/open_source.html
[6]http://data.statmt.org/news-crawl
[7]The sentences for IR and MOS test, audio samples and test reports can be founded in https://speechresearch.github.io/lrspeech.

## 3.2 Results

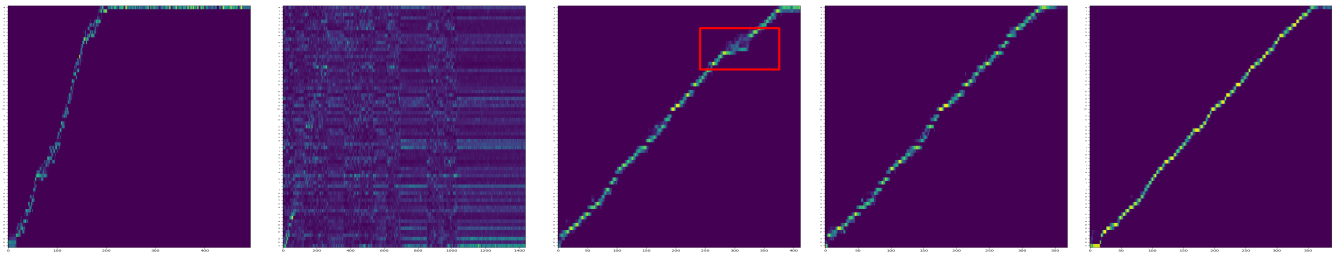
| Setting | TTS | | ASR | |
|---|---|---|---|---|
| | IR (%) | MOS | WER (%) | CER (%) |
| Baseline #1 | / | / | 148.29 | 100.16 |
| Baseline #2 | / | / | 122.09 | 97.91 |
| +PF | 93.09 | 2.84 | 103.70 | 69.53 |
| +PF+DT | 96.70 | 3.28 | 38.94 | 19.99 |
| +PF+DT+KD (LRSpeech) | **98.08** | **3.57** | **28.82** | **14.65** |
| GT (Parallel WaveGAN) | - | 3.88 | - | - |
| GT | - | 4.05 | - | - |

**Table 3: The accuracy comparisons for TTS and ASR. PF, DT and KD are the three components of LRSpeech, where PF represents pre-training and fine-tuning, DT represents dual transformation, KD represents knowledge distillation. GT is the ground-truth and GT (Parallel WaveGAN) is the audio generated with Parallel WaveGAN from the ground-truth mel-spectrogram. Baseline #1 and #2 are two baseline methods with limited paired data.**

*3.2.1 Main Results.* We compare LRSpeech with the baselines that purely leverage the limited paired data for training, including 1) Baseline #1, which trains TTS and ASR model only with corpus $D_h$, and 2) Baseline #2, which adds additional corpus $D_l$ on Baseline #1 for TTS and ASR model training. We also conduct experiments to analyze the effectiveness of each component (pre-training and fine-tuning, dual transformation, knowledge distillation) in LRSpeech. The results are shown in Table 3. We have several observations:

- Both baselines cannot synthesize reasonable speech and the corresponding IR and MOS are marked as "/". The WER and CER on ASR are also larger than 100%[8], which demonstrates the poor quality when only using the limited paired data $D_h$ and $D_l$ for TTS and ASR training.
- Based on Baseline #2, pre-training and fine-tuning (PF) can achieve an IR score of 93.09% and a MOS score of 2.84 for TTS, and reduce the WER to 103.70% and CER to 69.53%, which demonstrates the effectiveness of cross-lingual pre-training for TTS and ASR.
- However, the paired data in both rich-resource and low-resource languages cannot guarantee high accuracy, and thus we further leverage the unpaired speech corpus $Y^u_{\text{seen}}$ and $Y^u_{\text{unseen}}$, and unpaired text corpus $X^u$ through dual transformation (DT). DT can greatly improve IR to 96.70% and MOS to 3.28 on TTS, as well as WER to 38.94% and CER to 19.99%. The unpaired text and speech samples can cover more words and pronunciations, as well as more speech prosody, which help the synthesized speech in TTS achieves higher intelligibility (IR) and naturalness (MOS), and also help ASR achieves better WER and CER.

[8]The WER and CER can be larger than 100%, and the detailed reasons can be founded in Section A.3.



(a) Baseline #1 (b) Baseline #2 (c) + PF (d) + PF + DT (e) + PF + DT + KD

**Figure 3: The TTS attention alignments (where the column and row represent the source text and target speech respectively) of an example chosen from the test set. The source text is "the paper's author is alistair evans of monash university in australia".**

| Setting | WCR | ADR (%) |
|---|---|---|
| PF | 0.65 | 97.85 |
| PF + DT | 0.66 | 98.37 |
| PF + DT + KD (LRSpeech) | 0.72 | 98.81 |

**Table 4: The word coverage ratio (WCR) and attention diagonal ratio (ADR) scores in TTS model under different settings.**

- Furthermore, adding knowledge distillation (KD) brings 1.38% IR, 0.29 MOS, 10.12% WER and 5.34% CER improvements. We also list the speech quality in terms of MOS for the ground-truth recordings (GT) and the synthesized speech from the ground-truth mel-spectrogram by Parallel WaveGAN vocoder (GT (Parallel WaveGAN)) in Table 3 as the upper bounds for references. It can be seen that LRSpeech achieves a MOS score of 3.57, with a gap to the ground-truth recordings less than 0.5, demonstrating the high quality of the synthesized speech.
- There are also some related works focusing on low-resource TTS and ASR, such as Speech Chain [39], Almost Unsup [31], and SeqRQ-AE [23]. However, these methods require much data resource to build systems and thus cannot achieve reasonable accuracy in the extremely low-resource setting. For example, [31] requires a pronunciation lexicon to convert the character sequence into phoneme sequence, and dozens of hours of single-speaker high-quality unpaired speech data to improve the accuracy, which are costly and not available in the extremely low-resource setting. As a result, [31] cannot synthesize reasonable speech in TTS and achieves high WER according to our preliminary experiments.

As a summary, LRSpeech achieves an IR score of 98.08% and a MOS score of 3.57 for TTS with extremely low data cost, which meets the online requirements for deploying the TTS system. Besides, it also achieves a WER score of 28.82% and a CER score of 14.65%, which is highly competitive considering the data resource used, and shows great potential for further online deployment.

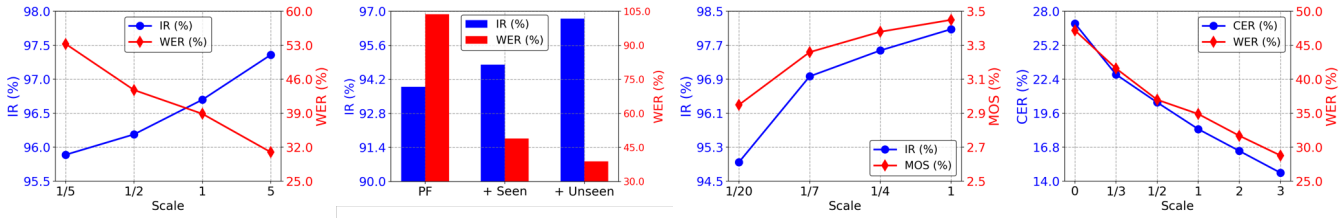
*3.2.2 Analyses on the Alignment Quality of TTS.* Since the quality of the attention alignments between the encoder (text) and decoder (speech) are good indicators of the performance of TTS model, we analyze the word coverage ratio (WCR) and attention diagonal ratio (ADR) as described in Section 2.5.1 and show their changes among different settings in Table 4. We also show the attention alignments of a sample case from each setting in Figure 3. We have several observations:

- As can be seen from Figure 3 (a) and (b), both Baseline #1 and #2 achieve poor attention alignments and their synthesized speech samples are crashed (ADR is smaller than 0.5). The attention weights of Baseline #2 are almost randomly assigned and the synthesized speech is crashed, which demonstrates that simply adding a few low-quality multi-speaker data ($D_l$) on $D_h$ cannot help the TTS model but make it worse. Due to the poor alignment quality of Baseline #1 and #2, we do not analyze their corresponding WCR.
- After adding pre-training and fine-tuning (PF), the attention alignments in Figure 3 (c) become diagonal, which demonstrates the TTS model pre-training in rich-resource languages can help build reasonable alignments between text and speech in low-resource languages. Although the synthesized speech can be roughly understood by humans, it still has many issues such as word skipping and repeating. For example, the word "in" in the red box of Figure 3 (c) has low attention weight (WCR), and thus the speech skips the word "in".
- Further adding dual transformation (DT) improves WCR and ADR, and also alleviates the words skipping and repeating issues. Accordingly, the attention alignments in Figure 3 (d) are better.
- Since there still exist some word skipping and repeating issues after DT, we filter the synthesized speech according to WCR and ADR during knowledge distillation (KD). The final WCR is further improved to 0.72 and ADR is improved to 98.81% as shown in Table 4, and the attention alignments in Figure 3 (e) are much more clear.

## 3.3 Further Analyses of LRSpeech

There are some questions to further investigate in LRSpeech:

- Low-quality speech data may bring noise to the TTS model. How can the accuracy change if using different scales of low-quality paired data $D_l$?
- As described in Section 2.4, supporting the LRSpeech training with unpaired speech data from seen and especially unseen speakers ($Y^u_{\text{seen}}$ and $Y^u_{\text{unseen}}$) is critical for a robust and scalable system. Can the accuracy be improved if using $Y^u_{\text{seen}}$ and $Y^u_{\text{unseen}}$?
- How can the accuracy change if using different scales of unpaired text data $X^u$ to synthesize speech during knowledge distillation?




(a) Varying the data scale of $D_l$ (b) Results using $Y^u_{\text{seen}}$ and $Y^u_{\text{unseen}}$ (c) Varying the data $X^u$ for TTS knowledge distillation (d) Varying the data $X^u$ for ASR knowledge distillation

**Figure 4: Analyses of LRSpeech with different training data.**

We conduct experimental analyses to answer these questions. For the first two questions, we simply analyze LRSpeech without knowledge distillation, and for the third question, we analyze in the knowledge distillation stage. The results are shown in Figure 4[9]. We have several observations:

- As shown in Figure 4 (a), we vary the size of $D_l$ with 1/5×, 1/2× and 5× of the default setting (1000 paired data, 3.5 hours) used in LRSpeech, and find that more low-quality paired data result in the better accuracy for TTS.
- As shown in Figure 4 (b), we add $Y^u_{\text{seen}}$ and $Y^u_{\text{unseen}}$ respectively, and find that both of them can boost the accuracy of TTS and ASR, which demonstrates the ability of LRSpeech to utilize unpaired speech from seen and especially unseen speakers.
- As shown in Figure 4 (c), we vary the number of synthesized speech for TTS during knowledge distillation with 1/20×, 1/7× and 1/4× of the default setting (20000 synthesized speech data), and find more synthesized speech data result in better accuracy.
- During knowledge distillation for ASR, we use two kinds of data: 1) the realistic speech data (8050 data in total), which contains $D_h$, $D_l$ and the pseudo paired data distilled from $Y^u_{\text{seen}}$ and $Y^u_{\text{unseen}}$ by the ASR model, 2) the synthesized speech data, which are the pseudo paired data distilled from $X^u$ by the TTS model. We vary the number of synthesized speech data from $X^u$ (the second type) with 0×, 1/3×, 1/2×, 1×, 2×, 3× of the realistic speech data (the first type) in Figure 4 (d). It can be seen that increasing the ratio of synthesized speech data can achieve better results.

All the observations above demonstrate the effectiveness and scalability of LRSpeech by leveraging more low-cost data resources.

### 3.4 Apply to Truly Low-Resource Language: Lithuanian

*Data Setting.* The data setting in Lithuanian is similar to that in English. We select a subset of Liepa corpus [20] and only use the characters as the raw texts. The $D_h$ contains 50 paired text and speech data (3.7 minutes), $D_l$ contains 1000 paired text and speech data (1.29 hours), $Y^u_{\text{seen}}$ contains 4000 unpaired speech data (5.1 hours), $Y^u_{\text{unseen}}$ contains 5000 unpaired speech data (6.7 hours), and $X^u$ contains 20000 unpaired texts.

We select Lithuanian text sentences from the news-crawl dataset as the test set for TTS. We randomly select 200 sentences for IR test and 20 sentences for MOS test, following the same test configuration in English. Each audio is listened by at least 5 testers for IR test and 20 testers for MOS test, who are all native Lithuanian speakers. For ASR evaluation, we randomly select 1000 speech data (1.3 hours) with 197 speakers from Liepa corpus to measure the WER and CER scores. The test sentences and speech for TTS and ASR do not appear in the training corpus.

*Results.* As shown in Table 5, the TTS model on Lithuanian achieves an IR score of 98.60% and a MOS score of 3.65, with a MOS gap to the ground-truth recording less than 0.5, which also meets the online deployment requirement[10]. The ASR model achieves a CER score of 10.30% and a WER score of 17.04%, which shows great potential under this low-resource setting.

| Setting | IR (%) | MOS | WER (%) | CER (%) |
|---|---|---|---|---|
| Lithuanian | 98.60 | 3.65 | 17.04 | 10.30 |
| GT (Parallel WaveGAN) | - | 3.89 | - | - |
| GT | - | 4.01 | - | - |

**Table 5: The results of LRSpeech on TTS and ASR with regard to Lithuanian.**

## 4 CONCLUSION

In this paper, we developed LRSpeech, a speech synthesis and recognition system under the extremely low-resource setting, which supports rare languages with low data costs. We proposed pre-training and fine-tuning, dual transformation and knowledge distillation in LRSpeech to leverage few paired speech and text data, and slightly more multi-speaker low-quality unpaired speech data to improve the accuracy of TTS and ASR models. Experiments on English and Lithuanian show that LRSpeech can meet the requirements of online deployment for TTS and achieve very promising results for ASR under the extremely low-resource setting, demonstrating the effectiveness of LRSpeech for rare languages.

Currently we are deploying LRSpeech to a large commercialized cloud TTS service. In the future, we will further improve the accuracy of ASR in LRSpeech and also deploy it to this commercialized cloud service.

## 5 ACKNOWLEDGEMENTS

Jin Xu and Jian Li are supported in part by the National Natural Science Foundation of China Grant 61822203, 61772297, 61632016, 61761146003, and the Zhongguancun Haihua Institute for Frontier

[9]The audio samples and complete experiments results on IR and MOS for TTS, and WER and CER for ASR can be founded in https://speechresearch.github.io/lrspeech.

[10]The audio samples can be founded in https://speechresearch.github.io/lrspeech

Information Technology, Turing AI Institute of Nanjing and Xi'an Institute for Interdisciplinary Information Core Technology.

## REFERENCES


[1] Mikel Artetxe, Sebastian Ruder, and Dani Yogatama. 2019. On the cross-lingual transferability of monolingual representations. *arXiv preprint arXiv:1910.11856* (2019).

[2] Alexei Baevski, Michael Auli, and Abdelrahman Mohamed. 2019. Effectiveness of self-supervised pre-training for speech recognition. *arXiv preprint arXiv:1911.03912* (2019).

[3] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473* (2014).

[4] Antoine Bruguier, Anton Bakhtin, and Dravyansh Sharma. 2018. Dictionary Augmented Sequence-to-Sequence Neural Network for Grapheme to Phoneme Prediction. *Proc. Interspeech 2018* (2018), 3733–3737.

[5] Hui Bu, Jiayu Du, Xingyu Na, Bengu Wu, and Hao Zheng. 2017. Aishell-1: An open-source mandarin speech corpus and a speech recognition baseline. In *2017 20th Conference of the Oriental Chapter of the International Coordinating Committee on Speech Databases and Speech I/O Systems and Assessment (O-COCOSDA)*. IEEE, 1–5.

[6] William Chan, Navdeep Jaitly, Quoc Le, and Oriol Vinyals. 2016. Listen, attend and spell: A neural network for large vocabulary conversational speech recognition. In *Acoustics, Speech and Signal Processing (ICASSP), 2016 IEEE International Conference on*. IEEE, 4960–4964.

[7] Tian-Yi Chen, Lan Zhang, Shi-Cong Zhang, Zi-Long Li, and Bai-Chuan Huang. 2019. Extensible cross-modal hashing. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*. AAAI Press, 2109–2115.

[8] Yi-Chen Chen, Chia-Hao Shen, Sung-Feng Huang, and Hung-yi Lee. 2018. Towards Unsupervised Automatic Speech Recognition Trained by Unaligned Speech and Text only. *arXiv preprint arXiv:1803.10952* (2018).

[9] Yuan-Jui Chen, Tao Tu, Cheng-chieh Yeh, and Hung-Yi Lee. 2019. End-to-end text-to-speech for low-resource languages by cross-lingual transfer learning. *Proc. Interspeech 2019* (2019), 2075–2079.

[10] Chung-Cheng Chiu, Tara N Sainath, Yonghui Wu, Rohit Prabhavalkar, Patrick Nguyen, Zhifeng Chen, Anjuli Kannan, Ron J Weiss, Kanishka Rao, Ekaterina Gonina, et al. 2018. State-of-the-art speech recognition with sequence-to-sequence models. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 4774–4778.

[11] Jan Chorowski, Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. End-to-end continuous speech recognition using attention-based recurrent nn: First results. In *NIPS 2014 Workshop on Deep Learning, December 2014*.

[12] Yu-An Chung, Yuxuan Wang, Wei-Ning Hsu, Yu Zhang, and RJ Skerry-Ryan. 2019. Semi-supervised training for improving data efficiency in end-to-end speech synthesis. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 6940–6944.

[13] Erica Cooper, Emily Li, and Julia Hirschberg. 2018. Characteristics of Text-to-Speech and Other Corpora. *Proceedings of Speech Prosody 2018* (2018).

[14] Erica Lindsay Cooper. 2019. *Text-to-speech synthesis using found data for low-resource languages*. Ph.D. Dissertation. Columbia University.

[15] Joel Harband. 2010. Text-to-Speech Costs — Licensing and Pricing. http://elearningtech.blogspot.com/2010/11/text-to-speech-costs-licensing-and.html

[16] Takaaki Hori, Ramon Astudillo, Tomoki Hayashi, Yu Zhang, Shinji Watanabe, and Jonathan Le Roux. 2019. Cycle-consistency training for end-to-end speech recognition. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 6271–6275.

[17] Keith Ito. 2017. The LJ Speech Dataset. https://keithito.com/LJ-Speech-Dataset/.

[18] Yoon Kim and Alexander M Rush. 2016. Sequence-Level Knowledge Distillation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. 1317–1327.

[19] Patricia K Kuhl, Barbara T Conboy, Sharon Coffey-Corina, Denise Padden, Maritza Rivera-Gaxiola, and Tobey Nelson. 2008. Phonetic learning as a pathway to language: new data and native language magnet theory expanded (NLM-e). *Philosophical Transactions of the Royal Society B: Biological Sciences* 363, 1493 (2008), 979–1000.

[20] Sigita Laurinčiukaitė, Laimutis Telksnys, Pijus Kasparaitis, Regina Kliukienė, and Vilma Paukštytė. 2018. Lithuanian Speech Corpus Liepa for development of human-computer interfaces working in voice recognition and synthesis mode. *Informatica* 29, 3 (2018), 487–498.

[21] M Paul Lewis and F Gary. 2013. Simons, and Charles D. Fennig (eds.).(2015). Ethnologue: Languages of the World, Dallas, Texas: SIL International. *Online version: http://www. ethnologue. com* (2013).

[22] Naihan Li, Shujie Liu, Yanqing Liu, Sheng Zhao, Ming Liu, and M Zhou. 2019. Neural Speech Synthesis with Transformer Network. AAAI.

[23] Alexander H Liu, Tao Tu, Hung-yi Lee, and Lin-shan Lee. 2019. Towards Unsupervised Speech Recognition and Synthesis with Quantized Speech Representation Learning. *arXiv preprint arXiv:1910.12729* (2019).

[24] Da-Rong Liu, Kuan-Yu Chen, Hung-yi Lee, and Lin-shan Lee. 2018. Completely Unsupervised Phoneme Recognition by Adversarially Learning Mapping Relationships from Audio Embeddings. *Proc. Interspeech 2018* (2018), 3748–3752.

[25] Minh-Thang Luong, Hieu Pham, and Christopher D Manning. 2015. Effective Approaches to Attention-based Neural Machine Translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. 1412–1421.

[26] 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)*. IEEE, 5206–5210.

[27] Daniel S Park, William Chan, Yu Zhang, Chung-Cheng Chiu, Barret Zoph, Ekin D Cubuk, and Quoc V Le. 2019. SpecAugment: A Simple Data Augmentation Method for Automatic Speech Recognition. *Proc. Interspeech 2019* (2019), 2613–2617.

[28] Wei Ping, Kainan Peng, Andrew Gibiansky, Sercan O. Arik, Ajay Kannan, Sharan Narang, Jonathan Raiman, and John Miller. 2018. Deep Voice 3: 2000-Speaker Neural Text-to-Speech. In *International Conference on Learning Representations*.

[29] Ofir Press and Lior Wolf. 2017. Using the Output Embedding to Improve Language Models. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*. 157–163.

[30] Yi Ren, Yangjun Ruan, Xu Tan, Tao Qin, Sheng Zhao, Zhou Zhao, and Tie-Yan Liu. 2019. Fastspeech: Fast, robust and controllable text to speech. In *Advances in Neural Information Processing Systems*. 3165–3174.

[31] Yi Ren, Xu Tan, Tao Qin, Sheng Zhao, Zhou Zhao, and Tie-Yan Liu. 2019. Almost Unsupervised Text to Speech and Automatic Speech Recognition. In *International Conference on Machine Learning*. 5410–5419.

[32] Andrew Rosenberg, Yu Zhang, Bhuvana Ramabhadran, Ye Jia, Pedro Moreno, Yonghui Wu, and Zelin Wu. 2019. Speech Recognition with Augmented Synthesized Speech. *arXiv preprint arXiv:1909.11699* (2019).

[33] Steffen Schneider, Alexei Baevski, Ronan Collobert, and Michael Auli. 2019. wav2vec: Unsupervised Pre-Training for Speech Recognition. *Proc. Interspeech 2019* (2019), 3465–3469.

[34] Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Improving Neural Machine Translation Models with Monolingual Data. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Vol. 1. 86–96.

[35] Jonathan Shen, Ruoming Pang, Ron J Weiss, Mike Schuster, Navdeep Jaitly, Zongheng Yang, Zhifeng Chen, Yu Zhang, Yuxuan Wang, Rj Skerrv-Ryan, et al. 2018. Natural tts synthesis by conditioning wavenet on mel spectrogram predictions. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 4779–4783.

[36] Hao Sun, Xu Tan, Jun-Wei Gan, Hongzhi Liu, Sheng Zhao, Tao Qin, and Tie-Yan Liu. 2019. Token-Level Ensemble Distillation for Grapheme-to-Phoneme Conversion. *Proc. Interspeech 2019* (2019), 2115–2119.

[37] Xu Tan, Yi Ren, Di He, Tao Qin, and Tie-Yan Liu. 2019. Multilingual Neural Machine Translation with Knowledge Distillation. In *International Conference on Learning Representations*. https://openreview.net/forum?id=S1gUsoR9YX

[38] Ye Kyaw Thu, Win Pa Pa, Yoshinori Sagisaka, and Naoto Iwahashi. 2016. Comparison of grapheme-to-phoneme conversion methods on a myanmar pronunciation dictionary. In *Proceedings of the 6th Workshop on South and Southeast Asian Natural Language Processing (WSSANLP2016)*. 11–22.

[39] Andros Tjandra, Sakriani Sakti, and Satoshi Nakamura. 2017. Listening while speaking: Speech chain by deep learning. In *Automatic Speech Recognition and Understanding Workshop (ASRU), 2017 IEEE*. IEEE, 301–308.

[40] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*. 5998–6008.

[41] Yuxuan Wang, RJ Skerry-Ryan, Daisy Stanton, Yonghui Wu, Ron J Weiss, Navdeep Jaitly, Zongheng Yang, Ying Xiao, Zhifeng Chen, Samy Bengio, et al. 2017. Tacotron: Towards End-to-End Speech Synthesis. *Proc. Interspeech 2017* (2017), 4006–4010.

[42] Jan Wind. 1989. The evolutionary history of the human speech organs. *Studies in language origins* 1 (1989), 173–197.

[43] Junichi Yamagishi, Bela Usabaev, Simon King, Oliver Watts, John Dines, Jilei Tian, Yong Guan, Rile Hu, Keiichiro Oura, Yi-Jian Wu, et al. 2010. Thousands of voices for HMM-based speech synthesis–Analysis and application of TTS systems built on various ASR corpora. *IEEE Transactions on Audio, Speech, and Language Processing* 18, 5 (2010), 984–1004.

[44] Ryuichi Yamamoto, Eunwoo Song, and Jae-Min Kim. 2019. Parallel WaveGAN: A fast waveform generation model based on generative adversarial networks with multi-resolution spectrogram. *arXiv preprint arXiv:1910.11480* (2019).

[45] Chih-Kuan Yeh, Jianshu Chen, Chengzhu Yu, and Dong Yu. 2019. Unsupervised Speech Recognition via Segmental Empirical Output Distribution Matching. *ICLR* (2019).

[46] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. 2017. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision*. 2223–2232.

# A REPRODUCIBILITY

## A.1 Datasets

We list the detailed information of all of the datasets used in this paper in Table 6. Next, we first describe the details of the data preprocessing for speech and text data, and then describe what is the "high-quality" and "low-quality" speech mentioned in this paper[11].

*Data Proprocessing.* For the speech data, we re-sample it to 16kHZ and convert the raw waveform into mel-spectrograms following Shen et al. [35] with 50ms frame size, 12.5ms hop size. For the text, we use text normalization rules to convert the irregular word into the normalized type which is easier to pronounce, e.g., "Sep 7th" will be converted into "September seventh".

*High-Quality Speech.* We use high-quality speech to refer the speech data from TTS corpus (e.g., LJSpeech, Data Baker as shown in Table 6), which are usually recorded in a professional recording studio with consistent characteristics such as speaking rate. Collecting high-quality speech data for TTS is typically costly [13].

*Low-Quality Speech.* We use low-quality speech to refer the speech data from ASR corpus (e.g., LibriSpeech, AIShell, Liepa as shown in Table 6). Compared to high-quality speech, low-quality speech usually contains noise due to the recording devices (e.g., smartphones, laptops) or the recording environment (e.g., room reverberation, traffic noise). However, low-quality speech cannot be too noisy for model training. We just use the term "low-quality" to differ from high-quality speech.

## A.2 Model Configurations and Training

Both the TTS and ASR models use the 6-layer encoder and 6-layer decoder. For both the TTS and ASR models, the hidden size and speaker ID embedding size is 384 and the number of attention heads is 4. The kernel sizes of 1D convolution in the 2-layer convolution network are set to 9 and 1 respectively, with input/output size of 384/1536 for the first layer and 1536/384 in the second layer. For the TTS model, the input module of the decoder consists of 3 fully-connected layers. The first two fully-connected layers have 64 neurons each and the third one has 384 neurons. The ReLU non-linearity is applied to the output of every fully-connected layer. We also insert 2 dropout layers in between the 3 fully-connected layers, with dropout probability 0.5. The output module of the decoder is a fully-connected layer with 80 neurons. For the ASR model, the encoder contains 3 convolution layers. The first two are $3 \times 3$ convolution layers with stride 2 and filter size 256, and the third one with stride 1 and filter size 256. The ReLU non-linearity is applied to the output of every convolution layer except the last one.

We implement LRSpeech based on the tensor2tensor codebase[12]. We use the Adam optimizer with $\beta_1 = 0.9$, $\beta_2 = 0.98$, $\epsilon = 10^{-9}$ and follow the same learning rate schedule in Vaswani et al. [40]. We train both the TTS and ASR models in LRSpeech on 4 NVIDIA V100 GPUs. Each batch contains 20,000 speech frames in total. The pre-training and fine-tuning, dual transformation and knowledge distillation take nearly 1, 7, 1 days respectively. We measure the TTS inference speed on a server with 12 Intel Xeon CPU, 256GB memory, 1 NVIDIA V100 GPU. The TTS model takes about 0.21s to generate 1.0s of speech, which satisfies online deployment requirements for inference speed.

[11] We show some high-quality speech (target speaker) and low-quality speech (other speakers) from the training set in the demo page: https://speechresearch.github.io/lrspeech.

[12] https://github.com/tensorflow/tensor2tensor

## A.3 Evaluation Details

*Mean Opinion Score (MOS).* The MOS test is a speech quality test for naturalness where listeners (testers) were asked to give their opinions on the speech quality in a five-point scale MOS: 5=excellent, 4=good, 3=fair, 2=poor, 1=bad. We randomly select 20 sentences to synthesize speech for MOS test and each audio is listened by 20 testers, who are all native speakers. We present a part of the MOS test results in Figure 5. The complete test report can be downloaded here[13].

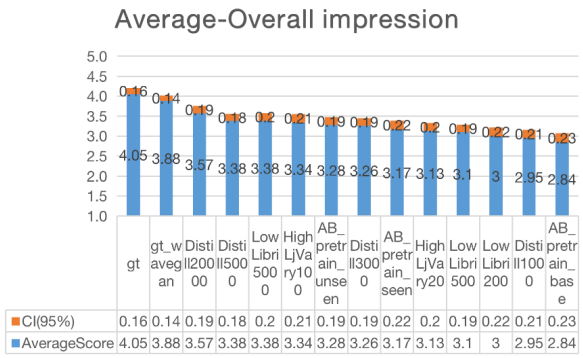



**Figure 5: A part of the English MOS test report.**

*Intelligibility Rate (IR).* The IR test is a speech quality test for Intelligibility. During the test, the listeners (testers) are requested to mark every unintelligible word in the text sentence. IR is calculated by the proportion of the words that are intelligible over the total test words. We randomly select 200 sentences to synthesize speech for IR test and each audio is listened by 5 testers, who are all native speakers. A part of the IR test results is shown in Figure 6. You can find more test reports from the demo link.

*WER and CER.* Given the reference text and predicted text, the WER calculates the edit distance between them and then normalizes the distance by dividing the number of words in the reference sentence. The WER is defined as $WER = \frac{S+D+I}{N}$, where the $N$ is the number of words in the reference sentence, $S$ is the number of substitutions, $D$ is the number of deletions and $I$ is the number of insertions. The WER can be larger than 100%. For example, given the reference text "an apple" and predicted text "what is history", the predicted text needs two substitution operations and one insertion operation. For this case, the WER is $\frac{2+1}{2}$=150%. The CER is similar to WER.

## A.4 Some Explorations in Experiments

We briefly describe some other explorations in training LRSpeech in this paper:

[13] https://speechresearch.github.io/lrspeech

| Dataset | Type | Speakers | Language | Open Source | Usage |
|---|---|---|---|---|---|
| Data Baker | High-quality speech | Single | Mandarin Chinese | ✓ | Pre-training |
| AIShell | Low-quality speech | Multiple | Mandarin Chinese | ✓ | Pre-training |
| LJSpeech | High-quality speech | Single | English | ✓ | Training |
| LibriSpeech | Low-quality speech | Multiple | English | ✓ | Training / Engish ASR test |
| Liepa | Low-quality speech | Multiple | Lithuanian | ✓ | Training / Lithuanian ASR test |
| news-crawl | Text | / | English/Lithuanian | ✓ | English/Lithuanian training and TTS test |

**Table 6: The datasets used in this paper.**

| Setting | Result |
|---|---|
| Reference | some mysterious force seemed to have brought about a convulsion of the elements |
| Baseline #1 | in no characters is the contrast between the ugly and vulgar illegibility of the modern type |
| Baseline #2 | the queen replied in a careless tone for instance now she went on |
| + PF | some of these ceriase for seen to him to have both of a down of the old lomests |
| + PF + DT | some misterious force seemed to have brought about a convulsion of the elements |
| + PF + DT + KD (LRSpeech) | some mysterious force seemed to have brought about a convulsion of the elements |

**Table 7: A case analysis for ASR model under different settings.**

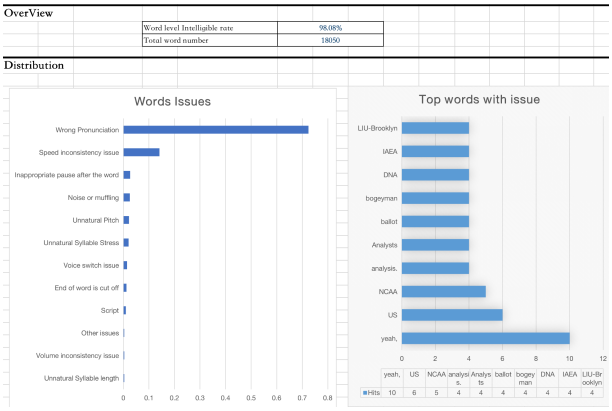



**Figure 6: A part of the English IR test report.**

- **Pre-training and Finetune** We also try different methods such as unifying the character spaces between rich-resource and low-resource languages, or learning the mapping between the character embeddings of rich- and low-resource languages as used in [9]. However, we find these methods result in similar accuracy for both TTS and ASR.
- **Speaker Module** To design the speaker module, we explore several ways including replacing softsign with ReLU, etc. Experimental results show that the design as Figure 2 (c) can help model reduce the repeating words and missing words.
- **Knowledge Distillation for TTS** We try to add the paired target speaker data for training. However, the result is slightly worse than that using only synthesized speech.
- **Knowledge Distillation for ASR** Since the synthesized speech can improve the performance, we try to remove the real speech and add plenty of synthesized speech for training. However, the ASR model cannot work well for real speech and WER is above 47%.
- **Vocoder Training** In our preliminary experiments, we only use the dataset $D_{\text{rich_tts}}$ in the rich-resource language (Mandarin Chinese) to train the Parallel WaveGAN. The vocoder can generate high-quality speech for Mandarin Chinese but fail to work for the low-resource languages. Considering that the vocoder has not been trained on the speech in the low-resource languages, we add single-speaker high-quality corpus $D_h$ and up-sample the speech data in $D_h$ to make it roughly the same with the speech data in $D_{\text{rich_tts}}$ for training. In this way, we find that the vocoder can work well for the low-resource languages.

## A.5 Case Analyses for ASR

We also conduct a case analysis on ASR as shown in Table 7. Please refer to Section 3.2 for the descriptions of each setting in this table. The generated text by Baseline #1 is completely irrelevant to the reference. Besides, we find from the test set that the generated text is usually the same for many completely different speech, due to the lack of paired data (only 50 paired data) for training. Baseline #2 can generate different text sentences for different speech, but still cannot generate reasonable results. After pre-training and fine-tuning (PF), the model can recognize some words like "have". Similar to the effect of pre-training on TTS, pre-training ASR on rich-resource language can also help to learn the alignment between speech and text. By further leveraging unpaired speech and text, with dual transformation (DT), the generated sentence is more accurate. However, for some hard words like "mysterious", the model cannot recognize it correctly. TTS and ASR can also help each other not only in dual transformation but also in knowledge distillation (KD). During KD, a large amount of pseudo paired data generated from the TTS model can help the ASR model recognize most words and give correct results as shown in Table 7.