External Text Based Data Augmentation for Low-Resource Speech Recognition in the Constrained Condition of OpenASR21 Challenge

Guolong Zhong[1], Hongyu Song[1], Ruoyu Wang[1], Lei Sun[2], Diyuan Liu[2], Jia Pan[2], Xin Fang[2], Jun Du[1], Jie Zhang[1], Lirong Dai[1]

[1]University of Science and Technology of China, Hefei, Anhui, P. R. China
[2]iFlytek Research, Hefei, Anhui, P. R. China

{cndragon, hongyusong}@mail.ustc.edu.cn, {jundu, jzhang6, lrdai}@ustc.edu.cn

Abstract

This paper describes our USTC_NELSLIP system submitted to the Open Automatic Speech Recognition (OpenASR21) Challenge for the Constrained condition, where only a 10-hour speech dataset is allowed for training while additional text data is unlimited. To improve the low-resource speech recognition performance, we collect external text data for language modeling and train a text-to-speech (TTS) model to generate speech-text paired data. Our system is then built based on the conventional hybrid structure, where various subsystems are developed using different acoustic neural network architectures and different data augmentation methods. Finally, system fusion is employed to obtain the final result. Experiments on the OpenASR21 challenge show that the proposed system achieves the best performance for all testing languages.

Index Terms: OpenASR21, low-resource languages, speech recognition, data augmentation, TTS, system fusion

1. Introduction

The deep learning based Automatic Speech Recognition (ASR) methods often require a large amount of annotated data, but there are many languages in the world that suffer from the insufficiency of annotated data [1]. The goal of the Open Automatic Speech Recognition (OpenASR21) Challenge, organized by the National Institute of Standards and Technology (NIST), is to assess the state of the art of ASR techniques for low-resource languages. The OpenASR21 Challenge [2] consists of ASR tasks for 15 low-resource languages with three different training conditions, i.e., *Constrained*, *Constrained-plus*, and *Unconstrained*. For the Constrained condition, the speech dataset allowed for training is only a 10-hour Build dataset provided by NIST for each language, while additional text data from publicly available resources are permissible during training. The Constrained-plus condition follows the same training data restrictions as the Constrained case, but publicly available and previously existing speech pre-trained models are allowed. For the Unconstrained condition, participants are allowed to use all publicly available speech and text data in any language.

For low-resource languages, the scarcity of transcribed data hampers the performance of the ASR model. To deal with this problem, data augmentation is widely used to increase the amount and the diversity of training data, for instance, speed and volume perturbation [3], reverberation and noise perturbation [4], vocal tract length perturbation [5], and SpecAug-

---

This paper presents a detailed description of USTC-NELSLIP system, which was submitted to the Constrained condition of OpenASR21 Challenge and obtained the first place for all testing languages among all submissions.

ment [6]. Although these methods are effective to some extent, they only increase the variations of existing acoustic signals or features [7]. One can also use text-to-speech (TTS) techniques to synthesize speech from the text-only data, which can be done by either training a TTS model independently for data augmentation [8, 9, 10] or jointly training both the ASR and TTS models [11, 12].

Compared with speech transcriptions, text-only data are easier to obtain. It is essential to optimize the language model (LM) to improve ASR performance. The LM can be trained using additional text-only data and then integrated into the ASR system [13, 14]. Nevertheless, there are usually only limited in-domain text data available to low-resource languages. To augment the LM training corpus, one can collect text from the web followed by carefully data filtering [15]. Another approach is to generate text that is relevant to the target domain [16].

In the OpenASR21 Challenge, we participated in the most challenging Constrained condition for all 15 languages. Since publicly available text data is unlimited, in this paper we propose to leverage external text data not only for language modeling but also for augmenting the acoustic training data. For LM, we collect as much in-domain text as possible and train a domain classifier to filter the text obtained from the web to reduce the domain mismatch. The LM trained on the filtered web-obtained text is combined with the in-domain LM by interpolation to obtain a better LM. For acoustic training, we train a TTS model to synthesize speech data on the external text, which introduces more acoustic and linguistic diversity to improve the system robustness. Based on these, we build a hybrid Deep Neural Network-Hidden Markov Model (DNN-HMM) based ASR system, where five acoustic neural network architectures are adopted for acoustic modeling. We also use the encoder of end-to-end (E2E) model as a feature extractor to build a supplementary hybrid system. The final ASR system turns out to be a combination of subsystems trained with different architectures and different data.

The remainder of this paper is organized as follows. Section 2 describes our proposed system for the Constrained condition in detail, followed by the experimental setup and results analysis in Section 3. Finally, Section 4 concludes this work.

2. System description

In the low-resource case, building a pure E2E ASR system might not be an appropriate choice due to the overfitting problem. Thus, we adopt the hybrid DNN-HMM based structure for the Constrained condition, as shown in Figure 1. To improve the performance of the system, we explore the use of external text to augment both acoustic and language model training data. Other data augmentation methods are also explored to increase

Figure 1: *The workflow of the proposed Constrained system.*

there still exist a chunk of silences and some noisy portions in the audio, data clean-up [22] is performed using the GMM-HMM model to re-segment the training data. We denote the data after clean-up as *cleaned* data.

We use the 3-way speed perturbation [3] and SpecAugment [6] as general steps in our system. We also explore another way to perturb the speeds of utterances by forcing their lengths to some allowed lengths spaced by a factor of 1.15 [23]. As this data augmentation procedure is time-consuming, we only apply it to a supplementary model to the system fusion.

As it was shown in [24] that the representations learned by the transformer encoders can contain high-level semantic information effectively, we use the transformer encoder as a feature extractor to build a hybrid model. We directly train an encoder-decoder (ED) ASR model, which is based on a VGG-transformer [25]. We find that the ED model is hard to converge if we train the model directly on the low-resource ASR dataset. To satisfy the great data demand in training an E2E ASR model, we apply speed perturbation to augment the training data, and then add the TTS synthesized data. Using the trained ED model, the latent representations extracted from the encoder are concatenated with the filterbank features as fused features, which are fed to the neural network of hybrid model. It will be shown by experiments that although the model trained with such fused features performs worse than the best single model, it can bring further gains at the system fusion stage.

2.3. Text-to-speech synthesis

We train a TTS model to synthesize features on the external text, which are added to the training set to mitigate overfitting during acoustic model training. The TTS model is built based on the Flow-TTS [26], which is a non-autoregressive model. The Flow-TTS consists of an encoder, a decoder, a length predictor and a positional attention layer. To enable multi-speaker training and generation, we include i-vector as speaker embedding to the encoder and each step of flow in the decoder. The i-vector extractor is trained based on a diagonal universal background model (UBM) [27]. Moreover, we adopt Gaussian up-sampling [28] to address the length mismatch between phoneme and spectrogram sequences. In the training stage, we train the length predictor and the rest of the model separately. We prepare phoneme alignments obtained by the GMM-HMM model for the 10-hour cleaned data. Using the cross-entropy (CE) loss, the length predictor takes the phoneme sequence and i-vector as inputs and returns the length of Mel-spectrograms. The rest of the model is trained with the maximum likelihood estimation objective. In the inference stage, an i-vector will be randomly selected for each sentence from utterances that were used in the training stage. To avoid serious bias towards synthesized data, we augment the cleaned data using speed perturbation and combine them with the synthesized data at a certain ratio. By including the synthesized data for acoustic model training, the overfitting problem can be alleviated.

2.4. Acoustic neural network training

For acoustic neural network training, the aforementioned GMM-HMM model is used to generate training alignment. We choose the ResNet-TDNNF architecture as the baseline acoustic model, which consists of Residual Network (ResNet) and Factorized Time Delay Neural Network (TDNNF) [29]. The ResNet contains 7 Res-blocks, e.g., see details in [30]. The TDNNF has 12 layers with a hidden dimension of 1536 and a bottleneck dimension of 160. The network takes filterbank fea-

the amount and the diversity of training data. Furthermore, we increase the acoustic model diversity, which is beneficial to the system fusion.

2.1. External text data collection

The datasets for most of the languages stem from the IARPA Babel program [17], except that Somali and Farsi stem from the IARPA MATERIAL program [18]. For the language that has a corresponding IARPA Babel language pack, we first use the pronunciation lexicon in the pack together with the transcriptions from the training set. For each language, we also collect a large amount of text data from websites. However, these web-obtained data are quite different from the provided data, which is conversational telephone speech (CTS). The out-of-vocabulary (OOV) rate of web-obtained text is very high. Thus we perform carefully data filtering for the web-obtained data to reduce the domain mismatch with the CTS data. In detail, at first the repeated sentences as well as sentences containing many repeated or OOV words are removed. Secondly, a domain classifier is trained to select the data similar to CTS. Finally, we add colloquial noise such as modal particles and reduplicated words to the selected text. The filtered text will be denoted as *Public text* in the following for brevity. The Babel text and Public text are used for both language modeling and TTS synthesis. Since pre-trained models such as BERT perform well on many downstream NLP tasks, the domain classifier is built based on a pre-trained model and then fine-tuned using all Babel data with an equal amount of web-obtained data. The pre-trained model Chinese-BERT-wwm [19] and the multilingual pre-trained model XLM-R [20] are adopted for Cantonese and other languages, respectively.

2.2. Data pre-processing and data augmentation

We segment the training data according to the timestamps provided in the training transcripts. A speaker adaptive Gaussian Mixture Model-Hidden Markov Model (GMM-HMM) model is firstly trained following the Babel recipe in Kaldi [21]. Since

tures and online i-vector features as inputs. Based on ResNet-TDNNF, we evaluate different data augmentation strategies and search for hyper-parameters for training.

We also adopt four different network architectures. Inspired by [31], we propose a ResNet-Multistream-TDNNF architecture by replacing the simple CNN with ResNet. The input features are processed by a single ResNet and then branched out into multiple parallel streams of TDNNF layers. The time-stride for the TDNNF layers in each stream is unique, which can enhance system robustness with diverse temporal resolutions. Output embedding vectors from all streams are concatenated and projected to the final layer. Besides, we compare the CNN-TDNNF [32] architecture and two architectures proposed in [30], namely ResNet-TDNNF-Attention and ResNet-TDNNF-RBiLSTM. All these models are trained using the lattice-free maximum mutual information (LF-MMI) criterion [33] with CE regularization.

2.5. Language modeling and rescoring

For the first-pass decoding, we train a trigram LM on the Babel text (Babel LM) using the SRILM [34] toolkit. We also interpolate the Babel LM with another trigram LM trained on the Public text to obtain the interpolated LM. The interpolation weight of Babel LM is set to be 0.8. For the second-pass LM rescoring, we adopt the Chinese-BERT-wwm for Cantonese and RNN-LM [35] for other languages. Due to the similar linguistic information between Cantonese and Mandarin, Chinese-BERT-wwm is more beneficial to Cantonese than RNN-LM, although RNN-LM is lighter and easier for training. The Chinese-BERT-wwm is trained for several iterations using the Cantonese Public text and then fine-tuned using Cantonese Babel text. For other languages, the RNN-LMs are initialized using the Public text and then fine-tuned using the corresponding Babel text. Each RNN-LM consists of one BLSTMP [36] layer with a hidden size of 256 and the embedding size of 256.

2.6. System fusion and post-processing

We use lattice fusion followed by minimum Bayes risk (MBR) decoding [37] to combine the recognition results obtained by different subsystems. For post-processing, we apply confidence filtering to the results obtained from lattice, where the recognized words with a confidence score below the predefined threshold are discarded.

3. Experiments and analysis

3.1. Experimental setup

The datasets provided for training (Build), development (Dev), and evaluation (Eval) are 10 hours, 10 hours, and 5 hours, respectively. All hybrid systems are implemented using Kaldi. During acoustic neural network training, the model is trained for 6 epochs with a batch size of 128 or 64. The learning rate decays from 0.001 to 0.00005. The TTS model and ED model are trained using Pytorch. Considering the very small difference between cleaned data and perturbed cleaned data, we regard both as real data. We mix the TTS data with the real data at a ratio of 1.5, where the former consists of *Babel TTS* and *Public TTS* data synthesized on Babel text and Public text, respectively. For test data, we train a TDNN-LSTM based Voice Activity Detection (VAD) model for each language using Kaldi. As the data-driven based VAD model is not always effective for some recordings, we consider an energy-based VAD method as

Table 1: *The WER (%) of the best single system and the final fusion system on 15 languages.*

Language	best single		best fusion	
	Dev	Eval	Dev	Eval
Amharic	35.0	43.1	32.0	39.9
Cantonese	42.3	41.5	38.2	37.6
Guarani	39.0	46.0	36.4	42.6
Javanese	51.9	52.8	47.8	48.1
Kurmanji-Kurdish	63.7	65.4	61.4	61.7
Mongolian	45.4	45.0	41.3	41.0
Pashto	45.2	47.5	41.4	43.2
Somali	55.9	59.1	52.7	55.6
Tamil	61.0	65.8	57.7	62.3
Vietnamese	43.9	43.6	40.9	40.3
Swahili	32.3	35.7	29.5	32.4
Tagalog	42.1	43.8	39.3	40.4
Georgian	37.5	43.0	34.9	39.2
Kazakh	46.1	54.3	40.1	50.0
Farsi	52.4	70.1	49.5	68.0

a complement. The final detection is the union of two methods. During decoding, the beam size is set to be 16. We search the LM weight from 7 to 17 at a step size of 1. The word insertion penalty is set to be 0, and the threshold of confidence filtering ranges from 0 to 0.5.

3.2. Results and analysis

Table 1 shows the WER of the best single and fusion systems, where the results on the Dev set are scored locally and the results on the Eval set are released by the OpenASR21 scoring server. All these results are obtained after LM rescoring and post-processing. We can see that compared to the best single system, the fusion system can reduce the WER on the Eval set by 2.1%-4.7%. It is worth mentioning that ResNet-Multistream-TDNNF yields the best performance in single system for Guarani, Mongolian, Vietnamese, Tagalog, Kazakh and Farsi. However, ResNet-TDNNF is the best option for other languages.

To explore the effects of training with different TTS data and decoding with different LMs, we show the first-pass results (without LM rescoring or post-processing) of different systems on the Dev set in Table 2. Without loss of generality, we show results for seven representative languages. It is clear that the Babel LM always outperforms the 10h LM on all languages, e.g., the WER is decreased by 3.6% on average. This is due to the fact that the 10-hour corpus is a subset of the Babel corpus. Moreover, using the Babel TTS data can further improve the performance for all languages with an average WER reduction of 1.2%, compared to the case without TTS data. This reveals the feasibility of using TTS data for data augmentation, even though the considered TTS model is trained with only 10-hour data. It can be seen that the interpolated LM only surpasses Babel LM for Cantonese, Mongolian and Kazakh. Using additional Public TTS data only yields better performance for a few languages such as Cantonese, Kazakh, Javanese and Farsi. This can be interpreted by the slight domain mismatch between the Public and Babel text. Besides, the proportion of Public TTS data in the training set is critical and influences the effect of using TTS data. In the evaluation period, we select the LM that achieves the lowest WER on the Dev set.

Table 2: *The WER (%) of single system using different TTS data and LMs on the Dev set. 'Baseline' corresponds to systems trained without TTS data based on the ResNet-TDNNF model. '10h LM' represents the LM trained on 10-hour transcriptions. Note that Farsi has no corresponding Babel pack, we thus temporarily treat the 10-hour text data as Babel text data and the TTS data synthesized on the 10-hour text as Babel TTS data.*

	Cantonese	Mongolian	Kazakh	Tamil	Pashto	Javanese	Farsi
The number of tokens in the Babel text (k)	884	400	267	485	877	308	64
The number of tokens in the Public text (k)	73,874	11,192	29,865	3,666	42,780	997	31,568
Baseline + 10h LM	48.6	51.2	52.9	67.5	50.2	56.5	55.5
+ Babel LM	45.7	47.9	48.7	62.4	47.1	53.8	-
+ interpolated LM	45.1	47.3	48.3	62.4	47.2	53.8	54.0
+ Babel TTS + 10h LM	47.1	50.0	52.0	66.8	48.6	55.3	55.1
+ Babel TTS + Babel LM	44.1	46.7	47.7	61.5	45.5	52.6	-
+ Babel TTS + interpolated LM	43.6	**46.2**	47.4	**61.4**	**45.5**	52.6	53.7
+ Babel & Public TTS + 10h LM	46.5	50.4	51.7	67.2	49.0	55.2	54.1
+ Babel & Public TTS + Babel LM	43.3	47.1	47.2	61.7	46.0	52.4	-
+ Babel & Public TTS + interpolated LM	**42.6**	46.4	**46.6**	61.7	45.9	**52.3**	**52.7**

Table 3: *Results of different models trained w/o or w/ TTS data on the Cantonese Dev set.*

Model	Context	WER(%)	
		w/o TTS	w/ TTS
CNN-TDNNF	144	45.9	43.5
ResNet-TDNNF	188	45.1	42.6
ResNet-TDNNF-Attention	108	46.7	45.6
ResNet-Multistream-TDNNF	344	45.9	42.8
ResNet-TDNNF-RBiLSTM	96	46.8	44.2

Table 4: *The effect of fusion strategy on the Cantonese Dev set.*

Fusion System	WER(%)
Fusion 1: REAL	42.5
Fusion 2: REAL + USP	42.2
Fusion 3: REAL + USP + SYN	40.2
Fusion 4: REAL + USP + SYN + TER	39.6
Fusion 5: $REAL_r$ + USP_r + SYN_r + TER_r	38.6
Fusion 6: Fusion 4 + Fusion 5	38.4

Table 3 shows the first-pass results of models trained without (w/o) or with (w/) TTS data for different models on the Cantonese Dev set. We can see that the TTS data are effective for all models. Moreover, adding TTS data is more beneficial for the models that have a wider temporal context, e.g., ResNet-Multistream-TDNNF, which somehow explains why ResNet-Multistream-TDNNF is the architecture of the best single system for some languages.

In Table 4, we show the effects of different fusion strategies on the Cantonese Dev set. All results are obtained before post-processing. 'REAL' refers to five models mentioned in Table 3 trained with real data. 'USP' refers to a model augmented by the utterance-level speed perturbation with a series of allowed lengths. 'SYN' refers to five models trained with TTS synthesized data. 'TER' represents a model trained with transformer encoder representations. The subscript r means that the results are obtained after the second-pass rescoring. We can see that by only using real data, the fusion of five models results in a WER reduction of 2.6% (from 45.1% to 42.5%), which is attributed to the effective complementarity of different models. It is clear that combining USP, SYN or TER can all improve the performance of the fusion system, while the positive effect of combining SYN is more obvious, e.g., the WER is decreased by 2.0% (from 42.2% to 40.2%). This is consistent with the results obtained by the single system in Table 3. Moreover, fusing the rescored results can reduce the WER on the Cantonese Dev set from 39.6% down to 38.6%. Note that such fusion is not always better than the fusion of the first-pass results for some languages, but our experiments show that combining both the

rescored and first-pass results yields a consistent improvement for all languages, e.g., the WER on the Cantonese Dev set is reduced to 38.4%. After post-processing, the WER is further reduced to 38.2%, as shown in Table 1.

4. Conclusion

In this paper, we presented a detailed description of our system submitted to the OpenASR21 challenge for the Constrained condition. We proposed to leverage external text for both language modeling and acoustic data augmentation. Experimental results showed the proposed method can improve the ASR performance under a low-resource scenario. Moreover, combining subsystems trained with different acoustic neural network architectures and different data can yield a significant improvement. Our submission ranked first for the Constrained condition in all 15 languages. Actually, we also submitted for the Unconstrained condition and won the first place in seven involved languages, where the inclusion of TTS data and the optimization of LMs were similarly considered. In the future, we will conduct more experiments and present both the Constrained and Unconstrained systems in a more complete report.

5. Acknowledgements

6. References

[1] L. Besacier, E. Barnard, A. Karpov, and T. Schultz, "Automatic speech recognition for under-resourced languages: A survey," *Speech Communication*, vol. 56, pp. 85–100, 2014.

[2] NIST, "OpenASR Challenge." [Online]. Available: https://www.nist.gov/itl/iad/mig/openasr-challenge

[3] T. Ko, V. Peddinti, D. Povey, and S. Khudanpur, "Audio augmentation for speech recognition," in *Sixteenth annual conference of the international speech communication association*, 2015.

[4] T. Ko, V. Peddinti, D. Povey, M. L. Seltzer, and S. Khudanpur, "A study on data augmentation of reverberant speech for robust speech recognition," in *IEEE Int. Conf. Acoust., Speech, Signal Process. (ICASSP)*, 2017, pp. 5220–5224.

[5] N. Jaitly and G. E. Hinton, "Vocal tract length perturbation (vtlp) improves speech recognition," in *Proc. ICML Workshop on Deep Learning for Audio, Speech and Language*, vol. 117, 2013, p. 21.

[6] D. S. Park, W. Chan, Y. Zhang, C.-C. Chiu, B. Zoph, E. D. Cubuk, and Q. V. Le, "Specaugment: A simple data augmentation method for automatic speech recognition," in *Proc. Interspeech*, 2019, pp. 2613–2617.

[7] J. Sun, Z. Tang, H. Yin, W. Wang, X. Zhao, S. Zhao, X. Lei, W. Zou, and X. Li, "Semantic Data Augmentation for End-to-End Mandarin Speech Recognition," in *Proc. Interspeech*, 2021.

[8] J. Li, R. Gadde, B. Ginsburg, and V. Lavrukhin, "Training neural speech recognition systems with synthetic speech augmentation," *CoRR*, vol. abs/1811.00707, 2018.

[9] A. Rosenberg, Y. Zhang, B. Ramabhadran, Y. Jia, P. Moreno, Y. Wu, and Z. Wu, "Speech recognition with augmented synthesized speech," in *2019 IEEE automatic speech recognition and understanding workshop (ASRU)*. IEEE, 2019, pp. 996–1002.

[10] N. Rossenbach, A. Zeyer, R. Schlüter, and H. Ney, "Generating synthetic audio data for attention-based speech recognition systems," in *IEEE Int. Conf. Acoust., Speech, Signal Process. (ICASSP)*, 2020, pp. 7069–7073.

[11] A. Tjandra, S. Sakti, and S. Nakamura, "Listening while speaking: Speech chain by deep learning," in *2017 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, 2017, pp. 301–308.

[12] T. Hori, R. Astudillo, T. Hayashi, Y. Zhang, S. Watanabe, and J. Le Roux, "Cycle-consistency training for end-to-end speech recognition," in *IEEE Int. Conf. Acoust., Speech, Signal Process. (ICASSP)*, 2019, pp. 6271–6275.

[13] C. Gulcehre, O. Firat, K. Xu, K. Cho, L. Barrault, H.-C. Lin, F. Bougares, H. Schwenk, and Y. Bengio, "On using monolingual corpora in neural machine translation," *arXiv preprint arXiv:1503.03535*, 2015.

[14] B.-J. Hsu, "Generalized linear interpolation of language models," in *2007 IEEE Workshop on Automatic Speech Recognition Understanding (ASRU)*, 2007.

[15] T. Ng, M. Ostendorf, M.-Y. Hwang, M. Siu, I. Bulyko, and X. Lei, "Web-data augmented language models for mandarin conversational speech recognition," in *IEEE Int. Conf. Acoust., Speech, Signal Process. (ICASSP)*, 2005.

[16] M. Suzuki, N. Itoh, T. Nagano, G. Kurata, and S. Thomas, "Improvements to n-gram language model using text generated from neural language model," in *IEEE Int. Conf. Acoust., Speech, Signal Process. (ICASSP)*, 2019, pp. 7245–7249.

[17] M. Harper, "The babel program and low resource speech technology," *Proceedings of the Automatic Speech Recognition and Understanding*, 2013.

[18] IARPA, "MATERIAL." [Online]. Available: https://www.iarpa.gov/index.php/research-programs/material

[19] Y. Cui, W. Che, T. Liu, B. Qin, and Z. Yang, "Pre-training with whole word masking for chinese bert," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 29, pp. 3504–3514, 2021.

[20] A. Conneau, K. Khandelwal, N. Goyal, V. Chaudhary, G. Wenzek, F. Guzmán, E. Grave, M. Ott, L. Zettlemoyer, and V. Stoyanov, "Unsupervised cross-lingual representation learning at scale," *arXiv preprint arXiv:1911.02116*, 2019.

[21] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz *et al.*, "The kaldi speech recognition toolkit," in *IEEE 2011 workshop on automatic speech recognition and understanding*, 2011.

[22] V. Manohar, D. Povey, and S. Khudanpur, "Jhu kaldi system for arabic mgb-3 asr challenge using diarization, audio-transcript alignment and transfer learning," in *2017 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, 2017, pp. 346–352.

[23] H. Hadian. [Online]. Available: https://github.com/kaldi-asr/kaldi/blob/master/egs/wsj/s5/utils/data/perturb_speed_to_allowed_lengths.py

[24] A. T. Liu, S.-W. Li, and H.-Y. Lee, "Tera: Self-supervised learning of transformer encoder representation for speech," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 29, pp. 2351–2366, 2021.

[25] A. Mohamed, D. Okhonko, and L. Zettlemoyer, "Transformers with convolutional context for asr," *arXiv preprint arXiv:1904.11660*, 2019.

[26] C. Miao, S. Liang, M. Chen, J. Ma, S. Wang, and J. Xiao, "Flow-tts: A non-autoregressive network for text to speech based on flow," in *IEEE Int. Conf. Acoust., Speech, Signal Process. (ICASSP)*, 2020, pp. 7209–7213.

[27] G. Saon, H. Soltau, D. Nahamoo, and M. Picheny, "Speaker adaptation of neural network acoustic models using i-vectors," in *2013 IEEE Workshop on Automatic Speech Recognition and Understanding*, 2013, pp. 55–59.

[28] J. Shen, Y. Jia, M. Chrzanowski, Y. Zhang, I. Elias, H. Zen, and Y. Wu, "Non-attentive tacotron: Robust and controllable neural tts synthesis including unsupervised duration modeling," *arXiv preprint arXiv:2010.04301*, 2020.

[29] D. Povey, G. Cheng, Y. Wang, K. Li, H. Xu, M. Yarmohammadi, and S. Khudanpur, "Semi-orthogonal low-rank matrix factorization for deep neural networks," in *Proc. Interspeech*, 2018, pp. 3743–3747.

[30] L. Chai, J. Du, D.-Y. Liu, Y.-H. Tu, and C.-H. Lee, "Acoustic modeling for multi-array conversational speech recognition in the chime-6 challenge," in *2021 IEEE Spoken Language Technology Workshop (SLT)*, 2021, pp. 912–918.

[31] K. J. Han, J. Pan, V. K. N. Tadala, T. Ma, and D. Povey, "Multistream cnn for robust acoustic modeling," in *IEEE Int. Conf. Acoust., Speech, Signal Process. (ICASSP)*, 2021, pp. 6873–6877.

[32] C. Zorilă, C. Boeddeker, R. Doddipatla, and R. Haeb-Umbach, "An investigation into the effectiveness of enhancement in asr training and test for chime-5 dinner party transcription," in *2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, 2019, pp. 47–53.

[33] D. Povey, V. Peddinti, D. Galvez, P. Ghahremani, V. Manohar, X. Na, Y. Wang, and S. Khudanpur, "Purely sequence-trained neural networks for asr based on lattice-free mmi," in *Proc. Interspeech*, 2016, pp. 2751–2755.

[34] A. Stolcke, "Srilm-an extensible language modeling toolkit," in *Seventh international conference on spoken language processing*, 2002.

[35] H. Xu, T. Chen, D. Gao, Y. Wang, K. Li, N. Goel, Y. Carmiel, D. Povey, and S. Khudanpur, "A pruned rnnlm lattice-rescoring algorithm for automatic speech recognition," in *IEEE Int. Conf. Acoust., Speech, Signal Process. (ICASSP)*, 2018, pp. 5929–5933.

[36] H. Sak, A. Senior, and F. Beaufays, "Long short-term memory recurrent neural network architectures for large scale acoustic modeling," *Proc. Interspeech*, pp. 338–342, 2014.

[37] H. Xu, D. Povey, L. Mangu, and J. Zhu, "Minimum bayes risk decoding and system combination based on a recursion for edit distance," *Comput. Speech Lang.*, vol. 25, no. 4, pp. 802–828, 2011.