



Knowledge Distillation Method for Pruned RNN-T Models via Pruning Bounds Sharing and Losses Confusion

Xiaocan Zhang, Weiwei Jiang, Guibin Zheng, Chenhao Jing, Jiqing Han, Tieran Zheng

Faculty of Computing, Harbin Institute of Technology, Harbin, China
{22b903093@stu., 22S003046@stu., zhengguibin@, chenhaojing@stu., jqhan@}hit.edu.cn

Abstract

Although the advantages of large models have been widely proved in speech recognition, small models are still required in several applications due to the limited computational resources or training data. The recognition accuracy of small models has always been a challenging issue. This paper proposes a distillation method for the pruned RNN-T structure to enhance the generalization ability of small models by leveraging information from large models, where the small model shares the pruning bounds of the large model as well as the decoder and connector structures, and multi-loss fusion is used to distill. Utilizing the Chinese speech dataset Aishell-1, experimental results demonstrated that the small model distilled from pre-trained large model significantly outperforms the directly trained model of the same size by a notable relative reduction of 30.4% in Character Error Rate (CER), thereby validating the effectiveness of the proposed knowledge distillation method.

Index Terms: ASR, Pruned RNN-T, knowledge distillation

1. Introduction

In the field of speech recognition, large-scale models trained using massive data have become the mainstream of research and application due to their excellent performance. The training and inference of large-scale models have high demands on computational resources. But several practical applications often face the problem of limited computational resources and scene data. Thereby small-scale models are needed.

Due to the lower training memory consumption and better training stability, the pruned RNN-T [1] architecture has become a commonly used training architecture for large models of speech recognition. However, when training the pruned RNN-T model, the range of output prediction for an input audio frame is determined by a complex calculation on input acoustic and linguistic features. During knowledge distillation using pruned RNN-T architecture, the student model and the teacher model usually use different encoders and decoders with different parameters, structures, etc. So, the same input audio frame will produce different text prediction ranges on the two models, resulting in the inability to compute the loss of the distillation prediction distribution, which is based on the premise of predicting the output range to be the same. Therefore, suitable distillation methods for pruned RNN-T architectures is needed. This paper proposes a knowledge distillation method for pruned RNN-T models, using a large model as the distillation teacher model, and a small model with a fewer encoder parameters as the student model, and the small model inheriting the parameters of the decoder and connector modules from the large model; during the training process, the results of the RNN-T pruned bounds of the large model are used for the

training of the small model directly, and the small model no longer calculates the bounds by itself, thus solving the problem that the prediction distributions of teacher and student models cannot be aligned; Meanwhile, the feature-level distillation loss at multiple frame rates are used under a fusion strategy of multiple losses along with the CTC loss [2], the RNN-T loss [3], and the prediction distribution distillation loss. The loss fusion strategy is carried out in different ways according to different stages of training according to and the model gradient.

The experimental results of the proposed method on the Aishell-1 dataset [4] demonstrated that the CER of the distilled small model was reduced by 30.48% compared to the small model trained directly. It proved that the distillation method is effective and a small model distilled from large model could be more effective than the one trained directly on same dataset.

2. Related work

Knowledge distillation (KD) was originally proposed by [5] to guide the training of student models through the output of teacher models and is often used to distill large teacher models into smaller student models [6][7][8][9].

End-to-end speech recognition methods include CTC, RNN-T, AED [10], and the pruned RNN-T method improved from RNN-T. The CTC method is premised on the Conditional Independence Assumption, which assumes that the output characters are only dependent on the current input. Whereas RNN-T introduces a decoder (or predictor) module that feeds the output information from the previous context back into the model, thereby avoiding the issue of conditional independence assumptions. The AED method, on the other hand, employs an attention mechanism that uses the previous context as the query and the acoustic features as key-value pairs to obtain the subsequent context in an autoregressive manner.

The RNN-T method considers the alignment between each acoustic and linguistic vector during the training process, and thus has the problems of slow speed, high memory usage, and unstable and difficult to converge results. The pruned RNN-T method linearly transforms the acoustic and linguistic features into the lexicon space in advance to obtain the acoustic prediction and linguistic prediction, and then computes the alignment boundaries of acoustic and linguistic features efficiently through these predictions. The pruned RNN-T is stable and easy to converge, and significantly reduces the memory usage during training.

Zipformer [11] was used as the encoder for the RNN-T architecture. Zipformer is derived from Conformer [12], a widely used encoder model in speech recognition. The Zipformer encoder employs a U-Net structure, where the embedding features of input data are extracted before the

encoder modules that are symmetrical along the temporal dimension. Each module uses a different downsampling rate to have different temporal resolutions, effectively addressing the issue of temporal alignment in audio sequences.

3. Proposed method

The overall structure of the knowledge distillation method proposed for pruned RNN-T is shown in Fig. 1. The teacher model contains three components: a decoder, a joiner, and an encoder with a large scale of parameters. The student model contains a randomly initialized encoder with fewer parameters. Both the teacher and student encoders adopt the Zipformer structure. The decoders utilize a stateless convolutional structure [13], and the joiner employs a linear layer structure. Label ① and ② in Fig. 1 indicate that the feature outputs from the teacher and student encoders are separately fed into the teacher joiner along with the outputs from the teacher decoder, thereby generating the predictive distribution respectively, i.e. teacher and student logits.

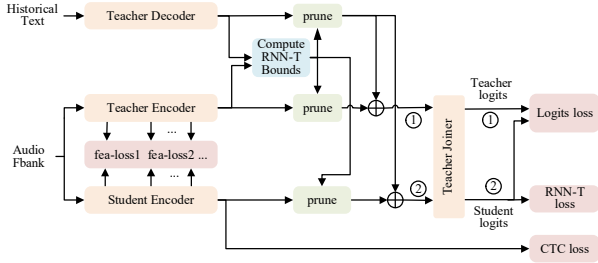


Figure 1: The overall distillation structure

3.1. Distillation sharing bounds of pruned RNN-T

In the process of distilling pruned RNN-T model, discrepancies may arise between the pruning bounds computed by the student and teacher models. As a result, the pruned outcomes based on these bounds may be different, leading to misalignment between the logits from the teacher and student models. Therefore, the logits from the teacher model cannot be used as effective labels to guide the student model. To address this issue, student model directly shares the pruning bounds of the teacher model to compute loss of logits.

The input acoustic Fbank features are converted into acoustic features by the teacher encoder $Encoder_T$:

$$\mathbf{am}_T = Encoder_T(\mathbf{X}) \quad (1)$$

where \mathbf{X} represents the input acoustic features, T denotes teacher model.

The texts are converted into textual features representation \mathbf{lm}_T by the teacher text decoder $Decoder_T$:

$$\mathbf{lm}_T = Decoder_T(\mathbf{Y}) \quad (2)$$

where \mathbf{Y} represents the text labels.

Boundaries of the pruned RNN-T, denoted as \mathbf{Bounds} , are computed based on \mathbf{am}_T and \mathbf{lm}_T , as shown in the following equation:

$$\mathbf{Bounds} = Compute_bounds(\mathbf{am}_T, \mathbf{lm}_T) \quad (3)$$

where, the function $Compute_bounds(\cdot)$ represents the bound calculation process described in [1].

Then, \mathbf{am}_T and \mathbf{lm}_T are pruned by \mathbf{Bounds} , and the result is fed into the joiner module $Joiner_T$ to obtain the model's output predictive distribution \mathbf{Logits}_T .

$$\mathbf{Pruned}_{\mathbf{am}_T}, \mathbf{Pruned}_{\mathbf{lm}_T} = Prun(\mathbf{am}_T, \mathbf{lm}_T, \mathbf{Bounds}) \quad (4)$$

$$\mathbf{Logits}_T = Joiner_T(\mathbf{Pruned}_{\mathbf{am}_T}, \mathbf{Pruned}_{\mathbf{lm}_T}, \mathbf{Bounds}) \quad (5)$$

Where the $Prun(\cdot)$ operation aligns \mathbf{am}_T and \mathbf{lm}_T based on the \mathbf{Bounds} as described in [1].

Similarly, the acoustic features are encoded by the student model encoder to obtain the student model's acoustic feature representation \mathbf{am}_S . Consequently, the \mathbf{Bounds} from the teacher model are used to prune \mathbf{am}_S to obtain $\mathbf{Pruned}_{\mathbf{am}_S}$, as shown in the following equation:

$$\mathbf{am}_S = Encoder_S(\mathbf{X}) \quad (6)$$

$$\mathbf{Pruned}_{\mathbf{am}_S} = Prun(\mathbf{am}_S, \mathbf{Bounds}) \quad (7)$$

Together with $\mathbf{Pruned}_{\mathbf{am}_S}$ and $\mathbf{Pruned}_{\mathbf{lm}_T}$, the student model's predictive distribution \mathbf{Logits}_S is computed by the teacher model's joiner module $Joiner_T$ as shown in the following equation:

$$\mathbf{Logits}_S = Joiner_T(\mathbf{Pruned}_{\mathbf{am}_S}, \mathbf{Pruned}_{\mathbf{lm}_T}) \quad (8)$$

For facilitating the computation of the distillation loss, the model predictions $\mathbf{Logits} \in \mathbb{R}^{B \times T \times S \times D}$ are reshaped to $\mathbf{Logits2D} \in \mathbb{R}^{N \times D}$, where N equals $B \times T \times S$. Here, B represents the batch size, T denotes the time steps, S indicates the span of the bounds and D is the vocabulary dimension:

$$\mathbf{Logits2D}_T = Reshape2(\mathbf{Logits}_T, D) \quad (9)$$

$$\mathbf{Logits2D}_S = Reshape2(\mathbf{Logits}_S, D) \quad (10)$$

where $Reshape2(\cdot)$ is the transformation of the output into the desired two-dimensional format.

The distillation loss is computed using the Kullback-Leibler (KL) divergence between the teacher and student predictive distributions, as Eq. (11-12):

$$D_{KL}(P \parallel Q) = \sum_i^D P(i) \log \frac{P(i)}{Q(i)} \quad (11)$$

$$Loss_{logits} = \sum_j^N D_{KL}(\mathbf{Logits2D}_T(j) \parallel \mathbf{Logits2D}_S(j)) \quad (12)$$

where $\mathbf{Logits2D}_T(j)$ and $\mathbf{Logits2D}_S(j)$ denote the predictive distribution vectors for the teacher model and student model.

Additionally, distillation loss of multi-frame-rate feature-level is introduced, using KL divergence to measure the difference between teacher and student models in feature level in order to guide the training of the encoder in student model. This process is illustrated in Fig. 2. CTC loss and RNN-T loss are also utilized to constrain the updates of model parameters using text labels.

3.2. Dynamic multi-Loss fusion

The distillation loss incorporates three components: the hard-label loss $Loss_{hard}$, the feature loss $Loss_{fea}$, and the predictive distribution loss $Loss_{logits}$.

The correct text labels are used to calculate $Loss_{CTC}$ and $Loss_{RNN-T}$ firstly, and then use the weighted sum of them as hard-label loss:

$$Loss_{hard} = Loss_{RNN-T} + \alpha \cdot Loss_{CTC} \quad (13)$$

where α is a hyperparameter.

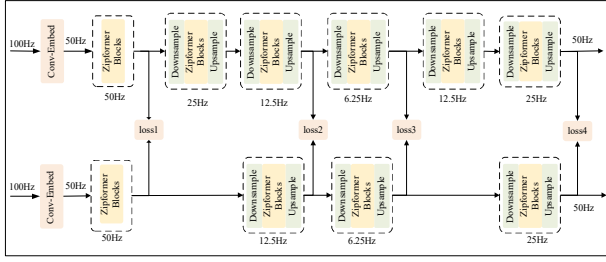


Figure 2: Multi-frame-rate feature-level distillation loss.

The soft-label losses, including the multi-frame rate feature loss $Loss_{fea}$ and the predictive distribution loss $Loss_{logits}$, helps the student model to learn both the feature representations and the output probability distribution of the teacher.

There is large discrepancy in the value scales of these three losses, and optimization direction for various losses are inconsistencies. In order to prevent some of the loss functions from being completely suppressed by the others and to ensure that the most suitable loss function plays a dominant role in different stages of training, the fusion strategy scales them by gradients to match the value scale of the specified loss, and then computes a weighted sum based on the predefined weights.

3.2.1. The Gradient significance factor

Since the losses are applied to different layers of the model, to ensure the gradients are computable, the gradient significance factor (GSF_{field}) of each loss is defined as the sum of the L2 norm of the gradients with respect to each layer of the Zipformer ConvEmbed module. The calculation formula is as follows:

$$GSF_{field} = \sum_{i=1}^N \sqrt{\sum_j \left(\frac{\partial L_{field}}{\partial \theta_{ij}} \right)^2} \quad (14)$$

where, L_{field} represents the target loss of different types, $field = \{fea, logits, hard\}$, θ_{ij} denotes the j -th parameter of the i -th layer of the module.

3.2.2. Staged selection of reference gradients significance factor

The distillation training is divided into three stages: initialization, distillation, and fine-tuning. In three stages, feature-level loss, distillation predictive distribution loss, and hard-label loss serve as the dominant loss respectively. This structure ensures that during the initialization stage, the intermediate layers of the model align with the teacher model's features; In the distillation stage, the student model learns from the teacher model through predictive distribution; In the fine-tuning stage, the text label is used as the dominant loss for optimizing.

To better control the update step size of the model parameters, $GSRF_{stage}$, corresponding to the different dominant losses, is used as the reference gradient during the various stages of training:

$$GSRF_{stage} = \begin{cases} GSF_{fea} & , if \ stage = 0 \\ GSF_{logits} & , if \ stage = 1 \\ GSF_{hard} & , if \ stage = 2 \end{cases} \quad (15)$$

where, stage 0 indicates the initialization stage of training, 1 is the distillation stage, and 2 is the fine-tuning stage.

3.2.3. Loss scaling factor

In the process of integrating multiple losses, the scaling coefficients for each loss are computed as follow:

$$scale_{field} = \max\left(\min\left(\frac{Ratio_{field}^{stage} \times GSRF_{stage}}{GSF_{field}}, 1\right), 0\right) \quad (16)$$

where, $Ratio_{field}^{stage}$ ($field = \{fea, logits, hard\}$, $stage = \{0,1,2\}$) is hyperparameter related to stage and loss type, and the value domain is $[0, 1]$.

In Eq. (16), the loss is regularized using GSF_{field} , scaled to the magnitude of the dominant loss by $GSRF_{stage}$, and finally adjusted based on empirical weighting through $Ratio_{field}^{stage}$.

The final overall loss is as follows:

$$Loss = scale_{hard} \times Loss_{hard} + scale_{logits} \times Loss_{logits} + scale_{fea} \times Loss_{fea} \quad (17)$$

By employing the approaches, it is sure that each loss can affect the parameter updates in accordance with the predefined proportions in different stage of training.

4. Experiments

4.1. Experimental setup

Dataset used in the distillation training is a 178-h Chinese Mandarin dataset, Aishell-1. The dataset used in training of the pre-trained large model is a 13,906h open source Chinese Mandarin dataset [4, 14-22].

Following models were tested in the experiment:

(1) Large Model and Small Model

Large Model and Small Model are two Zipformer models trained using Aishell-1 with different model depth and the number of nodes. Their parameter scales are 91M and 21M respectively for the large and the small model.

(2) Distilled-Large

Distilled-Large is a 21M student model distilled from the Large Model(91M) using Aishell-1 while freezing the teacher model parameters during training.

(3) Pretrained-Large

The pretrained large model is an open-sourced model with 150M parameters [23].

(4) Distilled-Pretrained-Large

A 21M student model distilled from the Pretrained-Large using Aishell-1 while freezing the parameters of teacher model during training.

Greedy search strategy is used for streaming decoding of the RNN-T model. The streaming window size (chunk-size) is set to 64, with a left-context-frame size of 256 to focus on the preceding frames.

The performance is evaluated using CER to measure the difference between the output character sequence and the target text sequence.

4.2. Experimental results

4.2.1. Comparison experiment

Table 1 provides experimental results of models on Aishell-1 dataset in CER. Comparing Distilled Large model and Small model, CER is reduced from 8.17% to 7.57%. However, since

CER of the teacher model, Large model, is 6.44%, the CER of the student model is inherently limited by the teacher model's capabilities, which can not reveal the full potential of the distillation method. Therefore, Pretrained-Large model with better CER (1.38%), was used as teacher model for distillation. The result model, Distilled-Pretrained-Large, has a CER of 5.68%, which is relative reduction of 30.48% compared to the directly trained small model. It proved the distillation method can effectively enhance the generalization ability of the small model by leveraging the large model's capabilities despite limited training resources.

Table 1: Results of models on Aishell-1 dataset in CER

Model	Parameters	Test-CER
Large	91M	6.44%
Small	21M	8.17%
Distilled-Large	21M	7.57%
Pretrained-Large	150M	1.38%
Distilled-Pretrained-Large	21M	5.68%

4.2.2. Ablation experiment

Table 2 summarizes the results of the ablation experiments utilizing various loss functions and strategies. The "Id" column corresponds to the experiment identifier. A ratio of 1:0.2 between the RNN-T and CTC losses is used for $Loss_{hard}$. The "Dynamic-Scale" indicates the application of the multi-loss fusion method. "Share-DJ" denotes the student model shares teacher model's decoder and joiner. In the table, "yes" denotes the corresponding strategy was applied, and "no" for not. The evaluation results are reported using the CER on test set of Aishell-1.

Table 2: Results of ablation experiments in CER

Id	$Loss_{hard}$	$Loss_{logits}$	$Loss_{fea}$	Dynamic-Scale	Share-DJ	Test-CER
1	yes	no	no	no	no	8.17%
2	no	yes	no	no	no	6.28%
3	no	yes	no	no	yes	6.08%
4	yes	yes	no	yes	yes	5.93%
5	yes	yes	yes	no	yes	8.16%
6	yes	yes	yes	yes	yes	5.68%

Using model of Experiment 1 as baseline, Fig. 3 visualizes the results of Table 2 from the perspective of relative CER reduction, making it easier to observe the performance differences among models with various strategy combinations.

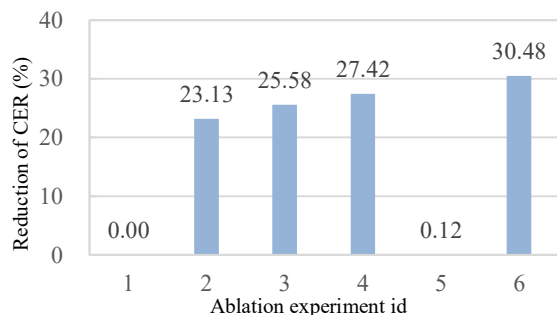


Figure 3: Relative CER reduction of models in ablation experiment using the model of Experiment 1 as baseline.

As can be seen from Table 2 and Fig.3:

(1) The comparison between Experiment 2 and Experiment 1 demonstrates that the sharing pruning boundary in distillation can significantly improve the generalization performance of the student model.

(2) In Experiment 2, the student model initializes its decoder and joiner modules randomly. Comparing with Experiment 2, Experiment 3 reveals that sharing the large model's decoder and joiner can further improve performance of the student model.

(3) Experiment 4 employs a two-stage loss fusion with a step ratio of 0.9:0.1. In the first stage, the weight ratio between the distillation prediction distribution loss and the hard-label loss is 0.2:1, and in the second stage, the ratio is 1:0.5. Experiment 6 extends the fusion to three stages with a step ratio of 0.1:0.8:0.1. The weight ratios for distillation feature loss, distillation prediction distribution loss, and hard-label loss are 1:0.5:2 in the first stage, 0.2:1:0.2 in the second, and 0.1:0.5:1 in the third. Comparing Experiment 4 with Experiment 6, it can be seen that multi-frame-rate feature loss can bring a substantial improvement in model performance.

(4) The comparison between Experiment 5 and Experiment 6 underscores the effectiveness of stage-wise dynamic loss fusion.

Finally, result of Experiment 6 is the best, which proves all strategies in the proposed method are effective and necessary.

4.2.3. Experiment of generalization ability

Dataset THCH30 and TestNet belong to distinct domains and differ from the Aishell dataset both in acquisition method and content. These two datasets are used to evaluate the generalization ability of speech recognition models on out-of-domain dataset.

The experimental results showed in Table 3 demonstrate that, the model distilled by the scheme proposed in this study exhibit significant improvements in generalization across two datasets. Compared to the small model trained directly, the CER on the THCH30-test is 22.92%, reflecting a CER reduction of 23.47%, while the CER on the TestNet is 47.29%, showing a reduction of 9.42%.

Table 3: Results on out-of-domain dataset in CER

Model	THCH30-test	Testnet
Small	29.95%	52.21%
Distilled-Pretrained-Large	22.92%	47.29%

5. Conclusions

This paper proposed a distillation method for pruned RNN-T model for streaming speech recognition, which can effectively distill the knowledge of large model into small model. Compared with the directly trained small models, the small models distilled from large model have better performance in the target domain and stronger generalization ability in other domains, proving the effectiveness of the distillation method in this paper and that distillation from large models of RNN-T architecture is still important.

6. References

- [1] F. Kuang, L. Guo, W. Kang, L. Lin, M. Luo, Z. Yao, and D. Povey, "Pruned mn-t for fast, memory-efficient asr training," in *Proc. Interspeech 2022*, 2022, pp. 2068–2072.
- [2] A. Graves, S. Fernandez, F. Gomez et al., "Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks," in *Proceedings of the 23rd International Conference on Machine Learning*, 2006, pp. 369–376.
- [3] A. Graves, "Sequence transduction with recurrent neural networks," *Computer Science*, vol. 58, no. 3, pp. 235–242, 2012.
- [4] H. Bu, J. Du, X. Na et al., "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, 2017, pp. 1–5.
- [5] Geoffrey Hinton, Oriol Vinyals, and Jeffrey Dean, "Distilling the knowledge in a neural network," in *NIPS Deep Learning and Representation Learning Workshop*, 2015.
- [6] V. Sanh et al., "Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter," *arXiv preprint arXiv:1910.01108*, 2019.
- [7] A. Mohammadshahi, V. Nikoulina, A. Berard et al., "Small-100: Introducing shallow multilingual machine translation model for low-resource languages," in *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, 2022.
- [8] Z. Shen, W. Guo, and B. Gu, "Language-universal adapter learning with knowledge distillation for end-to-end multilingual speech recognition," *arXiv preprint arXiv:2303.01249*, 2023.
- [9] T. P. Ferraz, M. Z. Boito, C. Brun et al., "Multilingual distilwhisper: Efficient distillation of multi-task speech models via language-specific experts," *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2024, pp. 10716–10720.
- [10] J. K. Chorowski, D. Bahdanau, D. Serdyuk et al., "Attention-based models for speech recognition," in *Advances in Neural Information Processing Systems*, vol. 28, 2015.
- [11] Z. Yao, L. Guo, X. Yang, W. Kang, F. Kuang, Y. Yang, Z. Jin, L. Lin, and D. Povey, "Zipformer: A faster and better encoder for automatic speech recognition," in *The Twelfth International Conference on Learning Representations*, 2024.
- [12] A. Gulati, J. Qin, C.-C. Chiu et al., "Conformer: Convolution-augmented transformer for speech recognition," in *Proc. Interspeech*, 2020, pp. 5036–5040.
- [13] M. Ghodsi, X. Liu, J. Apfel et al., "Rnntransducer with stateless prediction network," *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 7049–7053.
- [14] Z. Z. Dong and X. Wang, "Thchs-30: A free chinese speech corpus," <http://arxiv.org/abs/1512.01882>, 2015, accessed: 2025-02-10.
- [15] J. Du, X. Na, X. Liu, and H. Bu, "Aishell-2: Transforming mandarin asr research into industrial scale," *CoRR*, 2018.
- [16] Y. Fu, L. Cheng, S. Lv et al., "Aishell-4: An open-source dataset for speech enhancement, separation, recognition, and speaker diarization in conference scenario," in *Proc. INTERSPEECH*. ISCA, 2021, pp. 3665–3669.
- [17] Surfingtech, "St-cmds-20170001 1 free ST Chinese Mandarin corpus," 2017.
- [18] L. Primewords Information Technology Co., "Prime words Chinese corpus set 1," <https://www.primewords.cn>, 2018.
- [19] L. Magic Data Technology Co., "Magicdata Mandarin Chinese read speech corpus," <http://www.imagicdatatech.com/index.php/home/dataopensource/datainfo/id/101>, 2019.
- [20] F. Yu, S. Zhang et al., "M2met: The ICASSP2022 multi-channel multi-party meeting transcription challenge," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2022.
- [21] B. Zhang, H. Lv, P. Guo et al., "Wenets: A 10000+ hours multi-domain Mandarin corpus for speech recognition," *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2022, pp. 6182–6186.
- [22] Z. Tang, D. Wang, Y. Xu et al., "Kespeech: An open-source speech dataset of Mandarin and its eight sub-dialects," in *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*, 2021.
- [23] Yuekai, "Icefall-asr-multi-zh-hans-zipformer-large: A large-scale Zipformer-based multi-dialect Chinese ASR model," <https://huggingface.co/yuekai/icefall-asr-multi-zh-hans-zipformer-large>, 2024, accessed: 2025-02-07.