AN EFFICIENT AND ROBUST MULTI-STREAM FRAMEWORK FOR END-TO-END SPEECH RECOGNITION

by
Ruizhi Li

A dissertation submitted to Johns Hopkins University in conformity
with the requirements for the degree of Doctor of Philosophy

Baltimore, Maryland
October 2020

Abstract

In voice-enabled domestic or meeting environments, distributed microphone arrays aim to process distant-speech interaction into text with high accuracy. However, with dynamic corruption of noises and reverberations or human movement present, there is no guarantee that any microphone array (stream) is constantly informative. In these cases, an appropriate strategy to dynamically fuse streams is necessary.

The multi-stream paradigm in Automatic Speech Recognition (ASR) considers scenarios where parallel streams carry diverse or complementary task-related knowledge. Such streams could be defined as microphone arrays, frequency bands, various modalities or etc. Hence, a robust stream fusion is crucial to emphasize on more informative streams than corrupted ones, especially under unseen conditions. This thesis focuses on improving the performance and robustness of speech recognition in multi-stream scenarios.

With increasing use of Deep Neural Networks (DNNs) in ASR, End-to-End (E2E) approaches, which directly transcribe human speech into text, have received greater attention. In this thesis, a multi-stream framework is presented based on the joint Connectionist Temporal Classification/ATTention (CTC/ATT) E2E model, where parallel streams are represented by separate encoders. On top of regular attention networks, a secondary stream-fusion network is to steer the decoder toward the most informative streams.

The MEM-Array model aims at improving the far-field ASR robustness using microphone arrays which are activated by separate encoders. With an increasing number of streams (encoders) requiring substantial memory and massive amounts of parallel data, a practical two-stage training strategy is designated to address these issues. Furthermore, a two-stage augmentation scheme is present to improve robustness of the multi-stream model. In MEM-Res, two heterogeneous encoders with different architectures, temporal resolutions and separate CTC networks work in parallel to extract complementary information from the same acoustics. Compared with the best single-stream performance, both models have achieved substantial improvement, outperforming alternative fusion strategies.

While the proposed framework optimizes information in multi-stream scenarios, this thesis also studies the Performance Monitoring (PM) measures to predict if recognition results of an E2E model are reliable without growth-truth knowledge. Four PM techniques are investigated, suggesting that PM measures on attention distributions and decoder posteriors are well-correlated with true performances.

Thesis Committee

Primary Readers

Hynek Hermansky (Advisor)

Shinji Watanabe

Alternative Readers

Najim Dehak

Gregory Sell

Dedication

This thesis is dedicated to my parents, Jinfeng Xu and Xinghua Li, and my grandparents, Dezhen Zhong, Yutang Xu, Guixiang Luo, and Shengde Li, for their eternal love, trust, and support.

Acknowledgements

First and foremost, I am immensely grateful to my advisor, Hynek Hermansky, for taking me as your student while I was in a difficult situation. Thank you for your insight, patience, and constant support which made all this work possible.

I would like to thank Shinji Watanabe and Takaaki Hori for their thoughtful advice and discussion. Working with you in the 2018 CLSP workshop truly helped me find the right direction for my Ph.D. thesis. I am grateful to Alan McCree and Gregory Sell for involving me in the JHU-ULCA Speaker Identification project. Thank you Greg, for always being supportive and encouraging as both a mentor and a friend. I want to thank Najim for the opportunity to research Language Identification at the early stage of my Ph.D. and for graciously serving on my committee. I thank Sanjeev Khudanpur for always being cheerful and supportive to all the graduate students. I want to thank Aparna Khare, Yuzong Liu, I-Fan Chen, and Harish Mallidi for being my mentors while I was interning at Amazon.

My Ph.D. in Speech Processing started with work by my labmate Harish and the BUT folks (Olda, Ondrej, Pavel, Karel, Lukas, Katerina, Honza, and Lucas) in Brno, Czech Republic. I am thankful to them for helping me in my initial year. To Harish, I feel very lucky to have a senior labmate and friend who gave me so many of his honest opinions and helpful suggestions to get me through my Ph.D. successfully.

As for the CLSP folks, I am super grateful that we could meet here at JHU and spend the best years of our lives together. To all my friends (Jinyi, Xiaofei, Samik, Bernd, Johnathan, Aswin, Xuankai, Raghu, Phani, and Qinwan) from room 323 at Hackerman, thank you for all the discussion, laughter, gossip, and parties we had together. To David, Ke, Xiaohui, Vimal, Matt, JJ, Nanxin, Yiming, Ruizhe, Desh, Paola, Jesus, and all my other JHU friends and colleagues, thank you for being such great company and so willing to help me in times of need. I want to thank Ruth for taking care of all the administrative work and for being so considerate and patient.

To Di, thank you for motivating me to apply for JHU in the first place. To Jinyi, I am grateful that we are always there for each other. I am thankful to have Laure as my best music buddy and as a friend that never turns you away. I would like to thank Chris Lowy for our endless phone calls discussing life, work, and relationships. I want to thank Luke, Andrew, Tim, and Wyatt for all the good times we spent together. To Luke, thank you for always being there for my ups and downs.

I am grateful to have Ti in my life. You have left us too soon. I miss you.

Special thanks to my boyfriend, Chris. You never complained and always stayed optimistic. Your positivity really helped me get through the darkness before the dawn. I am grateful to have you in my life.

Lastly, I want to thank myself for having faith and not giving up.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Motivation

The multi-stream paradigm in speech processing considers scenarios where parallel streams carry diverse or complementary task-related knowledge. Recent advancements in speech technology enable a diverse set of applications. These examples include distributed voiced-enabled speakers like Google Home or Amazon Echo in a home environment, a meeting room equipped with close-talk and far-field microphones, and hearing-aid devices with multi-channel processing. In these cases, an appropriate strategy is crucial to reliably fuse streams or select the most informative source to handle scenarios with interference (noise or reverberation) in acoustic environments. This thesis focuses on improving the performance and robustness of Automatic Speech Recognition (ASR) systems in the multi-stream setting.

Recently, with the increasing use of Deep Neural Networks (DNNs) in ASR, End-to-End (E2E) speech recognition approaches, which directly transcribe human speech into text, have received greater attention. Without relying on the complicated legacy architectures from conventional ASR, the E2E model combines several disjoint components (acoustic model, pronunciation model, language model) into one single DNN for joint training. Previously, the multi-stream scheme has been widely studied using traditional ASR approaches. However, it has not been investigated for E2E models. In conventional ASR, stream fusion modules are often optimized separately with different objectives. The flexibility of E2E systems provides potentials to directly incorporate stream fusion in the system for joint optimization. Moreover, unlike multi-channel ASR, distant microphone arrays are often operated without time synchronization, which makes the beamforming technique infeasible to merge far-field signals. The multi-stream model presented in this thesis could combine parallel knowledge in the model without this presumption.

1.2 Focus of the Thesis

In this thesis, a novel multi-stream framework is presented to address ASR scenarios with parallel information sources. The proposed model is built on the joint Connectionist Temporal Classification/ATTention (CTC/ATT) end-to-end model, which takes advantage of both Connectionist Temporal Classification (CTC) and attention-based model for multi-task training and joint decoding.

Two representative frameworks are proposed and discussed, which are Multi-Encoder Multi-Array (MEM-Array) framework and Multi-Encoder Multi-Resolution (MEM-Res) framework, respectively. The MEM-Array framework aims at improving the far-field ASR robustness using multiple microphone arrays. Several techniques are proposed for effective training and handling a variety of unseen single-stream conditions and inter-stream dynamics. In the MEM-Res framework, two heterogeneous encoders with different architectures, temporal resolutions and separate CTC networks work in parallel to extract complementary information from the same acoustics. Moreover, four Performance Monitoring (PM) measures are presented to determine the quality of an E2E system's output based only on behavior of the system without any knowledge of the underlying truth.

1.3 Thesis Outline

This thesis is organized as follows:

Chapter 2 provides an overview of the multi-stream speech recognition. We review the basics of conventional ASR systems and three E2E approaches. A brief review of previously proposed techniques for multi-stream ASR is then presented.

In Chapter 3, we propose a general form of the multi-stream end-to-end model with detailed description for each module. Two representative networks, MEM-Array and MEM-Res, are briefly introduced.

Chapter 4 focuses on the MEM-Array model in the scenario of multiple microphone arrays. First, we explain all the components in the MEM-Array model. A practical two-stage training strategy is then introduced to efficiently scale the training procedure while improving the performance. Subsequently, two-stage augmentation scheme and adaptive CTC fusion are proposed to handle mismatched scenarios.

In Chapter 5, we describe the MEM-Res model as another realization of the multi-stream model. In the MEM-Res model, two parallel encoders with heterogeneous structures are mutually complementary in characterizing the speech signal.

In Chapter 6, four PM measures are presented to evaluate reliability of the system output. An analysis of these techniques on typical single-stream E2E models and proposed multi-stream models are provided.

Chapter 7 summarizes the contributions of this thesis and highlights several directions for future research.

1.4 Related Publications

Large portions of Chapter 2, 3, 4, 5 and 6 have appeared in the following papers:

1. Xiaofei Wang‡, Ruizhi Li‡, Sri Harish Mallidi, Takaaki Hori, Shinji Watanabe, and Hynek Hermansky. "Stream attention-based multi-array end-to-end speech recognition." In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 7105-7109. IEEE, 2019.

2. Ruizhi Li, Xiaofei Wang, Sri Harish Mallidi, Shinji Watanabe, Takaaki Hori, and Hynek Hermansky. "Multi-stream end-to-end speech recognition." *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 28 (2019): 646-655.

3. Ruizhi Li, Gregory Sell, and Hynek Hermansky, "Performance monitoring for end-to-end speech recognition." in *Interspeech*, pp. 2245-2249. 2019.

4. Ruizhi Li, Gregory Sell, Xiaofei Wang, Shinji Watanabe, and Hynek Hermansky. "A practical two-stage training strategy for multi-stream end-to-end speech recognition." In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 7014-7018. IEEE, 2020.

5. Ruizhi Li, Gregory Sell, Hynek Hermansky, "Two-stage augmentation and adaptive ctc fusion for improved robustness of multi-stream end-to-end asr.", in *Spoken Language Technology (SLT) 2021.* (Submitted)

Chapter 2

Overview of Multi-Stream Speech Recognition

2.1 Automatic Speech Recognition

An ASR system creates a monotonic mapping from T-length acoustic features $X = \{\mathbf{x}_t \in \mathbb{R}^D | t = 1, 2, ..., T\}$ in D dimensional space to a N-length word sequence $W = \{w_n \in \mathcal{V} | n = 1, 2, ..., N\}$ where $\mathcal{V}$ is a set of distinct words in the vocabulary. The mathematical ASR formulation is described based on Bayesian decision theory as follows [1, 2]:

$$\hat{W} = \arg\max_{W \in \mathcal{V}^*} p(W|X), \tag{2.1}$$

where the most probable word sequence, $\hat{W}$, is estimated among all possible word sequences, $\mathcal{V}^*$. The main focus of ASR is then to obtain the posterior probability $p(W|X)$.

2.1.1 Conventional Hybrid Approach

The typical paradigm for an ASR system is the so-called hybrid approach [3]. It introduces the state sequence, $S = \{s_t \in \mathcal{J} | t = 1, 2, ..., T\}$, from a Hidden Markov Model (HMM) [4], where $\mathcal{J}$ is a set of predefined HMM states. As stated in the following, $p(W|X)$ is then factorized into three components: acoustic model $p(X|S)$, lexicon $p(S|W)$, and language model $p(W)$:

$$\hat{W} = \arg\max_{W \in \mathcal{V}^*} p(W|X) \tag{2.2}$$

$$= \arg\max_{W \in \mathcal{V}^*} \sum_{S} p(X|S,W) p(S|W) p(W) \tag{2.3}$$

$$\approx \arg\max_{W \in \mathcal{V}^*} \sum_{S} p(X|S) p(S|W) p(W), \tag{2.4}$$

where conditional independence assumption is employed, i.e., $p(X|S,W) \approx p(X|S)$.

Using chain rule and conditional independence assumption, the acoustic model $p(X|S)$ is estimated as follows:

$$p(X|S) = \prod_{t=1}^{T} p(\mathbf{x}_t|\mathbf{x}_1, ..., \mathbf{x}_{t-1}, S) \tag{2.5}$$

$$\approx \prod_{t=1}^{T} p(\mathbf{x}_t|s_t) \propto \prod_{t=1}^{T} \frac{p(s_t|\mathbf{x}_t)}{p(s_t)}, \tag{2.6}$$

where advanced DNNs are applied to compute the frame-wise posterior distribution $\frac{p(s_t|\mathbf{x}_t)}{p(s_t)}$. It is often required to obtain the frame-wise state alignments, $\{s_t\}$, provided by an HMM/GMM (Gaussian Mixture Model) as DNN training targets. Similarly, lexicon model is formulated using chain rule and first-order Markov assumption as follows:

$$p(S|W) = \prod_{t=1}^{T} p(s_t|s_1, ..., s_{t-1}, W) \tag{2.7}$$

$$\approx \prod_{t=1}^{T} p(s_t|s_{t-1}, W). \tag{2.8}$$

This is a deterministic mapping created with a pronunciation dictionary. The language model $p(W)$ is factorized in the following:

$$p(W) = \prod_{n=1}^{N} p(w_n|w_1, ..., w_{n-1}) \tag{2.9}$$

$$\approx \prod_{n=1}^{N} p(w_n|w_{n-m+1}, ..., w_{n-1}), \tag{2.10}$$

where a probabilistic chain rule and conditional independence assumption are also applied.

Despite the impressive accuracy of the hybrid system, it requires hand-crafted pronunciation dictionary based on linguistic assumptions, extra training steps to derive context-dependent phonetic models, and text pre-processing such as tokenization for languages without explicit word boundaries. Consequently, it is quite difficult for non-experts to develop ASR systems for new applications, especially for new languages.

2.1.2 End-to-End Approach

End-to-end speech recognition approaches are designed to directly output word or character sequences from the audio signal. Three dominant E2E architectures for ASR are Connectionist Temporal Classification [5–7], attention-based encoder decoder [8, 9] models and recurrent neural network transducers [10, 11]. A joint CTC/ATT framework was proposed in [12–14] to benefit from both CTC and attention-based models.

2.1.2.1 Connectionist Temporal Classification (CTC)

Following Bayes decision theory, CTC formulates a sequence-to-sequence model in a speech recognition task. It enforces a monotonic mapping from a T-length speech feature sequence, $X = \{\mathbf{x}_t \in \mathbb{R}^D | t = 1, 2, ..., T\}$, to an L-length letter sequence [1], $C = \{c_l \in \mathcal{U} | l = 1, 2, ..., L\}$. Here $\mathbf{x}_t$ is a D-dimensional acoustic vector at frame t, and c_l is at position l a letter from $\mathcal{U}$, a set of distinct letters.

Unlike the traditional hybrid approach where forced alignment is required, when the CTC algorithm is employed, it derives alignments between the input and the output automatically without any intermediate process. The CTC network intro-

[1]It could also be a Byte-Pair Encoding (BPE) sequence.

duces a many-to-one function from frame-wise latent variable sequences, $Z = \{z_t \in \mathcal{U} \bigcup \text{blank} | t = 1, 2, ..., T\}$, to letter predictions with shorter lengths. Note that the additional "blank" symbol is used to handle the merging of repeating letters. This is a many-to-one function because many CTC paths can respond to the same label sequence by combining repeating characters and removing blank symbols. With several conditional independence assumptions, the posterior distribution, $p(C|X)$, is represented as follows:

$$p(C|X) \approx \sum_{Z} \prod_{t} p(z_t|X) \triangleq p_{\text{ctc}}(C|X), \tag{2.11}$$

where $p(z_t|X)$ is a frame-wise posterior distribution. We define the CTC objective function as $p_{\text{ctc}}(C|X)$.

Since the frame-wise posterior probability is conditioned on all input frames, models trained using the CTC objective typically apply encoders with Bidirectional Long Short-Term Memory (BLSTM) [15, 16] layers to estimate $p(z_t|X)$:

$$p(z_t|X) = \text{Softmax}(\text{LinB}(\mathbf{h}_t)), \tag{2.12}$$

$$\mathbf{h}_t = \text{Encoder}(X). \tag{2.13}$$

BLSTM-based encoder processes the full input sequence and outputs a hidden vector $\mathbf{h}_t$ at frame t. Encoder$(\cdot)$ represents the encoder component. LinB$(\cdot)$ is a linear layer with a bias term converting $\mathbf{h}_t$ to a $(|\mathcal{U}| + 1)$ dimensional vector, followed by a Softmax activation function.

Similar to the forward algorithm for HMMs, a dynamic programming technique is exploited to efficiently compute the summation over all possible paths, Z. While CTC does not fully carry out the strength of end-to-end ASR due to conditional assumptions, it preserves the benefits that it enforces the monotonic behavior of speech-label alignments, avoids the HMM/GMM construction step and preparation of pronunciation dictionaries.

2.1.2.2 Attention-based Model

As one of the most commonly used sequence modeling techniques, the attention-based framework selectively encodes an audio sequence of variable length into a fixed dimension vector representation, which is then consumed by the decoder to produce a distribution over the outputs. This approach models each output letter c_l with a conditional probability distribution on previous output letters $c_1, ..., c_{l-1}$ and the input features X. We can directly estimate the posterior distribution $p(C|X)$ using the chain rule:

$$p(C|X) = \prod_{l=1}^{L} p(c_l|c_1, ..., c_{l-1}, X) \triangleq p_{\text{att}}(C|X), \tag{2.14}$$

where $p_{\text{att}}(C|X)$ is defined as the attention-based objective function. During training, objective function in Eq. 2.14 is conditioned on previous ground truth labels, $c_1^{\dagger}, ..., c_{l-1}^{\dagger}$. Typically, a BLSTM-based encoder transforms the speech vectors $X = \{\mathbf{x}_1, ..., \mathbf{x}_T\}$ into a set of frame-wise hidden vectors $H = \{\mathbf{h}_1, ..., \mathbf{h}_{\lfloor T/s \rfloor}\}$, where the encoder often subsamples the input by a factor s to reduce the computational complexity [8, 9]. For each prediction c_l, the letter-wise context vector $\mathbf{r}_l$ is then formed as a weighted summation of frame-wise hidden vectors H using attention network [8]:

$$\mathbf{r}_l = \sum\nolimits_{t=1}^{\lfloor T/s \rfloor} a_{lt} \mathbf{h}_t, \tag{2.15}$$

$$a_{lt} = \text{Attention}(\{a_{l-1}^{(i)}\}_{t=1}^{\lfloor T/s \rfloor}, \mathbf{q}_{l-1}, \mathbf{h}_t), \tag{2.16}$$

where a_{lt} is the attention weight, a soft-alignment of $\mathbf{h}_t$ for output c_l, and $\mathbf{q}_{l-1}$ is the previous decoder state. Among a variety of attention mechanisms applicable for Eq.

2.16, we introduce two types of attention network used in this thesis: Content-based attention and Location-based attention [8].

Content-based attention is described as follows:

$$e_{lt} = \mathbf{g}^\top \tanh(\mathrm{Lin}(\mathbf{q}_{l-1}) + \mathrm{LinB}(\mathbf{h}_t)), \tag{2.17}$$

$$a_{lt} = \mathrm{Softmax}(\{e_{lt}\}_{t=1}^{\lfloor T/s \rfloor}). \tag{2.18}$$

$\mathbf{g}$ is a learnable vector parameter. $\tanh(\cdot)$ is a hyperbolic tangent function. $\{e_{lt}\}_{t=1}^{\lfloor T/s \rfloor}$ is a $\lfloor T/s \rfloor$-dimensional vector. $\mathrm{LinB}(\cdot)$ and $\mathrm{Lin}(\cdot)$ represent the linear transformation with or without bias term, respectively.

Location-based attention is an extension of content-based attention with additional convolutional features computed by previous attention distributions, $\mathbf{a}_{l-1} = [a_{l-1,1}, ..., a_{l-1,\lfloor T/s \rfloor}]^\top$:

$$\{\mathbf{f}_{l-1,t}\}_{t=1}^{\lfloor T/s \rfloor} = \mathbf{K} * \mathbf{a}_{l-1}, \tag{2.19}$$

$$e_{lt} = \mathbf{g}^\top \tanh(\mathrm{Lin}(\mathbf{q}_{l-1}) + \mathrm{LinB}(\mathbf{h}_t) + \mathrm{Lin}(\mathbf{f}_{l-1,t})), \tag{2.20}$$

$$a_{lt} = \mathrm{Softmax}(\{e_{lt}\}_{t=1}^{\lfloor T/s \rfloor}). \tag{2.21}$$

In Eq. 2.19, one dimensional convolution is performed along the time axis of attention vector, $\mathbf{a}_{l-1}$, with the convolution parameter, $\mathbf{K}$.

The decoder is typically a Long Short-Term Memory (LSTM) [15] network, where $\mathbf{q}_{l-1}$ is the previous hidden state and input is the concatenated vector of letter-wise context vector $\mathbf{r}_l$ and vector representation of the previous prediction c_{l-1}:

$$p(c_l|c_1, ..., c_{l-1}, X) = \mathrm{Decoder}(\mathbf{r}_l, \mathbf{q}_{l-1}, c_{l-1}). \tag{2.22}$$

The attention-based model implicitly associates acoustic, lexicon, and language models as encoder, attention, and decoder and combines these modules into one single network so that all the parameters can be jointly optimized by back-propagation to maximize $p(C|X)$. In comparison to CTC, not requiring conditional independence assumptions is one of the advantages of using the attention-based model. However, the attention is too flexible to satisfy monotonic alignment constraint in speech recognition tasks. There are previous publications to enhance the monotonic behavior in various ways [17–21]. These studies are similar in a way that they operate local attentions on the windowed encoder outputs to enforce monotonicity. As stated in [9], there is a strong assumption that the training function is a combination of letter-wise objectives conditioned on previous true labels instead of a desired sequence-level objective.

2.1.2.3 Joint CTC/ATT Model

The joint CTC/ATT model, illustrated in Fig. 2.1, takes advantage of both CTC and attention-based model through a Multi-Task Learning (MTL) mechanism and joint decoding. It is designed to directly map T-length acoustic features $X = \{\mathbf{x}_t \in \mathbb{R}^D | t = 1, 2, ..., T\}$ in D dimensional space to an L-length letter sequence $C = \{c_l \in \mathcal{U} | l = 1, 2, ..., L\}$ where $\mathcal{U}$ is a set of distinct letters. The attention-based structure solves the ASR problem as a sequence mapping by using an encoder-decoder architecture. Joint training with CTC is added to help enforce temporally monotonic behavior in the attention alignments.

The encoder transforms the acoustic sequence X into a higher-level feature representation $H = \{\mathbf{h}_1, ..., \mathbf{h}_{\lfloor T/s \rfloor}\}$, which is shared for the use of CTC and attention-based models.

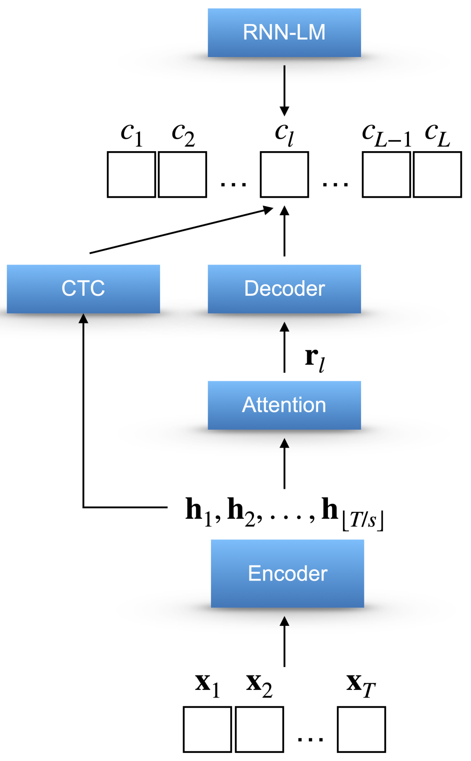


Figure 2.1. Joint CTC/ATT end-to-end architecture

The objective function to be maximized is as follows:

$$\mathcal{L}_{\text{MTL}} = \lambda \log p_{\text{ctc}}(C|X) + (1-\lambda) \log p^{\dagger}_{\text{att}}(C|X), \tag{2.23}$$

where the joint objective is a logarithmic linear combination of the CTC and attention training objectives, i.e., $p_{\text{ctc}}(C|X)$ and $p^{\dagger}_{\text{att}}(C|X)$, respectively. $\lambda \in [0,1]$ is a trade-off hyperparameter. $p^{\dagger}_{\text{att}}(C|X)$ is an approximated objective from Eq. 2.14 defined as follows:

$$p^{\dagger}_{\text{att}}(C|X) = \prod_{l=1}^{L} p(c_l|c^{\dagger}_1, ..., c^{\dagger}_{l-1}, X) \approx p_{\text{att}}(C|X). \tag{2.24}$$

As shown in Eq. 2.24, $\{c^{\dagger}_1, ..., c^{\dagger}_{l-1}\}$ denotes the ground truth of previous steps.

During inference, a label-synchronous beam search is employed to predict the most probable label sequence $\hat{C}$:

$$\hat{C} = \arg\max_{C \in \mathcal{U}^*} \{\lambda \log p_{\text{ctc}}(C|X) + (1-\lambda) \log p_{\text{att}}(C|X) + \gamma \log p_{\text{lm}}(C)\} \tag{2.25}$$

where $\log p_{\text{lm}}(C)$ is evaluated from an external Recurrent Neural Network Language Model (RNN-LM) with a scaling factor γ. For each partial hypothesis h in the beam search, the log probability of hypothesized label sequence, is computed as

$$\alpha(h) = \lambda \alpha_{\text{ctc}}(h) + (1-\lambda)\alpha_{\text{att}}(h) + \gamma \alpha_{\text{lm}}(h), \tag{2.26}$$

where the attention decoder score, $\alpha_{\text{att}}(h)$, can be accumulated recursively from hypothesis scores from one step before. In terms of the CTC score, $\alpha_{\text{ctc}}(h)$, we utilize the CTC prefix probability defined as the cumulative probability of all label sequences that have h as their prefix [22, 23]. In this work, we use the look-ahead word-based language model to give the RNN-LM score [24], $\alpha_{\text{lm}}(h)$. This language model enables us to decode with only a word-based model, which leads competitive accuracy and less computation in the beam search process, rather than using a multi-level Language Model (LM) which uses a character-level LM until the identity of the word is determined.

2.2 Multi-Stream Speech Processing

Multi-stream speech recognition deals with scenarios where parallel information streams are processed by an ASR system to obtain text messages from speeches. The approaches for knowledge fusion can be grouped into two categories: front-end level approaches and model level approaches.

2.2.1 Front-end Level Approaches

Stream combination in the front-end aims to generate single inputs from multiple inputs in signal-level or feature-level. Multi-channel ASR is a typical multi-stream scenario, in which several time-synchronized microphones on a single array are considered to cope with noise and reverberation in far-field environments. Multi-channel speech enhancement techniques that aim to produce an enhanced single-channel input are often employed to improve ASR performance and robustness. Several studies [25, 26] in conventional ASR have shown that pre-processing with multi-channel speech enhancement algorithms, especially beamforming techniques, achieved substantially better performances in the presence of strong background noise.

Beamforming is a popular multi-channel approach that utilizes spatial, temporal and spectral information. Traditional beamforming methods [27, 28], such as delay-and-sum and filter-and-sum, are optimized independent of ASR objective. They also require additional steps to determine steering vectors for ASR task [29]. Later on, a complex GMM-based time-frequency mask estimation was proposed in [30] to steer a beamformer, proven to be beneficial in real scenarios [31, 32]. On the other hand, with recent advancements in deep learning, several neural beamformers, including ones based on filter estimation [33–35] and mask estimation [36–43], have been proposed in hybrid ASR. More recently, multi-channel end-to-end framework was present in

[44–46] by jointly training a unified network including the beamforming component. This E2E approach shows robustness against channel reordering and unseen channel configurations. Aside from beamforming methods, a sensory attention mechanism [47, 48] is used in an end-to-end model to combine channels. The techniques mentioned above all operate under the assumption of time synchronization. This assumption may not be held when asynchronous audio devices are present, such as mobile phones and laptops in addition to microphone arrays in a meeting room.

In the multi-array setting with possible asynchronous signals, session-wise approach was present in [49, 50] by solving sampling frequency mismatch to synchronize multi-channel observations followed by a minimum variance distortionless response (MVDR) beamformer. Moreover, [51] proposed a system that generates speaker-annotated meeting transcripts by incorporating audio stream alignment and blind beamforming. A Signal-to-Noise-Ratio(SNR)-based array selection was demonstrated in [52] in multiple-array track of CHiME-5 challenge [53]. While methods mentioned above are investigated under hybrid ASR framework, it is still unknown towards an end-to-end approach in the multi-array setting.

2.2.2 Model Level Approaches

Previous model level methods are discussed here in which fusion strategies are operated based on behavior of the model. Various multi-stream applications have been studied via model space approaches for information fusion.

Earlier work in [54] on articulatory index suggests that the human auditory system decodes linguistic information independent in frequency bands and the final message is merged from these sub-bands. As one source of inspiration, multi-band processing [55] has been successfully applied to robust speech recognition [56–62], indicating the potential of multi-stream techniques in speech processing. For instance, in [58, 59, 63, 64], the fusion module evaluates stream performance by comparing the similarity of DNN output statistics between training and testing conditions, resulting in a process of stream confidence generation. These statistics could be computed using the autocorrelation matrix of transformed probability estimates [63] or predictions from the GMM [58] or the DNN auto-encoder [59]. In [61, 65], a DNN-based fusion module is proposed to be trained on concatenated DNN features from individual streams with random stream dropout strategy. This fusion technique reduces the complexity of multi-stream architecture and enhances robustness against noisy data.

Considering far-field ASR using multiple microphone arrays, without any knowledge of speaker-array distance or orientation, it is still challenging to speculate which array is most informative or least corrupted. Combination using the classifier's posterior probabilities followed by lattice generation has been investigated in several studies [66–69]. Compared to using the fully decoding results with paths pruning, the combination using the posteriors preserves all the information from the test speech as well as the classifier. For example, our previous study [68] proposed a stream attention framework to improve far-field ASR, where reliable HMM state posterior probabilities are generated by linearly combining the posteriors from each array, under the supervision of ASR performance monitors, i.e., mean time distance [70] and time-delayed DNN auto-encoder [68]. Channel-selection framework in [69] conducts entropy analysis of neural network posterior probabilities. Study in [71] fuses the decoding results by voting for the most confident words, for instance ROVER [72]. System combination can be also applied in multi-array setting at the transcript level, which can be achieved using minimum Bayes risk decoding [73].

Aside from the multi-stream scenarios for noise robustness and multi-array, numerous research directions also incorporate the multi-stream idea. For instance, [74] investigated several performance measures in spatial acoustic scenes to choose the most reliable source for hearing aids. [75] proposed a CTC-based system combination via procedures of subsystem selection, alignment and voting. The multi-modal applications combine visual [76] or symbolic [77] inputs together with audio signal to improve speech recognition.

Chapter 3

Multi-Stream End-to-End Speech Processing

3.1 Introduction

As discussed in the previous chapter, compared to the hybrid approach, end-to-end speech recognition approaches are designed to directly output word or character sequences from the input audio signal. This model subsumes several disjoint components in the hybrid ASR model (acoustic model, pronunciation model, language model) into a single neural network. As a result, all the components of an E2E model can be trained jointly to optimize a single objective.

While CTC efficiently addresses a sequence-to-sequence problem (speech vectors to word sequence mapping) by avoiding the alignment pre-construction step using dynamic programming, it assumes the conditional independence of label sequence given the input. The attention model does not assume conditional independence of a label sequence resulting in a more flexible model. However, attention-based methods encounter difficulty in satisfying the speech-label monotonic property. A joint CTC/ATT framework, as mentioned in the previous chapter, was proposed with the help of a monotonic model, CTC, to alleviate this issue. The joint model has shown to provide the state-of-the-art E2E results in several benchmark datasets [14].

The multi-stream paradigm in speech processing considers scenarios where parallel streams carry diverse or complementary task-related knowledge. In these cases, an appropriate strategy to fuse streams or select the most informative source is necessary. One potential source of inspiration in this setting is from the observations of parallel processing in the human auditory system, and resulting innovations have been successfully applied to conventional ASR frameworks, described in the previous Chapter. While various scenarios were explored within hybrid models, multi-stream approaches have not been fully investigated for end-to-end ASR schemes.

In this chapter, a novel multi-stream architecture is proposed within the joint CTC/ATT end-to-end framework. We present a general formulation to multi-stream framework and an introduction to two practical E2E applications: Multi-Encoder Multi-Array (MEM-Array) and Multi-Encoder Multi-Resolution (MEM-Res). The framework has the following highlights:

1. Multiple Encoders in parallel act as information streams. Two ways of forming the streams have been proposed in this work according to different applications: Parallel speech signals from multiple microphone arrays are fed into separate but identical encoders, which we refer to as Multi-Encoder Multi-Array (MEM-Array) model; Parallel encoders with different architectures and temporal resolutions operate on the same acoustics, which we refer to as Multi-Encoder Multi-Resolution (MEM-Res) model.

2. The Hierarchical Attention Network (HAN) [78–80] is introduced to dynamically combine knowledge from parallel streams. While one way of information fusion is to apply one attention mechanism across the outputs of multiple encoder [80], several studies demonstrated benefits of multiple attention mechanisms [78–83].

In [84, 85], secondary attention modules provide a way to incorporate additional contextual information beneficial to the tasks. Inspired by the advances in hierarchical attention mechanism in document classification task [78], multi-modal video description [79] and machine translation [80], we adopt the HAN component into our multi-stream model. The encoder that carries the most discriminative information for the prediction can dynamically receive a higher weight. On top of the frame-level attention mechanism for every encoder, stream attention is employed to steer toward the stream, which carries more task-related information.

3. Each encoder is associated with a separate CTC network to guide the frame-wise alignment process for each stream to potentially achieve better performance.

The MEM-Array model is one realization of our multi-stream E2E framework. Far-field ASR using multiple microphone arrays has become important strategies in the speech community toward a smart speaker scenario in a meeting room or house environment [53, 86, 87]. Individually, the microphone array is able to bring a substantial performance improvement with algorithms such as beamforming [88] and masking [89]. However, what kind of information can be extracted from each array and how to make multiple arrays work in cooperation are still challenging. Time synchronization among arrays is one of the main challenges that most distributed setups face [90]. Without any prior knowledge of speaker-array distance or video monitoring, it is difficult to estimate which array carries more reliable information or is less corrupted.

According to the reports from the CHiME-5 challenge [53], which targets the problem of multi-array conversational speech recognition in home environments, the common ways of utilizing multiple arrays in the hybrid ASR system are finding the one with highest Signal-to-Noise/Signal-to-Interference Ratio (SNR/SIR) for decoding [52] or fusing the decoding results by voting for the most confident words [71], e.g. ROVER [72]. While most of the end-to-end ASR studies engage in single-channel task or multi-channel task from one microphone array [44, 45, 47, 48], research on multi-array scenario is still unexplored within the E2E framework. Without assuming time synchronization across streams, the MEM-Array model is proposed to solve the aforementioned problem. The output of each microphone array is modeled by a separate encoder. Multiple encoders with the same configuration act as the acoustic models for individual arrays. Note that we integrate beamformed signals instead of using all multi-channel signals for the multi-stream framework, which is computationally efficient. This design can make use of the powerful beamforming algorithm for synchronized signals as well.

In the MEM-Res model, two parallel encoders with heterogeneous structures are mutually complementary in characterizing the speech signal. In end-to-end ASR, the encoder acts as an acoustic model providing higher-level features for decoding. BLSTM has been widely used due to its ability to model temporal sequences and their long-term dependencies as the encoder architecture; Deep Convolutional Neural Network (CNN) is introduced to model spectral local correlations and reduce spectral variations in E2E framework [13, 91, 92]. The encoder combining CNN with recurrent layers, is suggested to address the limitation of LSTM. While temporal subsampling in RNN and max-pooling in CNN aim to reduce the computational complexity and enhance the robustness, it is likely that subsampling technique results in loss of temporal resolution. The MEM-Res model is proposed to combine RNN-based and CNN-RNN-based networks to form a complementary multi-stream encoder setup.

3.2 Proposed Multi-Stream Framework

The proposed multi-stream architecture of N streams is shown in Fig. 3.1. Encoders are presented in parallel to capture information in various ways, followed by an attention fusion mechanism together with per-encoder CTC. An external RNN-LM is involved only during the inference step. We will describe the details of each component in the following sections.

3.2.1 Parallel Encoders as Multi-Stream

We denote a $T^{(i)}$-length sequence of D-dimensional speech vectors as $X^{(i)} = \{\mathbf{x}_t^{(i)} \in \mathbb{R}^D | t = 1, 2, ..., T^{(i)}\}$, where superscript $i \in \{1, ..., N\}$ is the index for Encoder$^{(i)}$ corresponding to stream i. The multi-stream E2E model directly maps N information sources, $X = \{X^{(1)}, X^{(2)}, ..., X^{(N)}\}$, into an L-length label sequence, $C = \{c_l \in \mathcal{U} | l = 1, 2, ..., L\}$. Here $\mathcal{U}$ is a set of distinct labels. Similar to acoustic modeling in conventional ASR, the encoder maps audio features into higher-level feature representations for the use of CTC and attention models. Each encoder operates separately on a parallel input $X^{(i)}$ to extract a set of frame-wise hidden vectors:

$$H^{(i)} = \text{Encoder}^{(i)}(X^{(i)}), \tag{3.1}$$

where $H^{(i)} = \{\mathbf{h}_t^{(i)} \in \mathbb{R}^{D^h} | t = 1, 2, ..., \lfloor T^{(i)}/s^{(i)} \rfloor\}$. $\mathbf{h}_t^{(i)}$ is a D^h-dimensional frame-wise hidden vector of stream i. Note that it is not mandatory to have frame-level synchronization across all streams since $T^{(i)}, i \in \{1, ..., N\}$, could be different in the proposed model. Because stream-specific subsampling factor $s^{(i)}$ is defined by the architecture of each encoder, stream i will have $\lfloor T^{(i)}/s^{(i)} \rfloor$ time instances at the encoder-output level. Rounding process of $\lfloor T^{(i)}/s^{(i)} \rfloor$ is performed in the encoder based on different types of architecture. For instance, in the multi-stream model

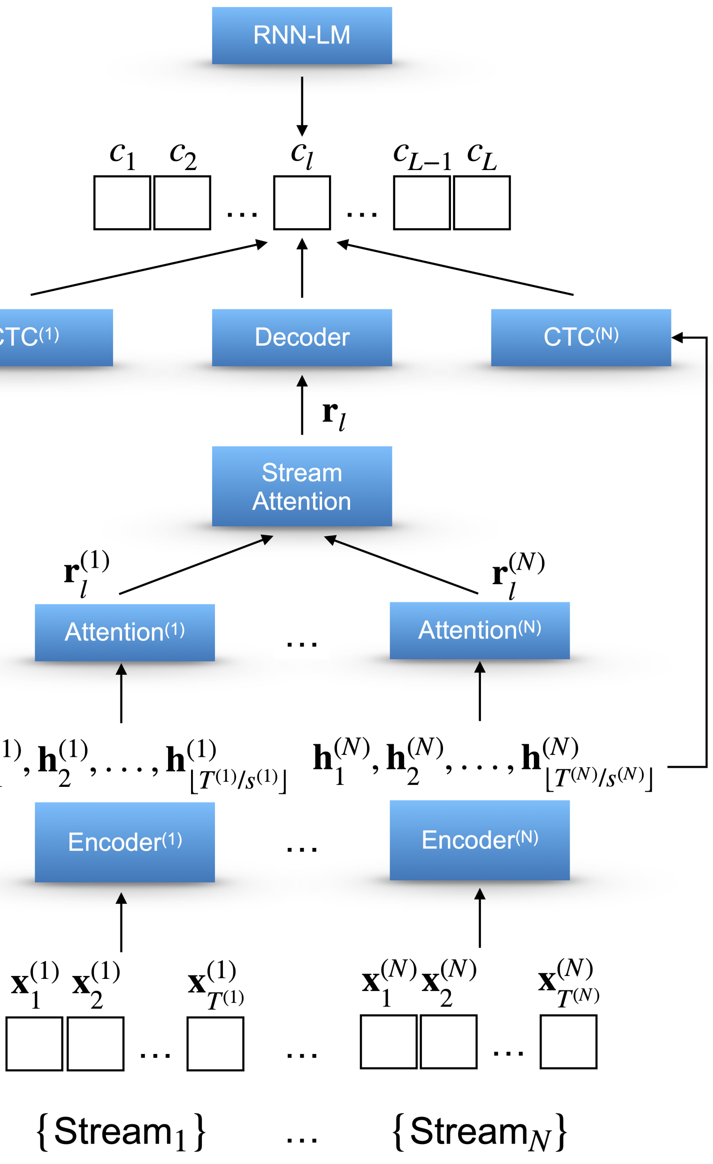

RNN-LM
c_1 c_2 c_l c_{L-1} c_L
CTC(1)
Decoder
CTC(N)
$\mathbf{r}_l$
Stream Attention
$\mathbf{r}_l^{(1)}$
$\mathbf{r}_l^{(N)}$
Attention(1)
Attention(N)
$\mathbf{h}_1^{(1)}, \mathbf{h}_2^{(1)}, \ldots, \mathbf{h}_{\lfloor T^{(1)}/s^{(1)} \rfloor}^{(1)}$
$\mathbf{h}_1^{(N)}, \mathbf{h}_2^{(N)}, \ldots, \mathbf{h}_{\lfloor T^{(N)}/s^{(N)} \rfloor}^{(N)}$
Encoder(1)
Encoder(N)
$\mathbf{x}_1^{(1)}$ $\mathbf{x}_2^{(1)}$ $\mathbf{x}_{T^{(1)}}^{(1)}$
$\mathbf{x}_1^{(N)}$ $\mathbf{x}_2^{(N)}$ $\mathbf{x}_{T^{(N)}}^{(N)}$
{Stream$_1$}
{Stream$_N$}

Figure 3.1. The proposed multi-stream end-to-end ASR framework

with $N = 2$, two encoders in parallel take different input features, $X^{(1)}$ with $T^{(1)}$ frames and $X^{(2)}$ with $T^{(2)}$ frames, respectively. Each encoder operates on different temporal resolution with subsampling factor $s^{(1)}$ and $s^{(2)}$, where subsampling could be performed in stacked RNN layers or max-pooling layers in CNN.

3.2.2 Hierarchical Attention

In the multi-stream setting, the contribution of each stream changes dynamically. Hence, a secondary stream attention, the HAN component, is exploited for the purpose of robustness. We adopt a hierarchical attention network for information fusion. The decoder with the HAN component is trained to selectively attend to an appropriate encoder, based on the context of previous prediction and higher-level acoustic features from encoders, to achieve better performance. With knowledge fusion in the secondary attention network, no time alignment constraint is placed for parallel inputs. For instance, the number of input frames for each stream could be different with hierarchical fusion.

For single stream E2E described in Sec. 2.1.2.3, there is only frame-level attention network connecting the encoder and the decoder. For the proposed multi-stream setting, each stream (encoder) is assigned with a separate frame-level attention network. To predict l-th character in the sentence, the stream-specific context vector, $\mathbf{r}_l^{(i)}$, is computed as follows:

$$\mathbf{r}_l^{(i)} = \sum\nolimits_{t=1}^{\lfloor T^{(i)}/s^{(i)} \rfloor} a_{lt}^{(i)} \mathbf{h}_t^{(i)}, \tag{3.2}$$

$$a_{lt}^{(i)} = \text{Attention}(\{a_{l-1}^{(i)}\}_{t=1}^{\lfloor T^{(i)}/s^{(i)} \rfloor}, \mathbf{q}_{l-1}, \mathbf{h}_t^{(i)}), \tag{3.3}$$

$$\sum_{t=1}^{\lfloor T^{(i)}/s^{(i)} \rfloor} a_{lt}^{(i)} = 1. \tag{3.4}$$

$a_{lt}^{(i)}$ is the attention weight, a soft-alignment of $\mathbf{h}_t^{(i)}$ for output c_l. $\mathbf{q}_{l-1}$ is the previous decoder state. In this work, content-based or location-based attention mechanisms, described in Sec. 2.1.2.2, are applied in the frame-level attention networks for different scenarios. Note that since the encoder may perform downsampling, summation is till $\lfloor T^{(i)}/s^{(i)} \rfloor$ for stream i in Eq. 3.2.

The fusion context vector $\mathbf{r}_l$ is obtained as a convex combination of $\mathbf{r}_l^{(i)}, i \in \{1, ..., N\}$, as illustrated in the following:

$$\mathbf{r}_l = \textstyle\sum_{i=1}^{N} \beta_l^{(i)} \mathbf{r}_l^{(i)}, \tag{3.5}$$

$$\beta_l^{(i)} = \mathrm{HierarchicalAttention}(\mathbf{q}_{l-1}, \mathbf{r}_l^{(i)}), \tag{3.6}$$

$$\sum_{t=1}^{N} \beta_{lt}^{(i)} = 1. \tag{3.7}$$

HierarchicalAttention$(\cdot)$ represents the secondary attention mechanism on top of the frame-level attention. As stated in Eq. 3.6, the stream-level attention weight, $\beta_l^{(i)}$, is estimated according to the previous decoder state $\mathbf{q}_{l-1}$ and context vector $\mathbf{r}_l^{(i)}$ from an individual encoder i. $\beta_l^{(i)}$ is one of Softmax outputs across N streams from the HAN component, a stream-level attention weight for stream i of prediction c_l. Content-based attention is used as the stream fusion component. An LSTM-based decoder network predicts the next letter based on $\mathbf{r}_l$ and the previous prediction c_{l-1}.

3.2.3 Training and Decoding with Per-encoder CTC

In the joint CTC/ATT model with a single encoder, the CTC objective serves as an auxiliary task to speed up the procedure of realizing monotonic alignment and providing a sequence-level objective. In the multi-stream framework, we introduce per-encoder CTC where a separate CTC mechanism is active for each encoder stream during training and decoding. Sharing one set of CTC among encoders is a soft constraint that limits the potential of diverse encoders to reveal complementary information. Sharing CTC refers to the case that linear layers connecting hidden vectors to CTC Softmax layers for each encoder are shared. In the case that encoders are with different temporal resolutions and network architectures, per-encoder CTC can further align speech with labels in a monotonic order and customize the sequence modeling of individual streams. For each CTC network, the frame-wise posterior probability $p(z_t^{(i)}|X^{(i)})$ is estimated in the following way:

$$p(z_t|X) = \mathrm{Softmax}(\mathrm{LinB}(\mathbf{h}_t^{(i)})) \tag{3.8}$$

where $\mathrm{LinB}(\cdot)$ is a linear layer with bias term converting $\mathbf{h}_t$ to a $(|\mathcal{U}|+1)$ dimensional vector, followed by a Softmax activation function and $z_t \in \mathcal{U} \bigcup blank$. Note that the "blank" symbol is used to handle the merging of repeating letters.

Similar to a single-stream model, the training objective function to be maximized is a logarithmic linear combination of the CTC and attention objectives, i.e., $p_{\mathrm{ctc}}(C|X)$ and $p^{\dagger}_{\mathrm{att}}(C|X)$:

$$\mathcal{L}_{\mathrm{MTL}} = \lambda \log p_{\mathrm{ctc}}(C|X) + (1-\lambda)\log p^{\dagger}_{\mathrm{att}}(C|X), \tag{3.9}$$

where λ is a tunable scalar satisfying $0 \leq \lambda \leq 1$. $p^{\dagger}_{\mathrm{att}}(C|X)$ is an approximated letter-wise objective where the probability of a prediction is conditioned on previous true labels. The CTC objective $\log p_{\mathrm{ctc}}(C|X)$ is computed in the following way:

$$\log p_{\mathrm{ctc}}(C|X) = \frac{1}{N}\sum\nolimits_{i=1}^{N} \log p_{\mathrm{ctc}^{(i)}}(C|X), \tag{3.10}$$

where joint CTC loss is the averaged per-encoder CTC losses.

During inference, the model performs a label-synchronous beam search. The most probable letter sequence $\hat{C}$ given the speech input X is computed according to

$$\hat{C} = \arg\max_{C \in \mathcal{U}^*} \{\lambda \log p_{\text{ctc}}(C|X) + (1-\lambda) \log p_{\text{att}}(C|X) + \gamma \log p_{\text{lm}}(C)\}, \tag{3.11}$$

where external RNN-LM probability $\log p_{\text{lm}}(C)$ is added with a scaling factor γ. For each partial hypothesis h in the beam search, the log probability of hypothesized label sequence can be computed as

$$\alpha(h) = \lambda \alpha_{\text{ctc}}(h) + (1-\lambda)\alpha_{\text{att}}(h) + \gamma \alpha_{\text{lm}}(h), \tag{3.12}$$

where $\alpha_{\text{att}}(h)$, $\alpha_{\text{ctc}}(h)$ and $\alpha_{\text{lm}}(h)$ are the hypothesis scores from attention decoder, CTC and RNN-LM, respectively. In the beam search, the CTC prefix score of hypothesized sequence h is altered as follows:

$$\alpha_{\text{ctc}}(h) = \frac{1}{N} \sum\nolimits_{i=1}^{N} \alpha_{\text{ctc}^{(i)}}(h), \tag{3.13}$$

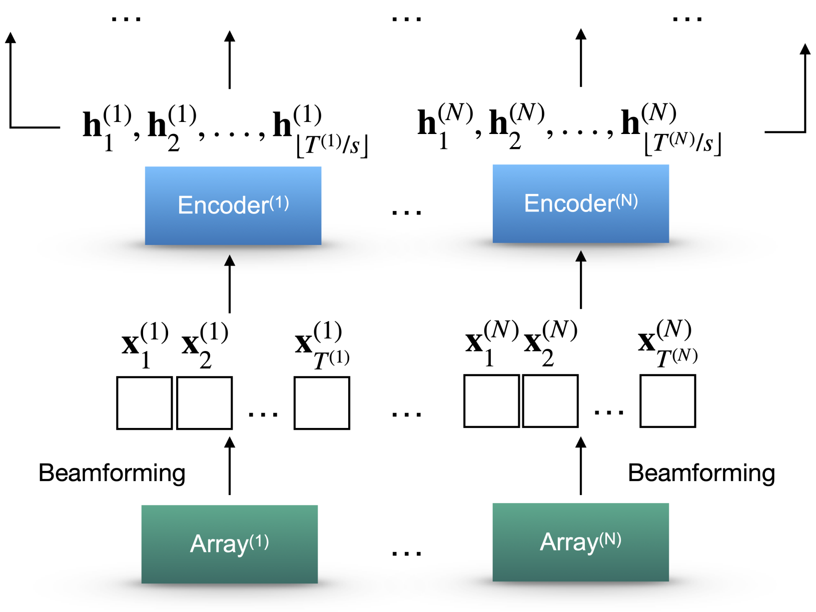
where equal weight is assigned to each CTC network.

3.3 Two Realizations

An end-to-end ASR model addressing the general multi-stream setting was introduced in the previous section. Two realizations of multi-stream framework are presented here, which are MEM-Array model and MEM-Res model targeting different applications. In multi-array scenarios, taking advantage of all the information that each array shared and contributed is crucial in this task. Different from multi-channel scenario, it is often not a guarantee that distant microphone arrays operate with synchronous signals. The MEM-Array model is proposed to fuse array knowledge dynamically with no presumption of time synchronization. In MEM-Res architecture, one acoustic input is characterized by two heterogeneous encoders with different configurations and temporal resolutions. The hierarchical attention then combines diverse views from both encoders.

3.3.1 Multi-Encoder Multi-Array

In this section, we briefly introduce the Multi-Encoder Multi-Array (MEM-Array) model. Detailed descriptions and experiments are carried out in the next chapter.



$\mathbf{h}_1^{(1)}, \mathbf{h}_2^{(1)}, \ldots, \mathbf{h}_{\lfloor T^{(1)}/s \rfloor}^{(1)}$
$\mathbf{h}_1^{(N)}, \mathbf{h}_2^{(N)}, \ldots, \mathbf{h}_{\lfloor T^{(N)}/s \rfloor}^{(N)}$
Encoder(1)
Encoder(N)
$\mathbf{x}_1^{(1)}$ $\mathbf{x}_2^{(1)}$ $\mathbf{x}_{T^{(1)}}^{(1)}$
$\mathbf{x}_1^{(N)}$ $\mathbf{x}_2^{(N)}$ $\mathbf{x}_{T^{(N)}}^{(N)}$
Beamforming
Beamforming
Array(1)
Array(N)

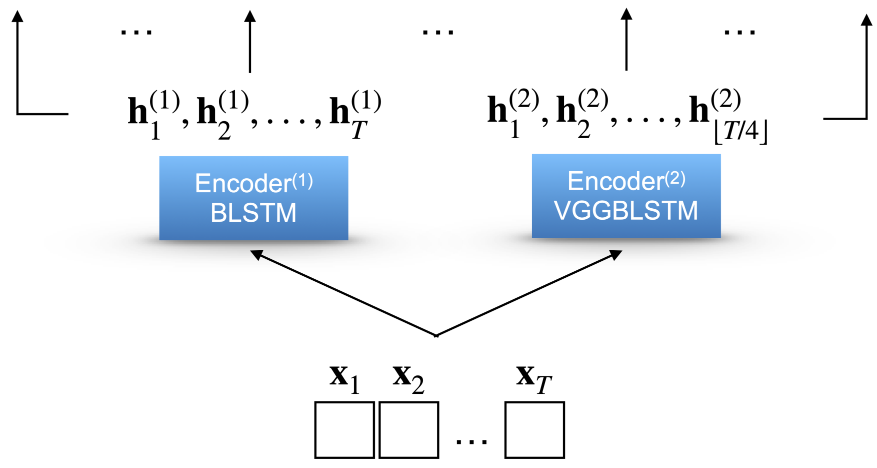
Figure 3.2. Part of the Multi-Encoder Multi-Array (MEM-Array) architecture

As one representative framework, MEM-Array concentrates on cases of far-field microphone arrays to handle different dynamics of streams. The architecture of N streams is shown in Fig. 3.2. In comparison to the general form of multi-stream model, we specify microphone-array inputs as parallel information sources. Identical encoder architecture is assigned to each stream. For each array, multi-channel signals are merged to one-channel input via beamforming technique. Each encoder operates separately on a parallel input $X^{(i)} = \{\mathbf{x}_1^{(i)}, ..., \mathbf{x}_{T^{(i)}}^{(i)}\}$ to extract a set of frame-wise hidden vectors $H^{(i)} = \{\mathbf{h}_1^{(i)}, ..., \mathbf{h}_{\lfloor T^{(i)}/s \rfloor}^{(i)}\}$:

$$H^{(i)} = \text{Encoder}^{(i)}(X^{(i)}), i \in \{1, ..., N\}. \tag{3.14}$$

Subsequently, a hierarchical attention mechanism is designed to combine information using encoder outputs $\{H^{(i)}\}_N^{i=1}$. In the multi-stream setting, one inherent problem is that the contribution of each stream (array) changes dynamically. Specially, when one of the streams takes corrupted audio, the network should be able to pay more attention to other streams for the purpose of robustness. A stream-level fusion on the letter-wise context vector is a natural fit to achieve the goal of encoder selectivity.

3.3.2 Multi-Encoder Multi-Resolution



$\mathbf{h}_1^{(1)}, \mathbf{h}_2^{(1)}, \ldots, \mathbf{h}_T^{(1)}$
$\mathbf{h}_1^{(2)}, \mathbf{h}_2^{(2)}, \ldots, \mathbf{h}_{\lfloor T/4 \rfloor}^{(2)}$
Encoder(1) BLSTM
Encoder(2) VGGBLSTM
$\mathbf{x}_1$ $\mathbf{x}_2$ $\mathbf{x}_T$

Figure 3.3. Part of the Multi-Encoder Multi-Resolution (MEM-Res) architecture

As another realization of multi-stream framework, we propose a Multi-Encoder Multi-Resolution (MEM-Res) architecture that has two encoders, RNN-based and CNN-RNN-based. Both encoders take the same input features in parallel operating on different temporal resolutions, aiming to capture complementary information in the speech as depicted in Fig. 3.3.

The RNN-based encoder, denoted as stream 1, is designed to model temporal sequences with their long-range dependencies. Subsampling in BLSTM is often used to decrease the computational cost, but performing subsampling might result in lost information which could be better modeled in RNN. In MEM-Res, the BLSTM encoder has only BLSTM layers that extract the frame-wise hidden vector $H^{(1)}$ without subsampling in any layer, i.e. $s^{(1)} = 1$.

The combination of CNN and RNN allows the convolutional feature extractor applied on the input to reveal local correlations in both time and frequency dimensions. The RNN block on top of CNN makes it easier to learn temporal structure from the CNN output, to avoid modeling direct speech features with more underlying variations. The pooling layer is essential in CNN to reduce the spatial size of the representation to control over-fitting. In MEM-Res, convolutional layers together with max-pooling layers are exploited as an initial network. It is followed by stacked BLSTM layers with no subsampling. This CNN-RNN-based encoder is labeled as stream 2.

Chapter 4

Multi-Encoder Multi-Array (MEM-Array)

4.1 Proposed Multi-Array Framework

Automatic speech recognition using multiple microphone arrays has achieved great success in the far-field robustness. Taking advantage of all the information that each array shares and contributes is crucial in this task. As one representative framework of multi-stream end-to-end ASR, a MEM-Array model is proposed in this chapter. Microphone arrays, acting as information streams, are activated by separate encoders and decoded under the instruction of both CTC and attention networks. On top of the regular frame-level attention networks, stream attention is introduced to steer the decoder toward the most informative encoders.

4.1.1 Introduction

Far-field ASR using multiple microphone arrays has been a widely adopted strategy in the speech processing community. Individually, the microphone array is able to bring a substantial performance improvement with algorithms such as beamforming [88] and masking [89]. For a single microphone array, signals from parallel channels are synchronized across all channels, where advanced beamforming techniques could be adopted to combine multi-channel inputs into single-channel audios. However, distributed microphone arrays are often without presumption of time synchronization. What kind of information can be extracted from each array and how to make multiple arrays work in cooperation are still challenging, especially in a far-field environment with noise and reverberations. For scenarios of multi-array ASR, the common ways of utilizing multiple arrays in hybrid systems are described in Sec. 2.2. This work tackles this issue under end-to-end ASR framework.

In the previous chapter, a general formation of multi-stream end-to-end ASR is introduced within a joint CTC/ATT model. In this chapter, we propose an attention-based multi-array E2E architecture, MEM-Array, to address the aforementioned issues in far-field ASR. The framework has highlights as follows:

1. The output of each microphone array is modeled by a separate encoder. Multiple encoders with the same configuration act as acoustic models for individual arrays.

2. The hierarchical attention mechanism is introduced to dynamically combine knowledge from parallel streams. We adopt this network in a multi-array scheme, where the stream-level fusion is employed on top of the frame-level attention mechanisms.

3. Each encoder is associated with a CTC network to guide the frame-wise alignment process for each array to potentially achieve a better performance.

4.1.2 MEM-Array Model

The MEM-Array architecture of N microphone arrays is illustrated in Fig. 4.1 as one of the practical multi-stream ASR scenarios. Following the same notation from Chapter

3, the MEM-Array model directly maps N parallel inputs, $X = \{X^{(1)}, X^{(2)}, ..., X^{(N)}\}$ into an L-length letter sequence, $C = \{c_l \in \mathcal{U} | l = 1, 2, ..., L\}$.

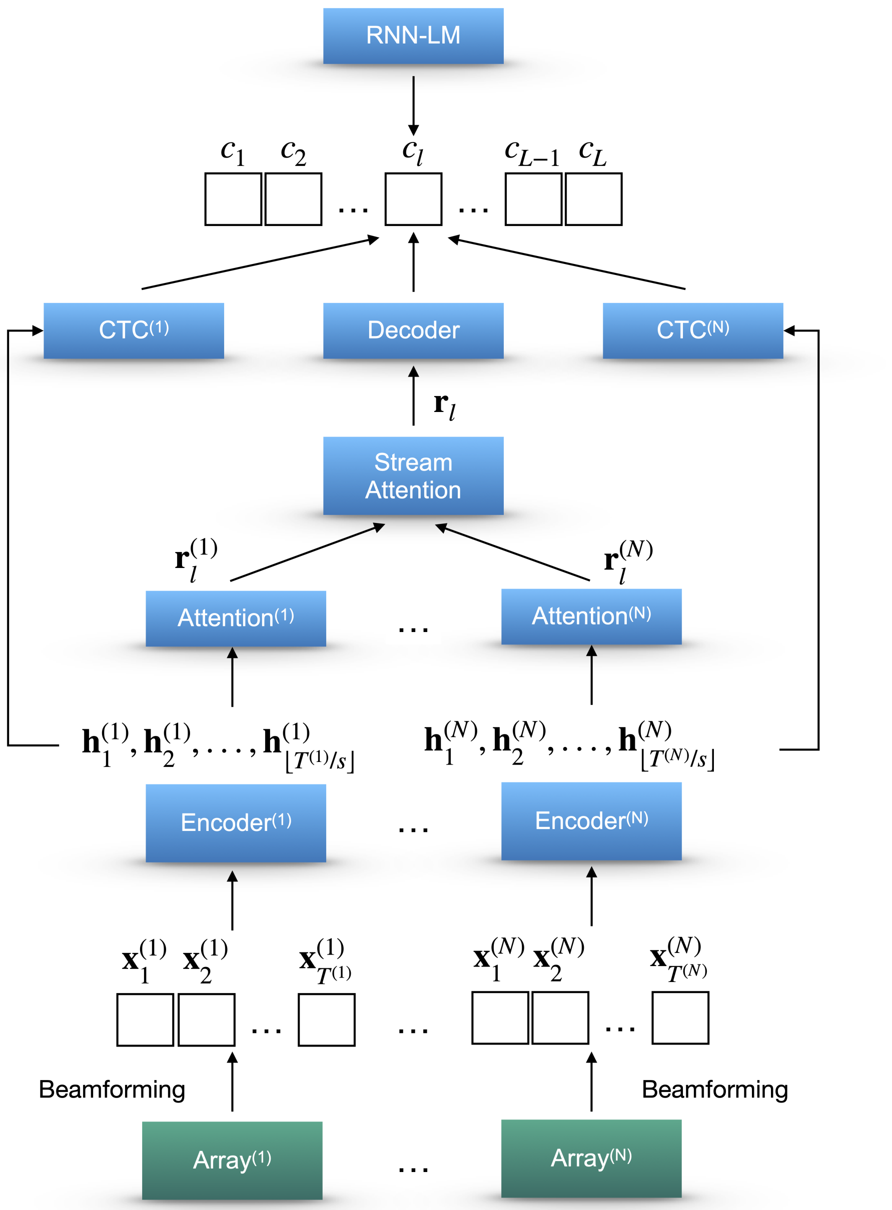

RNN-LM
c_1 c_2 … c_l … c_{L-1} c_L
CTC(1)
Decoder
CTC(N)
$\mathbf{r}_l$
Stream Attention
$\mathbf{r}_l^{(1)}$
$\mathbf{r}_l^{(N)}$
Attention(1)
Attention(N)
$\mathbf{h}_1^{(1)}, \mathbf{h}_2^{(1)}, \ldots, \mathbf{h}_{\lfloor T^{(1)}/s \rfloor}^{(1)}$
$\mathbf{h}_1^{(N)}, \mathbf{h}_2^{(N)}, \ldots, \mathbf{h}_{\lfloor T^{(N)}/s \rfloor}^{(N)}$
Encoder(1)
Encoder(N)
$\mathbf{x}_1^{(1)}$ $\mathbf{x}_2^{(1)}$ $\mathbf{x}_{T^{(1)}}^{(1)}$
$\mathbf{x}_1^{(N)}$ $\mathbf{x}_2^{(N)}$ $\mathbf{x}_{T^{(N)}}^{(N)}$
Beamforming
Beamforming
Array(1)
Array(N)

Figure 4.1. The Multi-Encoder Multi-Array (MEM-Array) architecture

4.1.2.1 Multi-Array Architecture with Stream Attention

In the MEM-Array model, multiple microphone arrays are activated by separate encoders with identical architectures to capture diverse information. The proposed architecture has N encoders, with each mapping the speech features of a single array $X^{(i)}$ to higher level representations $H^{(i)} = \{\mathbf{h}_1^{(i)}, ..., \mathbf{h}_{\lfloor T^{(i)}/s \rfloor}^{(i)}\}$ corresponding to array i:

$$H^{(i)} = \text{Encoder}^{(i)}(X^{(i)}), i \in \{1, ..., N\}. \tag{4.1}$$

Here, $\lfloor T^{(i)}/s \rfloor$ time instances are generated for each stream at the encoder-output level with a subsampling factor of s defined by the encoder architecture. Note that all of the encoders have the same configurations receiving parallel speech data collected from multiple microphone arrays. Each encoder is shared by a stream-specific frame-level attention and a CTC network.

Two levels of attention mechanisms are designated to combine the different views as stated in the previous chapter. A frame-level attention mechanism is assigned to each encoder to obtain the stream-specific speech-label alignment. A hierarchical stream-level attention mechanism then handles different dynamics across the streams.

In comparison to direct fusion on frame-wise hidden vectors $\mathbf{h}_t^{(i)}$, stream-level fusion can deal with temporal misalignment from multiple arrays at the stream level. Furthermore, adding an extra microphone array j could be simply implemented with an additional term $\beta_l^{(j)}\mathbf{r}_l^{(j)}$ in Eq. 3.5.

4.1.2.2 Training and Decoding with Per-encoder CTC

Following the architecture of general multi-stream framework in the previous chapter, we assign each encoder with a separate CTC network. Per-encoder CTC modules have predefined equal contributions for joint training and decoding, as described in Eq.

3.10 and Eq. 3.13, respectively. The loss function to be optimized during training and the objective function during label-synchronous beam search are the same as stated in chapter 3.

4.1.3 Data

Two dataset, DIRHA English WSJ and AMI Meeting Corpus are used to evaluate the MEM-Array model.

The DIRHA English WSJ [87] is a part of Distant-speech Interaction for Robust Home Applications (DIRHA) project which addresses the challenge of speech interaction via distant microphones. A total of 32 microphones are used in a domestic environment of a living room (26 microphones) and a kitchen (6 microphones). The microphone network consists of 2 6-mic circular arrays on the ceiling of the living-room and the kitchen, a linear array of 11 sensors in the living-room, and 9 single microphones distributed on the living-room walls. Two microphone arrays, Beam Circular Array (BCA) and Beam Linear Array (BLA) in the living room, are chosen as parallel streams for experiments in this section. The contaminated version of the original Wall Street Journal-0 (WSJ0) and WSJ1 corpora is used for training, providing room impulse responses for corresponding arrays. The development set for cross validation is simulated with typical domestic background noises and reverberations. The evaluation set has 409 read utterances from WSJ recorded by six native English speakers in a real domestic setting. During the recording, the speaker is asked to move to a different position and take a different orientation after reading several sentences.

The AMI meeting corpus [86] is created in three instrumented meeting rooms (Edinburgh, Idiap and TNO Room) focusing on developing meeting browsing technologies. There are 100 hours of far-field signal-synchronized recordings collected using microphone arrays placed in each room. There are two arrays placed in each meeting room. In every room, there is one 10 cm radius circular array between the speakers consisting of 8 omni-directional microphones. The setup of the second microphone array is different among the rooms: In the room of Edinburgh, it is a 10 cm radius circular array with 8 microphones placed at the end of the table; In Idiap, a second microphone array with 4 elements is mounted on the ceiling; A 10-element linear array is placed above the presentation screen in the TNO room. The training, development and evaluation sets are comprised of 81 hours, 9 hours and 9 hours of meeting recordings, respectively.

For both datasets, two microphone arrays, denoted by Arr_1 and Arr_2, are applied to train a MEM-Array model, where configuration of arrays for each dataset is described in Table 4.1. Note that for each array, multi-channel input is synthesized into a single-channel audio using Delay-and-Sum beamforming technique with BeamformIt Toolkit [93].

Table 4.1. Description of array configurations in the two-stream E2E experiments

Dataset	Arr_1	Arr_2
DIRHA	6-mic Beam Circular Array	11-mic Beam Linear Array
AMI	8-mic Circular Array	8-mic Circular Array (Edinburgh) 4-mic Circular Array (Idiap) 10-mic Linear Array (TNO)

4.1.4 Experiment Setup

The MEM-Array model with $N = 2$ streams is used for experiments and analysis. In this work, we explore two types of encoder structures: BLSTM (RNN-based) and

VGGBLSTM (CNN-RNN-based)[13, 92]:

$$\text{Encoder}^{(i)}() = \text{BLSTM}() \quad \text{or} \quad \text{VGGBLSTM}(). \tag{4.2}$$

Note that every recurrent layer is equipped with an additional projection layer on top of the outputs. In both encoder architectures, subsampling factors $s^{(1)} = s^{(2)} = 4$ is applied to decrease the computational cost. For BLSTM encoder, a subsampling factor of 2 is applied to the first two BLSTM layers. In terms of VGGBLSTM, we use the initial layers of the VGG net architecture [94], stated in table 4.2, followed by BLSTM layers as VGGBLSTM decoder. Two maxpooling layers with $stride = 2$ downsample the input features by a factor of $s^{(2)} = 4$ in both temporal and spectral directions. There is no subsampling in the recurrent layers.

Table 4.2. Initial six-layer VGG configurations

Layer	Configuration
Convolution 2D	in = 1, out = 64, filter = 3× 3
Convolution 2D	in = 64, out = 64, filter = 3× 3
Maxpool 2D	patch = 2×2, stride = 2×2
Convolution 2D	in = 64, out = 128, filter = 3× 3
Convolution 2D	in = 128, out = 128, filter = 3× 3
Maxpool 2D	patch = 2×2, stride = 2×2

The letter-wise context vectors, $\mathbf{r}_l^{(1)}$ and $\mathbf{r}_l^{(2)}$, from individual encoders are computed following Eq. 3.2:

$$\mathbf{r}_l^{(i)} = \sum_{t=1}^{T^{(i)}/4} a_{lt}^{(i)} \mathbf{h}_t^{(i)}, i \in \{1, 2\} \tag{4.3}$$

where the summation is performed from 1 to $T^{(i)}/4$ due to subsampling. The fusion context vector $\mathbf{r}_l$ is obtained as a combination of $\mathbf{r}_l^{(1)}$ and $\mathbf{r}_l^{(2)}$ as illustrated:

$$\mathbf{r}_l = \beta_l^{(1)} \mathbf{r}_l^{(1)} + \beta_l^{(2)} \mathbf{r}_l^{(2)} \tag{4.4}$$

$$\beta_l^{(i)} = \text{HierarchicalAttention}(\mathbf{q}_{l-1}, \mathbf{r}_l^{(i)}), i = 1, 2 \tag{4.5}$$

The stream-level attention weights $\beta_l^{(1)}$ and $\beta_l^{(2)}$ are estimated according to the feedback from the previous decoder state, $\mathbf{q}_{l-1}$, and context vectors, $\mathbf{r}_l^1$ and $\mathbf{r}_l^2$, from individual encoders. The fusion context vector is then fed into the decoder to predict the next letter.

During multi-task training and joint decoding, we follow the formulas depicted by Eq. 3.10 and Eq. 3.13 with $N = 2$:

$$\log p_{\text{ctc}}(C|X) = \frac{\lambda}{2}(\log p_{\text{ctc}_1}(C|X) + \log p_{\text{ctc}_2}(C|X)), \tag{4.6}$$

$$\alpha_{\text{ctc}}(h) = \frac{1}{2}(\alpha_{\text{ctc}_1}(h) + \alpha_{\text{ctc}_2}(h)). \tag{4.7}$$

In this work, we integrate the look-ahead word-based language model to give the RNN-LM score, $\alpha_{lm}(h)$.

All the experiments were implemented by ESPnet, an end-to-end speech processing toolkit [95] with the configuration as described in Table 4.3. In all experiments, 80-dimensional mel-scale filterbank coefficients with additional 3-dimensional pitch features serve as the input features. We use 52 distinct labels including 26 English letters and other special tokens, i.e., punctuations and sos/eos.

4.1.5 Results and Analysis

We start with discussion on single-stream architectures, followed by analysis of the effectiveness of our proposed MEM-Array model.

Table 4.3. Experimental configuration

Feature	
Each Stream	80-dim log-mel filter bank + 3-dim pitch
Number of Streams	2
Model	
Encoder type	*BLSTM* or *VGGBLSTM*
Encoder layers	*VGGBLSTM*: 6(VGG)+4(BLSTM)
	BLSTM: 4(BLSTM)
Encoder units	320 cells (BLSTM layers)
Encoder projection	320 cells (BLSTM layers)
Subsampling	4
Frame-level Attention	320-cell Content-based
Stream Attention	320-cell Content-based
Decoder type	LSTM
Decoder layers	1
Decoder units	300 cells
Decoder Softmax	52 labels
	(26 English letters+punctuation+sos/eos)
Train and Decode	
Optimizer	AdaDelta
Batch size	15
Training Epoch	15 epochs
CTC weight λ (train)	DIRHA:0.2; AMI:0.5
CTC weight λ (decode)	DIRHA:0.3; AMI:0.3
Label Smoothing	Type: Unigram [96], Weight: 0.05
Beam size	30
RNN-LM	
Type	Look-ahead Word-level RNN-LM
Size	1-Layer LSTM with 1,000 cells
Vocabulary	65,000
Train data	DIRHA:WSJ0-1+extra WSJ text; AMI:AMI
LM weight γ	DIRHA:1.0; AMI:0.5
Optimizer	Stochastic Gradient Descent
Batch size	300
Training Epoch	DIRHA: 60; AMI: 20
Learning Rate	DIRHA: 0.5; AMI: 1.0

4.1.5.1 Single-Array Model

First, we explore the ASR performance for the individual array (single stream). As illustrated in Table 4.4, the single-stream system with VGGBLSTM encoder noticeably outperforms the one with BLSTM encoder, both in Character Error Rate (CER) and Word Error Rate (WER). Joint training of CTC and attention-based model helps in terms of CERs since CTC can enforce the monotonic behavior of attention alignments, rather than merely estimating the desired alignment for long sequence. With the help of word-level RNN-LM during inference , we observed substantial improvements of the WERs on both datasets. The WERs of Arr_1 are 35.1% and 56.9% for DIRHA Real and AMI Eval, respectively. The architecture with the best performance (VGGBLSTM+CTC+ATT+RNN-LM) is chosen for further experiments on both streams.

Table 4.4. Exploration of best encoder and decoding strategy for single-stream E2E model

Model (Single Stream)	DIRHA Real		AMI Eval	
	CER	WER	CER	WER
BLSTM (Arr$_1$)				
Attention	42.7	68.7	45.1	60.9
+ CTC	38.5	74.8	41.7	63.0
+ Word RNN-LM	29.4	47.4	41.7	59.1
VGGBLSTM (Arr$_1$)				
Attention	39.5	71.4	43.2	59.7
+ CTC	30.1	61.8	40.2	62.0
+ Word RNN-LM	**21.2**	**35.1**	**39.6**	**56.9**
VGGBLSTM (Arr$_2$)	**22.5**	**38.4**	**45.6**	**64.0**

4.1.5.2 MEM-Array v.s. Conventional Methods

As illustrated in Table 4.5, our proposed framework is able to fuse information successfully from both streams by achieving lower error rates than best single-array systems, i.e., DIRHA (35.1% → 31.7%) and AMI (56.9% → 54.9%). Moreover, several conventional fusion strategies are discussed in Table 4.5: signal-level fusion through WAV alignment and average; feature-level frame-by-frame concatenation; word-level prediction fusion using ROVER. The MEM-Array model outperforms all three fusion techniques, even including the case when doubling BLSTM layers in signal-level fusion for a comparable amount of parameters (33.7 million versus 31.6 million).

Table 4.5. Comparison between proposed multi-stream approach and alternative single-stream strategies (WER %)

Encoder *VGGBLSTM* (ATT + CTC + RNN-LM)	#Param	DIRHA Real	AMI Eval
Single-stream model			
Concatenating Arr_1&Arr_2	23.3M	33.5	56.7
WAV alignment and average	26.2M	43.5	56.7
+ model parameter extension	33.7M	39.6	56.9
Two single-stream models			
ROVER Arr_1&Arr_2	52.5M(26.2×2)	37.0	60.7
Multi-stream model			
Proposed framework	31.6M	**31.7**	**54.9**

4.1.5.3 Stream Fusion with Noise Corruption

To investigate the robustness of stream attention, we design an experiment with Arr_1 injected with zero-mean, unit-variance Gaussian noise in the signal level while keeping Arr_2 untouched. Fig. 4.2 displays an example from DIRHA evaluation set during inference. Noise corruption Arr_1 $((a) \rightarrow (c))$ makes attention alignments fairly blurred, thus less trusted. As expected, a positive average shift of stream weights towards Arr_2 is observed (upper yellow line in Fig. 4.2(e)).

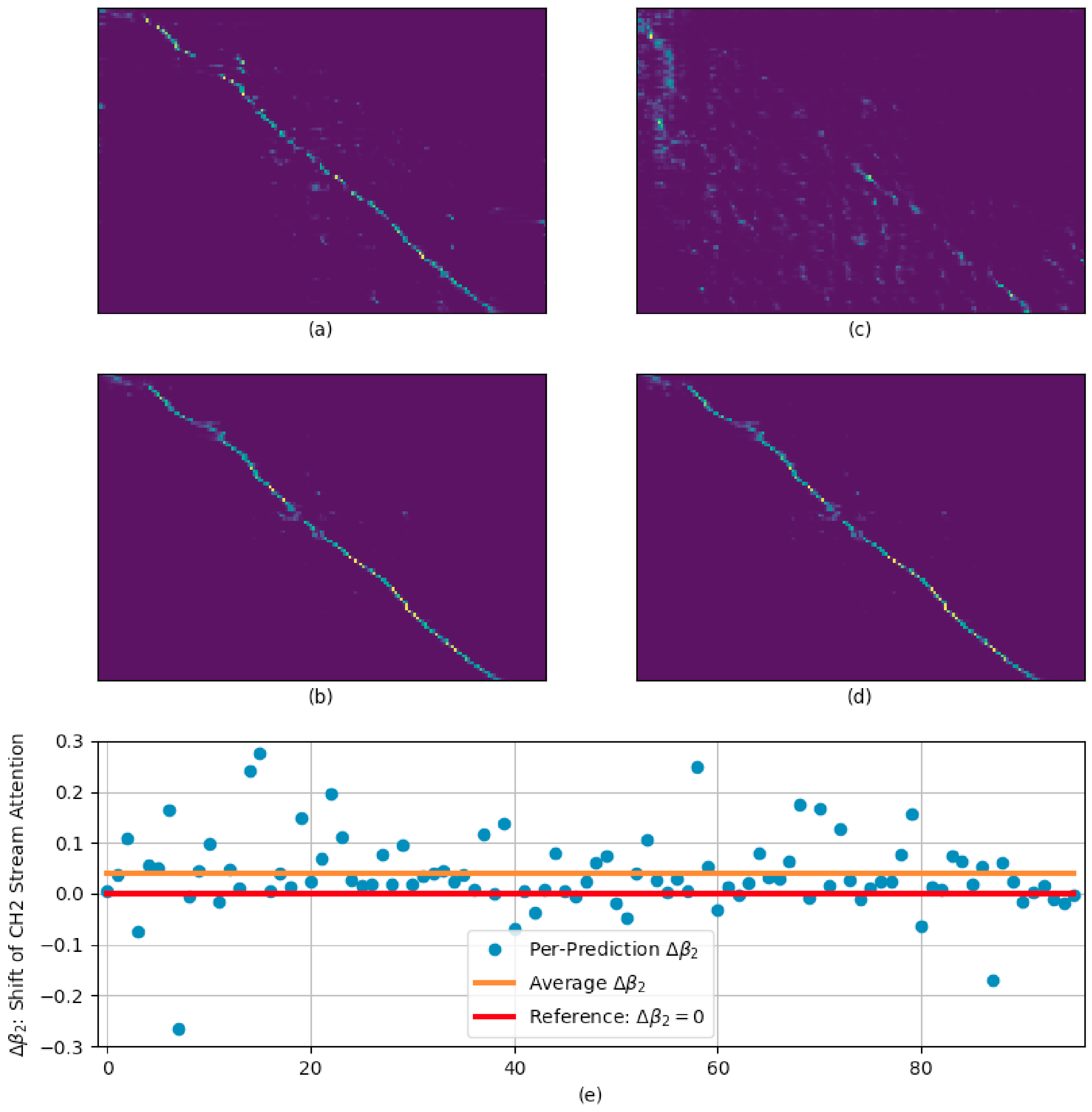


Figure 4.2. Comparison of the alignments between characters (y-axix) and acoustic frames (x-axis) before (**(a)** Arr_1; **(b)** Arr_2) and after (**(c)** Arr_1; **(d)** Arr_2) noise corruption of Arr_1. **(e)** shows the attention weight shift of Arr_2 between two cases (x-axis is the letter sequence).

4.1.5.4 Comparison with Hybrid Model

Table 4.6 shows fusion results of six streams in conventional hybrid ASR systems from a previous study [68]. DNN posterior combination approach and ROVER technique were used in [68] for stream fusion. Relative WER reductions of 7.2% and 5.8% were reported compared to the best single stream performance, respectively. Meanwhile, the MEM-Array system with two streams reduces the WER by 9.7% relatively. In spite of more training data involved in E2E, MEM-Array shows a promising direction for fusion of more streams.

Table 4.6. Comparison between the hybrid and end-to-end system on DIRHA dataset. #Streams denotes the number of streams. (WER %)

System	#Streams	Method	Best Single Stream	Multi Stream
Hybrid	6	Posterior Combination	29.2	27.1 (**7.2%**)
	6	ROVER	29.2	27.5 (**5.8%**)
E2E	2	MEM-Array	35.1	31.7 (**9.7%**)

4.1.6 Conclusion

In this section, we present a multi-stream end-to-end ASR framework, MEM-Array, targeting the distributed microphone array situation. Stream attention is achieved through a hierarchical connection between the decoder and encoders, with each modeling one array into higher-level representations. Thanks to the success of joint training of per-encoder CTC and attention, substantial WER reductions are shown in both DIRHA and AMI corpora, demonstrating the potentials of the proposed architecture. Compared with the best single-array results, the proposed framework has achieved relative WER reductions of 3.7% and 9.7% in the two datasets, respectively, which is better than conventional strategies as well.

4.2 A Practical Two-Stage Training Strategy

The previous section offers a promising direction within end-to-end ASR, where parallel encoders aim to capture diverse information followed by a stream-level fusion based on attention mechanisms to combine the different views. However, with an increasing number of streams resulting in an increasing number of encoders, the previous approach could require substantial memory and massive amounts of parallel data for joint training. In this section, we propose a practical two-stage training scheme. Stage-1 is to train a Universal Feature Extractor (UFE), where encoder outputs are produced from a single-stream model trained with all data. Stage-2 formulates a multi-stream scheme intending to solely train the attention fusion module using the UFE features and pre-trained components from Stage-1.

4.2.1 Introduction

The multi-stream paradigm in speech processing considers scenarios where parallel streams carry diverse or complementary task-related knowledge. In these cases, an appropriate strategy to fuse streams or select the most informative source is necessary. The work that follows considers far-field ASR using multiple microphone arrays, a specific case of multi-stream paradigm. Without any knowledge of speaker-array distance or orientation, it is still challenging to speculate which array is most informative or least corrupted.

In Chapter 3, we propose a novel multi-stream model based on a joint CTC/ATT E2E scheme, where each stream is characterized by a separate encoder and CTC network. A hierarchical attention network acts as a fusion component to dynamically assign higher weights for streams carrying more discriminative information for prediction. The Multi-Encoder Multi-Array (MEM-Array) framework is introduced in

Sec. 4.1 to improve the robustness of distant microphone arrays, where each array is represented by a separate encoder. While substantial improvements were reported within a two-stream configuration, there are two concerns when more streams are involved. First, during training, fitting all parallel encoders in device computing memory is potentially impractical for joint optimization, as the encoder is typically the largest component by far, i.e., 88% of total parameters in this work. Second, due to the data-hungry nature of DNNs and the expensive cost of collecting parallel data, training multiple models with excess degrees of freedom is not optimal.

In this section, we present a practical two-stage training strategy on the MEM-Array framework targeting the aforementioned issues. The proposed technique has the following highlights:

1. In Stage-1, a single-stream model is trained using all data for better model generalization. The encoder will then acts as a Universal Feature Extractor (UFE) to process parallel data individually to generate a set of high-level parallel features.

2. Initializing components (CTC, decoder, frame-level attention) from Stage-1, Stage-2 training only optimizes the HAN component operating directly on UFE parallel features. The resulting memory and computation savings greatly simplify training, potentially allowing for more hyperparameter exploration or consideration of more complicated architectures.

3. Lack of adequate volume of data, specially parallel data, leads to overfit or is hard to tackle unseen data. The proposed two-stage strategy better defines the data augmentation scheme. Augmentation in Stage-1 aims to extract more discriminative high-level features and provides well pre-trained modules for

Stage-2, whereas Stage-2 could focus on improving the robustness of information fusion.

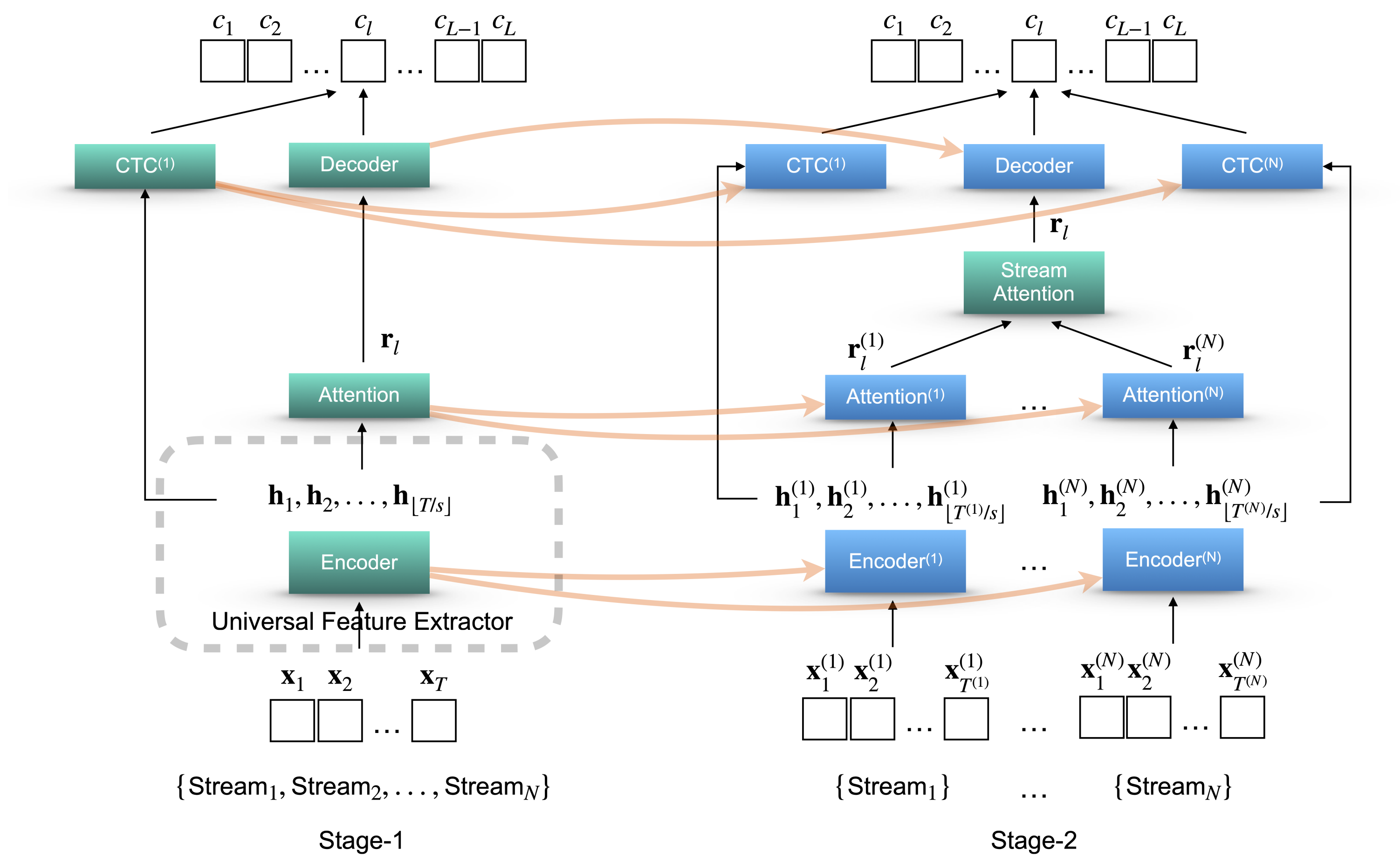
4.2.2 Two-Stage Training Strategy

In this section, we present a practical two-stage training strategy for the MEM-Array model, depicted in Fig. 4.3. The basics of MEM-Array model is described in Sec. 4.1. The details of each stage are discussed in the following.

4.2.2.1 Stage 1: Universal Feature Extractor

The intent of Stage-1 is to obtain a single well-trained encoder, which we refer to as universal feature extractor, to prepare a new set of high-level features for Stage-2. Encoder in E2E model can be viewed as an acoustic model that generates sequences $H = \{\mathbf{h}_1, ..., \mathbf{h}_{\lfloor T/s \rfloor}\}$ with more discriminative power for prediction. We denote the encoder outputs H as the UFE features. In general, the majority of the overall parameters are contained in the encoder. Introducing such a highly-parameterized nonlinear model, it is crucial in the task to make sure a well-generalized encoder.

In Stage-1, a single-stream joint CTC/ATT model is optimized as shown in Fig. 4.3. Audio features from all available streams are used to train the model. After training, we extract UFE features $H^{(i)} = \{\mathbf{h}_1^{(i)}, ..., \mathbf{h}_{\lfloor T^{(i)}/s \rfloor}^{(i)}\}$ for each stream i, separately. Since subsampling mitigates the increased dimension of UFE features, it is possible to save the UFE features at a similar size to the original speech features. Moreover, byproducts in Stage-1, such as decoder, CTC and frame-level attention, can be used for initialization in Stage-2.



c_1 c_2 … c_l … c_{L-1} c_L
CTC(1)
Decoder
$\mathbf{r}_l$
Attention
$\mathbf{h}_1, \mathbf{h}_2, \ldots, \mathbf{h}_{\lfloor T/s \rfloor}$
Encoder
Universal Feature Extractor
$\mathbf{x}_1$ $\mathbf{x}_2$ … $\mathbf{x}_T$
$\{\text{Stream}_1, \text{Stream}_2, \ldots, \text{Stream}_N\}$
Stage-1
c_1 c_2 … c_l … c_{L-1} c_L
CTC(1)
Decoder
CTC(N)
$\mathbf{r}_l$
Stream Attention
$\mathbf{r}_l^{(1)}$
$\mathbf{r}_l^{(N)}$
Attention(1)
Attention(N)
$\mathbf{h}_1^{(1)}, \mathbf{h}_2^{(1)}, \ldots, \mathbf{h}_{\lfloor T^{(1)}/s \rfloor}^{(1)}$
$\mathbf{h}_1^{(N)}, \mathbf{h}_2^{(N)}, \ldots, \mathbf{h}_{\lfloor T^{(N)}/s \rfloor}^{(N)}$
Encoder(1)
Encoder(N)
$\mathbf{x}_1^{(1)}$ $\mathbf{x}_2^{(1)}$ … $\mathbf{x}_{T^{(1)}}^{(1)}$
$\mathbf{x}_1^{(N)}$ $\mathbf{x}_2^{(N)}$ … $\mathbf{x}_{T^{(N)}}^{(N)}$
$\{\text{Stream}_1\}$ … $\{\text{Stream}_N\}$
Stage-2

Figure 4.3. Proposed two-stage training strategy. Color “green” indicates the components are trainable; Color “blue” means parameters of the components are frozen.

4.2.2.2 Stage 2: Parallel-Stream Fusion

As illustrated in Fig. 4.3, Stage-2 focuses on training the fusion component within the multi-stream context. The MEM-Array model uses parallel encoders as information streams. The previous strategy uses joint training with multiple large encoders, which is expensive in memory and time for more complex models or more streams. Taking advantage of UFE features greatly alleviates this complication.

In Stage-2, we formulate a multi-stream scheme on UFE features $\{H^{(i)}\}_{i=1}^{N}$ as parallel inputs. In this model, parameters of all components, except the stream attention module, are initialized from Stage-1 and frozen during optimization. The stream fusion component is randomly initialized, and is the only trainable element in Stage-2. Without any involvement of encoders, frame-level attention directly operates on UFE features. This setup not only reduces the amount of required parallel data, but it also greatly reduces memory and time requirements, allowing for more thorough hyperparameter exploration or utilization of more complex architectures.

4.2.3 Data

We demonstrated the two-stage training strategy using the same datasets: DIHRA English WSJ and AMI Meeting Corpus. Detailed information for both datasets are introduced in Sec. 4.1.3. We designed both 2-stream and 3-stream settings for DIRHA and 2-stream experiments for AMI. For DIRHA English WSJ, a single microphone (L1C) is picked to serve as a third stream in 3-stream experiments, while keeping the original two selected streams.

4.2.4 Experiment Setup

Experiments are conducted using a Pytorch back-end on ESPnet configured as described in Table 4.7

4.2.5 Results and Analysis

Firstly, we examine UFE features in a single-stream setting. Next, the full proposed strategy is analyzed in comparison to the previous approach as well as to several conventional fusion methods on DIRHA 2-stream case. Results on AMI and extension with more streams on DIRHA are explored as well. Lastly, we consider the potential benefits of data augmentation in this framework.

4.2.5.1 Multi-Stream v.s. Conventional Methods

In Table 4.8, the MEM-Array model with our two-stage training strategy consistently outperforms the baseline model which needs joint training after random initialization. 18.8%, 32.4%, and 8.2% relative WER reductions are achieved in 2-stream DIRHA, 3-stream DIRHA, and 2-stream AMI, respectively. Note that AMI experiments are conducted using VGGBLSTM with 2-layer BLSTM layers without any close-talk recordings and data perturbations. It is worth mentioning that those reductions in WERs are accomplished while simultaneously significantly decreasing the number of unique parameters in training by avoiding costly multiples of the large encoder component. For instance, in our setup, one single encoder has 10 million parameters which is 88% of the total parameters in a single-stream model. Two-stage training could greatly avoid creating separate encoders for individual streams.

In addition, results from several conventional fusion strategies are shown in Table 4.8: signal-level fusion via WAV alignment and average; feature-level frame-by-frame

Table 4.7. Experimental configuration

Feature	
Each Stream	80-dim log-mel filter bank + 3-dim pitch
Number of Streams	2 or 3
Model	
Encoder type	VGGBLSTM
Encoder layers	6(VGG)+2(BLSTM)
Encoder units	320 cells (BLSTM layers)
Encoder projection	320 cells (BLSTM layers)
Subsampling	4
Frame-level Attention	320-cell Location-based
Stream Attention	320-cell Content-based
Decoder type	LSTM
Decoder layers	1
Decoder units	300 cells
Decoder Softmax	52 labels (26 English letters+punctuation+sos/eos)
Train and Decode	
Optimizer	AdaDelta
Batch size	15
Training Epoch	30 epochs (patience:3 epochs)
CTC weight λ (train)	0.2
CTC weight λ (decode)	0.3
Label Smoothing	Type: Unigram, Weight: 0.05
Beam size	30
RNN-LM	
Type	Look-ahead Word-level RNN-LM
Size	1-Layer LSTM with 1,000 cells
Vocabulary	65,000
Train data	DIRHA:WSJ0-1+extra WSJ text; AMI:AMI
LM weight γ	DIRHA:1.0; AMI:0.5
Optimizer	Stochastic Gradient Descent
Batch size	300
Training Epoch	20
Learning Rate	1.0

concatenation; word-level prediction fusion using ROVER. For fair comparison, single-level and word-level fusion models utilized Stage-1 pre-trained models as their initialization. Note that word-level fusion operates on decoding results from pre-trained single-stream from Stage-1. Still, our proposed strategy consistently performs better than all other fusion methods in all conditions.

Table 4.8. Comparison between proposed two-stage approach and alternative conventional methods (WER %)

	Unique	# Streams		
	Params	DIRHA		AMI
Model	(in million)	2	3	2
MEM-Array Model				
Baseline [Sec. 4.1]	21.8(2),32.1(3)	33.0	32.1	59.5
Proposed Strategy	11.6	**26.8**	**21.7**	**54.6**
Other Fusion Methods				
WAV Align.& Avg.	11.4	32.4	30.1	55.9
Frame Concat.	16.9(2),23.8(3)	33.7	33.8	59.4
ROVER	11.4	34.2	23.6	58.0

4.2.5.2 Effectiveness of Stage-1 Training

In this section, we discuss the results on 2-stream DIRHA to demonstrate the value of proposed strategy. First, to evaluate Stage-1 training, CER/WER results on single stream systems are summarized in Table 4.9. Training the model using data from both streams improves performance substantially on the individual arrays, i.e., 37.6% $\rightarrow$ 33.9% and 39.2% $\rightarrow$ 30.7%. The UFE features are the outputs of an encoder trained with this improved strategy. In our setup, 320-dimensional UFE features take slightly smaller space than 83-dimensional acoustic frames since the subsampling factor $s = 4$.

Table 4.9. Stage-1 results on 2-stream DIRHA

	Arr_1		Arr_2	
Train Data	CER(%)	WER(%)	CER(%)	WER(%)
Single Stream				
Arr_1	22.3	37.6	–	–
Arr_2	–	–	23.0	39.2
$\text{Arr}_1, \text{Arr}_2$	**20.1**	**33.9**	**17.9**	**30.7**

4.2.5.3 Comparison among various Stage-2 Training Strategies

Table 4.10 illustrates several training strategies in Stage-2. Since Stage-2 operates on UFE features directly, its training only involves, at most, frame-level attention (ATT), decoder (DEC), hierarchical attention (HAN) and CTC. These experiments consider which of these components should be initialized from their Stage-1 counterparts, as well as which components should be fine-tuned or frozen during Stage-2 updates. In both cases of fine-tuning or freezing Pre-Trained (PT) modules in Stage-2, more noticeable improvements are reported with introducing more pretraining knowledge, i.e., 32.9% $\rightarrow$ 28.4% and 31.8% $\rightarrow$ 26.8%, respectively. Moreover, keeping all PT components frozen during Stage-2 and training solely the fusion module shows relative WER reduction of 5.6% (28.4% $\rightarrow$ 26.8%) with only 0.2 million active parameters. Overall, a substantial improvement of 18.8% relative WER reduction (33.0% $\rightarrow$ 26.8%) is observed compared to jointly training a massive model, including encoders, from scratch.

4.2.5.4 Discussion on Data Augmentation

The two-stage training strategy provides various opportunities for data augmentation. Stage-1 does not consider parallel data, so any augmentation technique for regular E2E ASR could be applied in this stage to improve the robustness of the UFE. Stage-2

Table 4.10. Comparison among various Stage-2 training strategies on 2-stream DIRHA. Note that components with random initialization in Stage-2 are listed in parentheses of the first column. The amount of trainable parameters in Stage-2 when freezing Stage-1 Pre-Trained (PT) components is stated in parentheses of the last column. (WER %)

Initialization with PT Comp. (rand. init. comp.)	Fine-tune PT Comp.	Freeze PT Comp.
No Two-Stage		
Baseline	–	33.0 (21.82M)
Two-Stage		
– (ATT, DEC, CTC, HAN)	32.9	31.8 (1.78M)
CTC (ATT, DEC, HAN)	34.4	30.7 (1.75M)
ATT (DEC, CTC, HAN)	33.3	30.6 (1.37M)
ATT, DEC (CTC, HAN)	29.0	27.4 (0.23M)
ATT, DEC, CTC (HAN)	**28.4**	**26.8** (0.20M)

augmentation, on the other hand, would be expected to improve robustness of the combination of corrupted individual streams. In this study, we employ a simple data augmentation technique called SpecAugment [97], which randomly removes sections of the input signal in a structured fashion [61], to demonstrate the potential of this direction. SpecAugment is applied to input frames instead of UFE features in all experiments. Table 4.11 shows the results from applying SpecAugment on two separate training stages. For experiments on DIRHA, the best performance is from data augmentation on Stage-1 when freezing all Stage-1 pre-trained components. With additional Stage-2 SpecAugment, there is not a noticeable difference in terms of WERs (22.6% v.s. 22.4% and 22.6% v.s. 22.5%). 5% absolute WER reduction is achieved in AMI with two stage augmentation. However, it is important to remember that, while the performance gap from fine-tuning versus freezing pre-trained components is narrowed with Stage-2 augmentation, the reductions in Stage-2 memory and computation requirements are still substantially better with frozen parameters.

Table 4.11. Performance investigation of two-stage data augmentation using SpecAugment on 2-stream DIRHA and AMI (WER %)

	DIHRA		
	Fine-tune	Freeze	
Model	PT Comp.	PT Comp.	AMI
Augmentation			
no SpecAugment	28.4	26.8	54.6
Stage-1	22.6	**22.4**	55.8
Stage-1, Stage-2	22.5	22.6	**49.2**

4.2.6 Conclusion

In this section, we propose a practical two-stage training strategy to improve multi-stream end-to-end ASR. A universal feature extractor is trained in Stage-1 with all available data. In Stage-2, a set of high-level UFE features are used to train a multi-stream model without requiring highly-parameterized parallel encoders. This two-stage strategy remarkably alleviates the burden of optimizing a massive multi-encoder model while still substantially improving the ASR performance. Separate stages focusing on training encoder and fusion module significantly improve the performance in the multi-stream setting. Experiments have been conducted on two datasets, DIRHA and AMI, as a multi-stream scenario. Compared with our previous method, this strategy achieves relative WER reductions of 8.2–32.4%, while consistently outperforming several conventional combination methods.

4.3 Improved Robustness of Multi-Stream End-to-End ASR

Performance degradation of an ASR system is commonly observed when the test acoustic condition is different from training. Hence, it is essential to make ASR systems robust against various environmental distortions, such as background noises and reverberations. In a multi-stream paradigm, improving robustness takes account of handling a variety of unseen single-stream conditions and inter-stream dynamics. In the previous section, a practical two-stage training strategy is proposed within multi-stream end-to-end ASR, where Stage-2 formulates the multi-stream model with features from Stage-1 universal feature extractor. In this section, as an extension, we introduce a two-stage augmentation scheme focusing on mismatch scenarios: Stage-1 Augmentation aims to address single-stream input varieties with data augmentation techniques; Stage-2 Time Masking applies temporal masks on UFE features of randomly selected streams to simulate diverse stream combinations. During inference, we also present adaptive CTC fusion with the help of hierarchical attention mechanisms.

4.3.1 Introduction

The multi-stream paradigm of speech processing has been an active research area, in which parallel information sources are simultaneously considered for knowledge fusion. A robust fusion strategy is crucial to reliably address a variety of scenarios with different dynamics across streams. This work concentrates on the setting of multiple far-field microphone arrays, e.g., meeting rooms or domestic scenarios.

The multi-stream end-to-end framework is presented in Chapter 3, in which the MEM-Array model is introduced for multi-array applications in Sec. 4.1. It is a single neural network that takes multiple inputs and directly outputs word/letter sequences.

This framework is proposed based on a joint CTC/ATT E2E scheme, where each stream is characterized by a separate encoder and CTC network. A hierarchical attention network acts as a fusion component to dynamically guide the system towards streams carrying more discriminative information. A practical two-stage training strategy is introduced later in Sec. 4.2. In Stage-1, an universal feature extractor is optimized without requiring parallel data; Stage-2 formulates a multi-stream model directly on the UFE features with focus on solely training the HAN component.

The two-stage training strategy in Sec. 4.2 offers a promising direction to further improve the robustness of multi-stream systems. It involves augmentation of training data, with an emphasis on single-stream variations in Stage-1 and inter-stream dynamics in Stage-2. Moreover, predefined equal CTC contributions during inference can potentially confuse the decoding procedure, especially when acoustic conditions among streams are dramatically different.

In this section, we present a two-stage augmentation scheme and adaptive CTC fusion targeting the aforementioned situations. The proposed techniques have the following highlights:

1. Stage-1 Augmentation aims to train a well-generalized encoder so that the resulting UFE features could be robust against different unseen stream conditions. Both online augmentation (SpecAugment) and offline augmentation approaches are explored. Stage-2 Time Masking applies temporal masks on the UFE features. It provides a simple online augmentation technique to create inter-stream dynamics.

2. Adaptive CTC fusion applies the stream fusion vector to the CTC networks in the decoding step. CTC contributions then change dynamically depending on the HAN component, instead of the previous approach of pre-fixed weights.

4.3.2 Two-Stage Augmentation Scheme

Following the framework of the two-stage training strategy, the proposed two-stage augmentation scheme defines individual steps to simulate single-stream variations and inter-stream dynamics, respectively. Here we define the streams used for Stage-2 training as target streams.

4.3.2.1 Stage 1 Augmentation

The goal of Stage-1 training is to obtain a set of UFE features with more discriminative power for Stage-2 prediction. With limited amount of data for target streams in Stage-2, data augmentation in Stage-1 is a strategy to create more data with a diverse set of conditions and also to involve audio from non-target arrays. In this work, we explore two approaches to improve training with data augmentation in the multi-array scenario:

- SpecAugment is an online augmentation technique that degrades input on the fly in the training mini-matches. It views the spectrogram as a visual representation, and modifies the spectrogram by warping it in the time direction and applying masks in frequency and time.
- The second approach is offline augmentation that generates extra data before training. In the multi-stream framework, we conduct experiments with either simulated audio or real recordings from non-target streams. In the DIRHA dataset, several reverberated versions of clean speech are generated using pre-measured room impulse responses; In the AMI corpus, recordings from close-talk microphones in addition to microphone arrays are used for Stage-1 training.

4.3.2.2 Stage 2 Time Masking

Stage-2 augmentation aims to improve a multi-stream model's robustness against variations in inter-stream dynamics. For instance, the model needs to learn how to reliably handle the situation if one of the arrays suddenly fails in a meeting setting. Since the UFE features are the direct inputs for Stage-2, we consider augmentation on UFE features instead of log-Mel filter bank features.

In this work, we introduce Stage-2 Time Masking, a simple but effective method to create differences across the streams. Inspired by temporal masking in SpecAugment, Stage-2 Time Masking masks the UFE features in time for individual streams. The time mask is utterance-specific, in that it replaces the features with the mean value of the UFE features for that utterance. The applied location and duration of a mask are both randomly chosen from a uniform distribution. The Stage-2 Time Masking is intended to mimic the situation of a partial loss of a speech segment for one of the streams. Compared with augmentation at the acoustic level, Stage-2 Time Masking is computationally easy to apply with no additional data.

4.3.3 Adaptive CTC Fusion

In the Sec. 4.2, the CTC component of each stream is pre-trained in Stage-1 and kept frozen in Stage-2 for training. During inference in a multi-stream setting, equal decoding weights across all streams are assigned to the CTC components. In the beam search, the CTC prefix score $\alpha_{\text{ctc}}(h)$ of hypothesized sequence h is as follows:

$$\alpha_{\text{ctc}}(h) = \frac{1}{N}\sum\nolimits_{i=1}^{N}\alpha_{\text{ctc}^{(i)}}(h). \tag{4.8}$$

These predefined CTC weights could be problematic if one array is in an acoustic condition that is significantly worse than the others.

In this work, we propose adaptive CTC fusion during decoding to mitigate the problem above. For every prediction, the HAN component produces an attention vector $[\beta_l^{(1)}, \beta_l^{(2)}, ..., \beta_l^{(N)}]$ across all streams, which steers the system to more informative streams:

$$\beta_l^{(i)} = \text{HierarchicalAttention}(\mathbf{q}_{l-1}, \mathbf{r}_l^{(i)}), i \in \{1, ..., N\}. \tag{4.9}$$

The Softmax output $\beta_l^{(i)}$ represents a stream-level attention weight for stream i of letter prediction c_l. $\mathbf{q}_{l-1}$ is the previous decoder state and $\mathbf{r}_l^{(i)}$ is the context vector from stream i. Since a label-synchronous beam search is employed during inference, the stream attention vector can be combined with CTC prefix scores $\alpha_{\text{ctc}}(h)$ for a hypothesized sequence h:

$$\alpha_{\text{ctc}}(h) = \textstyle\sum_{i=1}^{N} \beta_l^{(i)} \cdot \alpha_{\text{ctc}^{(i)}}(h), \tag{4.10}$$

where adaptive stream weight $\beta_l^{(\cdot)}$ is applied to each CTC network and l is the index of the latest prediction of hypothesis h.

4.3.4 Data

Two datasets, DIRHA English WSJ and AMI meeting corpus, were used for experiments and analysis. Both of the datasets are discussed in Sec. 4.1.3.

For experiments on DIRHA English WSJ, we included five single microphones (depicted in Fig. 4.4) in addition to two microphone arrays, Beam Linear Array (BLA) and Beam Circular Array (BCA). Moreover, we create a synthetic test stream, *NoMic*, to replicate the scenario of signal cut-off, where inputs are all zeros after mean and variance normalization.

The AMI meeting corpus was created in three instrumented rooms with meeting conversations. Each meeting room is configured with two microphone arrays and close-talk microphones for individual speakers, resulting in 100 hours of far-field

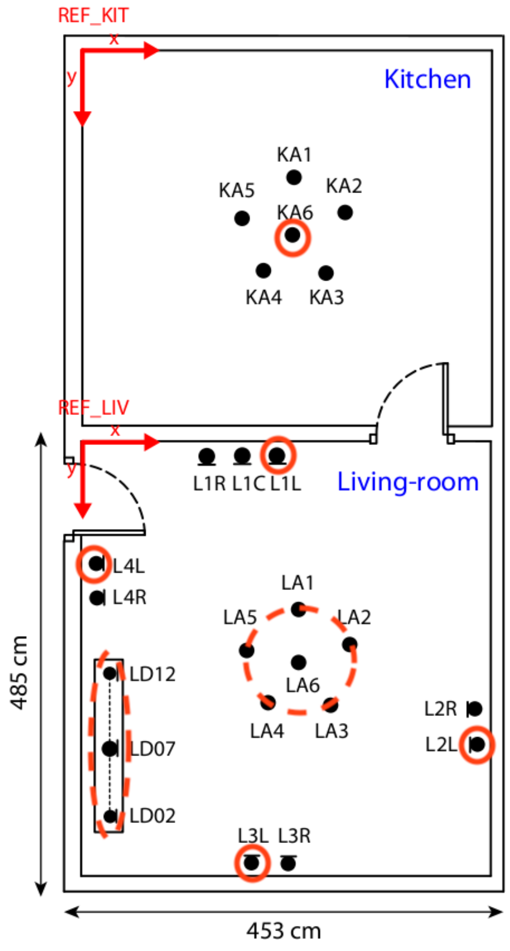


Figure 4.4. DIRHA English WSJ microphone configuration. Streams selected are in red circles. Beam Circular Array contains 6 microphones (LA1-LA6), Beam Linear Array includes 11 microphones (LD02-LD12).

signal-synchronized recordings. Table 4.12 summarizes the stream descriptions used in subsequent experiments. For stream *IHM*, the close-talk microphone with the most energy among all attendees is selected at each time frame. In contrast, stream *IHM0* always takes speech from speaker-0, regardless of if speaker-0 is speaking. Similar to the DIRHA setup, *NoMic* is created to mimic constant microphone dropout.

For each array in both datasets, multi-channel input is synthesized into a single-channel audio using the Delay-and-Sum beamforming technique with the BeamformIt Toolkit.

Table 4.12. AMI meeting corpus stream configuration

Stream	Description
MDM	first microphone array
SMDM	second microphone array
IHM	individual headset microphones
IHM0	individual headset microphones (fixed speaker-0 for each meeting)
NoMic	constant stream dropout (all-zero inputs)

4.3.5 Experiment Setup

All the experiments are conducted using the Pytorch backend on ESPnet. Table 4.13 describes the relevant setup information for the various experiments. Two model configurations are explored: *Config-1* includes two BLSTM layers in the encoder and one LSTM layer in the decoder. A more complex model with *Config-2* has an additional two BLSTM layers and an extra LSTM layer as well. We use 52 distinct labels including 26 English letters and other special tokens, i.e., punctuation and sos/eos. A look-ahead word-level RNN-LM is incorporated during inference. It is trained separately using Stochastic Gradient Descent (SGD) for 20 epochs.

Table 4.13. Experimental Configuration

Feature	
Each Stream	80-dim log-mel filter bank + 3-dim pitch
Number of Streams	2
Model	
Encoder type	VGGBLSTM
Encoder layers	Config-1: 6(VGG)+2(BLSTM)
	Config-2: 6(VGG)+4(BLSTM)
Encoder units	320 cells (BLSTM layers)
Encoder projection	320 cells (BLSTM layers)
Subsampling	4
Frame-level Attention	320-cell Location-based
Stream Attention	320-cell Content-based
Decoder type	LSTM
Decoder layers	1 (Config-1) or 2 (Config-2)
Decoder units	300 cells
Decoder Softmax	52 labels
	(26 English letters+punctuation+sos/eos)
Train and Decode	
Optimizer	AdaDelta
Batch size	30 (Stage-1); 15 (Stage-2)
Training Epoch	30 epochs (patience:3 epochs)
CTC weight λ (train)	0.2
CTC weight λ (decode)	0.3
Label Smoothing	Type: Unigram , Weight: 0.05
Beam size	30
RNN-LM	
Type	Look-ahead Word-level RNN-LM
Size	1-Layer LSTM with 1,000 cells
Vocabulary	65,000
Train data	DIRHA:WSJ0-1+extra WSJ text; AMI:AMI
LM weight γ	DIRHA:1.0; AMI:0.5
Optimizer	Stochastic Gradient Descent
Batch size	300
Training Epoch	20
SpecAugment	
Time mask	#masks: 2; T: 40
Frequency mask	#masks: 2; F: 30

4.3.6 Results and Analysis

4.3.6.1 Stage-1 Augmentation

To investigate the effectiveness of Stage-1 augmentation, we evaluate online and offline augmentation techniques on DIRHA and AMI datasets. Table 4.14 illustrates Stage-1 single-stream results using the proposed augmentation schemes. With each model configuration, substantial WER reductions are reported with SpecAugment, i.e., *D1* v.s. *D3* and *D2* v.s. *D4*. Moreover, the more complex network *Config-2* does not necessarily improve over the smaller model *Config-1* until augmentation is utilized in training (i.e., *D1* outperforms *D2*, but *D4* outperforms all earlier models). We create additional reverberated copies of clean WSJ data using room impulse responses measured for four single microphones, i.e., *L1L*, *L2L*, *L3L* and *L4L*. *D11* achieves better WERs across six streams compared to *D5-D10*. More importantly, *D11*, trained with all six streams, outperforms *D4* on the *BCA* and *BLA* evaluations, showing the value of the additional out-of-set data. From here, *D11* is selected as the Stage-1 model for the remaining DIRHA experiments.

Table 4.15 summarizes Stage-1 augmentation results of AMI in a similar way to Table 4.14. It is clear looking at *A1-A4* that online augmentation (SpecAugment) consistently decreases error rates. Including additional the close-talk stream *IHM*, *D8* shows lower WERs comparing to *D4*. From here, *D8* is utilized for AMI Stage-2 training.

4.3.6.2 Adaptive CTC Fusion

4.3.6.2.1 Issues with Predefined CTC weights

In Sec. 4.2, each CTC network in the multi-stream setting contributes equally during inference. These pre-defined CTC weights could cause performance degradation if

Table 4.14. Stage-1 Augmentation (DIRHA). Model size (2, 1) and (4, 2) represent *Config-1* and *Config-2* in Table 4.13. (WER %)

	Train		Model			Test Data			
ID	Data	SpecAug	Size	BCA	BLA	L1L	L2L	L3L	L4L
D1	BCA+BLA	No	(2,1)	33.9	30.7	–	–	–	–
D2	BCA+BLA	No	(4,2)	34	32	–	–	–	–
D3	BCA+BLA	Yes	(2,1)	27.1	24.4	–	–	–	–
D4	BCA+BLA	Yes	(4,2)	24.9	22.6	–	–	–	–
D5	BCA	Yes	(4,2)	27.1	–	–	–	–	–
D6	BLA	Yes	(4,2)	–	27.7	–	–	–	–
D7	L1L	Yes	(4,2)	–	–	28.3	–	–	–
D8	L2L	Yes	(4,2)	–	–	–	35.4	–	–
D9	L3L	Yes	(4,2)	–	–	–	–	33	–
D10	L4L	Yes	(4,2)	–	–	–	–	–	30.4
D11	All Streams	Yes	(4,2)	**19.8**	**17.2**	**22.6**	**24.1**	**22.6**	**22.6**

Table 4.15. Stage-1 Augmentation (AMI). Model size (2, 1) and (4, 2) represent *Config-1* and *Config-2* in Table 4.13. (WER %)

	Train		Model		Test Data	
ID	Data	SpecAug	Size	MDM	SMDM	IHM
A1	MDM+SMDM	No	(2,1)	56.9	61.7	–
A2	MDM+SMDM	No	(4,2)	53.1	58.3	–
A3	MDM+SMDM	Yes	(2,1)	50.3	54.9	–
A4	MDM+SMDM	Yes	(4,2)	46.1	50.5	–
A5	MDM	Yes	(4,2)	50.5	–	–
A6	SMDM	Yes	(4,2)	–	55.5	–
A7	IHM	Yes	(4,2)	–	–	30.4
A8	All Streams	Yes	(4,2)	**42.8**	**48.1**	**27.6**

one of streams is corrupted. We design simple experiments in DIRHA to illustrate this issue. After Stage-1, we formulate a two-stream model using target streams, *BLA* and *NoMic* for training and testing. Since *BLA* is known to be the only informative source, stage-1 performance of 17.2% for *BLA* is viewed as the best possible result. In Table 4.16, the *Oracle* Stage-2 decoding setup with CTC weights $[1.0; 0.0]$ achieves WER of 17.3%, essentially equivalent to the single-stream performance. However, WER increases to 20.5% when equal weights are applied. The proposed adaptive CTC fusion makes the model more robust with the help of stream attention, reaching Stage-1 performance of 17.2% without any pre-existing knowledge of the relative value of the streams.

Table 4.16. Issues with predefined CTC weights (WER %)

Model	Test
Stage-1: BLA only	
D11 in Table 4.14	17.2
Stage-2: BLA-NoMic	
Pre-defined CTC Weights [1.0; 0.0]	17.3
Pre-defined CTC Weights [0.5; 0.5]	20.5
Adaptive CTC Fusion	**17.2**

4.3.6.2.2 Adaptive CTC Fusion: Matched Condition

To show the influence of adaptive CTC fusion in matched conditions, we conduct experiments with different two-stream acoustic conditions. In each experiment, training and evaluation data are drawn from the same arrays. Results are displayed in Table 4.17. In order to pick diverse conditions in DIRHA, three two-stream configurations are chosen, *BLA-L2L*, *BLA-BCA* and *L3L-L4L*. According to the Stage-1 performance, *BLA* is the most informative single stream. *BCA*/*L2L* are the most similar/different streams to *BLA* in terms of WER. *L3L* and *L4L* result in the same WER of 22.6%. For AMI, all three combinations of the three streams are selected. WER improvements are observed across all six cases in two datasets.

Table 4.17. Adaptive CTC fusion in matched conditions (WER %)

Decoding Strategy	Train/Test Data		
DIRHA	*BLA-L2L*	*BLA-BCA*	*L3L-L4L*
Pre-defined CTC [0.5; 0.5]	17.2	16.5	20.4
Adaptive CTC Fusion	**16.9**	**16.1**	**20.1**
AMI	*MDM-SMDM*	*MDM-IHM*	*SMDM-IHM*
Pre-defined CTC [0.5; 0.5]	42	29.3	29.8
Adaptive CTC Fusion	**41.6**	**28.2**	**28.3**

4.3.6.2.3 Adaptive CTC Fusion: Mismatched Condition

For the following experiments, we designate *BLA-L2L* and *MDM-SMDM* as the training stream configurations for DIRHA and AMI, respectively. In DIRHA, three mismatched test conditions are chosen: *BLA-NoMic* and *BLA-KA6* are the unseen scenarios where one stream (*BLA*) is known to greatly outperform the other. Note that *KA6* (Stage-1 WER: 61%) is a microphone in the kitchen while speakers read in the living room; *L3L-L4L* are the microphones with the same Stage-1 performances. We specify two mismatched conditions for AMI: *MDM-NoMic* and *MDM-IHM0*. Recall *IHM0* (Stage-1 WER: 73.7%) is the close-talk microphone attached to speaker-0. In DIRHA, results in Table 4.18 report moderate improvement except *BLA-NoMic*, which sees a modest decline. Stream *NoMic* is an extreme case and may be too aggressive as an unseen test stream. For AMI, relative WER reductions of 6.5% and 4.8% are shown for the mismatched conditions.

Table 4.18. Adaptive CTC fusion in mismatched conditions (WER %)

Decoding Strategy	Test Data		
DIRHA (BLA-L2L)	*BLA-NoMic*	*BLA-KA6*	*L3L-L4L*
Pre-defined CTC [0.5; 0.5]	**26.9**	21	20.3
Adaptive CTC Fusion	27.1	**20.7**	**20**
AMI (MDM-SMDM)	*MDM-NoMic*	*MDM-IHM0*	–
Pre-defined CTC [0.5; 0.5]	46.1	44	–
Adaptive CTC Fusion	**43.1**	**41.9**	–

4.3.6.3 Stage-2 Time Masking

To demonstrate another potential weakness of the previous MEM-Array system, we design experiments in DIRHA to demonstrate potential performance degradation because of a mismatched test condition, as depicted in Table 4.19. *BLA-L2L* and *BLA-NoMic* are used to train and test two Stage-2 models. While the matched conditions on the diagonal of Table 4.19 exhibit reasonable results, the model trained with *BLA-L2L* is unable to handle the unseen condition *BLA-NoMic*, degrading by nearly 10% absolute WER decrease comparing to Stage-1 *BLA* performance, 17.2%.

Table 4.19. Comparison in matched and mismatched conditions (WER %)

	Test Data	
Model	*BLA-L2L*	*BLA-NoMic*
Stage-2 *BLA-L2L*	16.9	**27.1**
Stage-2 *BLA-NoMic*	17.6	17.2

Stage-2 Time Masking is proposed to target the scenario mentioned above with results in Table 4.20. Two multi-stream models are trained using *BLA-L2L* (DIRHA) and *MDM-SMDM* (AMI), respectively. During training, a time mask is created with the length uniformly sampled from $[0, 10]$ (in frames). Note that 10 frames account for 0.4 second due to subsampling. The mask is applied in a randomly selected position on the UFE features.

We experiment with different numbers of time masks. For DIRHA experiments, a model trained with 3 time masks per stream gives the optimal results. In particular, substantial absolute WER improvement of 9.3% is seen when evaluating *BLA-NoMic*, presumably because it is essentially the situation that stage-2 augmentation is simulating. WER on *BLA-KA6* also decreases while keeping other conditions unchanged or slightly improved. In AMI experiments, Stage-2 augmentation keeps all test conditions under similar performances. It is likely that the AMI model could already handle these unseen conditions properly. For instance, without Stage-2 time masking, *MDM-NoMic* achieves a WER of 43.1% which is close to the Stage-1 *MDM* performance of 42.8%. The number of masks is set to be 3 for DIRHA and 1 for AMI based on matched condition performances.

For comparison, input dropout on the UFE features is implemented. Stage-2 Time Masking constantly obtains lower WERs in all conditions, which support the idea of creating stream dynamics instead of unit dropout over the inputs.

4.3.6.4 Discussion on Amount of Parallel Data

Generally, parallel data is more expensive to collect. In this section, we examine how much parallel data could be sufficient for Stage-2 model training with a reasonable performance. We use *BLA-L2L* in DIRHA for this demonstration. As described in Table 4.21, 1 hour data per stream could maintain fair WERs with only an average of 5.3% performance degradation, indicating a relatively low burden for data resources.

Table 4.20. Stage-2 Time Masking (WER %)

Model	Test Data			
DIRHA	*BLA-L2L*	*BLA-NoMic*	*BLA-KA6*	*L3L-L4L*
Stage2 BLA-L2L	16.9	27.1	20.7	20
- Input Dropout 0.2	17.7	38.1	22.1	20.6
- Input Dropout 0.5	19.2	21	23.6	22.6
- Time Masking (#mask=1)	17	18	19.3	20.1
- Time Masking (#mask=2)	16.9	18.2	19.4	20
- Time Masking (#mask=3)	**16.6**	**17.8**	**19.2**	**20**
AMI	*MDM-SMDM*	*MDM-NoMic*	*MDM-IHM0*	–
Stage2 MDM-SMDM	41.6	43.1	41.9	–
- Input Dropout 0.2	42.3	44.5	42.6	–
- Input Dropout 0.5	45.2	49.5	46.3	–
- Time Masking (#mask=1)	**41.6**	**43.1**	**41.6**	–
- Time Masking (#mask=2)	41.6	43	41.9	–
- Time Masking (#mask=3)	41.7	43	41.8	–

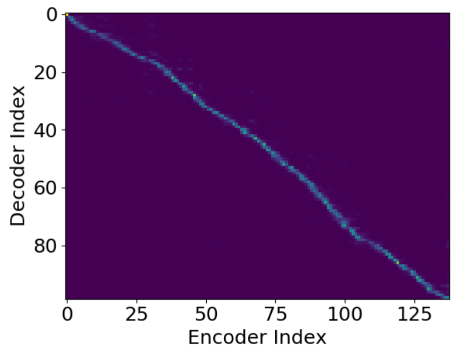
Table 4.21. Discussion on amount of parallel data (WER %)

Training Data	Test Data			
(Hours)	*BLA-L2L*	*BLA-NoMic*	*BLA-KA6*	*L3L-L4L*
0.1	17.1	24	21.3	20.4
1	17	20.4	20	20.1
10	16.7	18.1	18.9	20.2
81 (All)	16.6	17.8	19.2	20.1

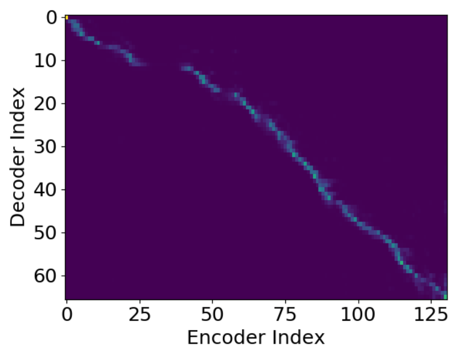
4.3.6.5 Overall

Table 4.22 summarizes the contributions of each proposed step, in this case using *BLA-L2L* and *MDM-SMDM* as the training stream configurations for DIRHA and AMI, respectively, while test data includes matched and mismatched conditions. Stage-1 augmentation together with a more complex model consistently reduces the WERs. Adaptive CTC fusion and Stage-2 time masking provide notable improvements in various scenarios. Overall, compared to the previous training strategy in Sec. 4.2, we observe average relative WER reductions of 45.2% (DIRHA) and 30.7% (AMI). In particular, substantial relative WER improvement of $29.7-59.3\%$ is reported across several mismatched stream conditions. For fair comparison, we also evaluate the model where the HAN component is replaced by fixed stream fusion weights $[0.5; 0.5]$ for fusion of context vectors. In these cases, the components, including CTC, frame-level attention and decoder, are optimized during Stage-2. Our proposed model greatly outperforms the model with no stream attention.

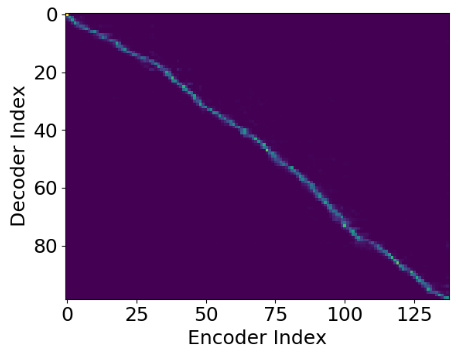
To visualize the effect of the stream attention, Fig. 4.5 shows attention plots of two examples from the evaluation set *MDM-IHM0* in AMI. In the first example, (a)-(c), speaker-0 was speaking, and as a result both *MDM* and *IHM0* were informative sources, and the stream attention in (c) gave weights to both inputs, though shifted slightly towards *IHM0* since this close-talk stream had better speech quality. In the second example, (d)-(f), speaker-0 was not speaking and so another speaker's audio was recorded by *MDM* while *IHM0* could barely capture any speech. In this case, the stream fusion mechanism correctly attends to *MDM* with nearly 100% confidence.



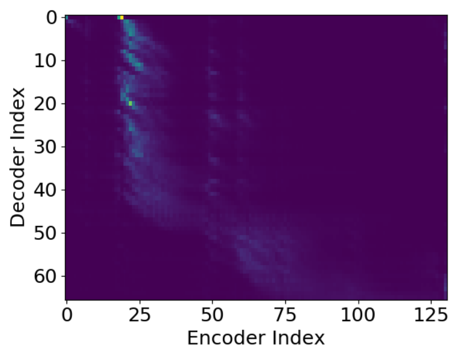
(a) MDM



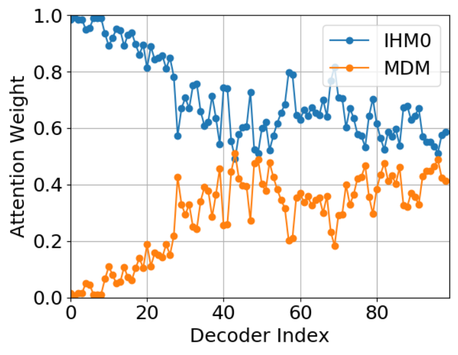
(d) MDM



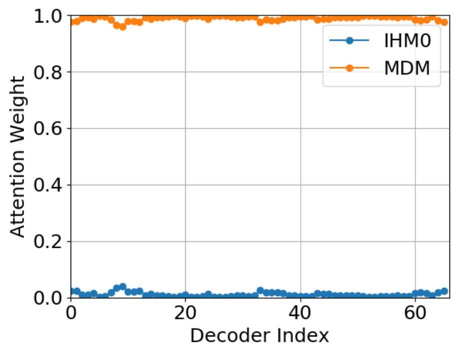
(b) IHM0



(e) IHM0



(c) MDM-IHM0



(f) MDM-IHM0

Figure 4.5. Sentence analysis of attention mechanism during inference. Example 1 (speaker-0 speaking) includes (a),(b),(c); Example 2 (speaker-0 not speaking) includes (d),(e),(f). (a) and (d) are frame-wise attention alignments of *MDM*; (b) and (e) are frame-wise attention alignments of *IHM0*; (c) and (f) are stream attention weights of *MDM-IHM0*.

Table 4.22. Overall results (WER %)

Model	Test Data			
DIRHA (BLA-L2L)	*BLA-L2L*	*BLA-NoMic*	*BLA-KA6*	*L3L-L4L*
Two-Stage Training	27.4	43.8	37.9	29.7
+ Large Model	28	57.4	37.8	29.7
+ Stage-1 Augment.	17.2	26.9	21	20.3
+ Adaptive CTC Fusion	16.9	27.1	20.7	20
+ Stage-2 Time Masking	**16.6(39.4%)**	**17.8(59.3%)**	**19.2(49.3%)**	**20(32.7%)**
No Stream Attention	36.2	66.5	49.4	37.2
AMI (MDM-SMDM)	*MDM-SMDM*	*MDM-NoMic*	*MDM-IHM0*	–
Two-Stage Training	55.5	69	59.2	–
+ Large Model	52	62	55.1	–
+ Stage-1 Augment.	42	46.1	44	–
+ Adaptive CTC Fusion	41.6	43.1	41.9	–
+ Stage-2 Time Masking	**41.6(25.0%)**	**43.1(37.5%)**	**41.6(29.7%)**	–
No Stream Attention	56	69.7	65.8	–

4.3.7 Conclusion

In this work, we present a two-stage augmentation scheme and adaptive CTC fusion for the purpose of improving robustness of the multi-stream end-to-end model against diverse testing conditions. Inherited from the two-stage training strategy, the two-stage augmentation consistently improves performance across matched and mismatched conditions; adaptive CTC fusion enhances the robustness by applying stream attention weights dynamically. Experiments have been conducted on two datasets, DIRHA and AMI, as a multi-stream scenario. Compared with the previous training strategy, substantial improvements are reported with relative WER reductions of $29.7 - 59.3\%$ across several unseen stream combinations.

Chapter 5

Multi-Encoder Multi-Resolution (MEM-Res)

5.1 Introduction

In end-to-end ASR approaches, the encoder acts as an acoustic model providing higher-level features for decoding. BLSTM has been widely used due to its ability to model temporal sequences and their long-term dependencies as the encoder architecture; On the other hand, deep CNN was introduced to model spectral local correlations and reduce spectral variations in E2E framework. The encoder architecture combining CNN with recurrent layers, was suggested to address the limitation of LSTM. While temporal subsampling in RNN and max-pooling in CNN aim to reduce the computational complexity and enhance the robustness, it is likely that subsampling technique results in loss of temporal resolution.

As one realization of the multi-stream E2E model in Chapter 3, we propose a Multi-Encoder Multi-Resolution (MEM-Res) model within the joint CTC/ATT framework. Two heterogeneous encoders, RNN-based and CNN-RNN-based, with different architectures, temporal resolutions and separate CTC networks work in parallel to extract complimentary acoustic information. On top of the regular attention networks, the HAN component is introduced to steer the decoder toward the more informative encoder. The encoder that carries the most discriminate information for the prediction can dynamically receive a stronger weight. Each encoder is associated with a CTC network to guide the frame-wise alignment process for individual encoders.

5.2 MEM-Res Model

The overall architecture is shown in Fig. 5.1. Two types of encoders with different temporal resolutions are presented in parallel to capture acoustic information in various ways, followed by an attention fusion mechanism together with per-encoder CTC. An external RNN-LM is also involved during the inference step. Following the same notation in Chapter 3, the MEM-Res model maps from a T-length speech feature sequence, $X = \{\mathbf{x}_t \in \mathbb{R}^D | t = 1, 2, ..., T\}$, to an L-length letter sequence, $C = \{c_l \in \mathcal{U} | l = 1, 2, ..., L\}$.

5.2.1 Multi-Encoder with Multi-Resolution

We propose a Multi-Encoder Multi-Resolution (MEM-Res) architecture that has two encoders, RNN-based and CNN-RNN-based. Both encoders take the same input features in parallel operating on different temporal resolutions, aiming to capture complimentary information in the speech.

The RNN-based BLSTM encoder is designed to model temporal sequences with their long-range dependencies. Subsampling in BLSTM is often used to decrease the computational cost, but performing subsampling might result in lost information which could be better modeled in RNN. In MEM-Res, the RNN-based encoder has stacked BLSTM layers that extract frame-wise hidden vectors $H^{(1)} = \{\mathbf{h}_1^{(1)}, ..., \mathbf{h}_T^{(1)}\}$

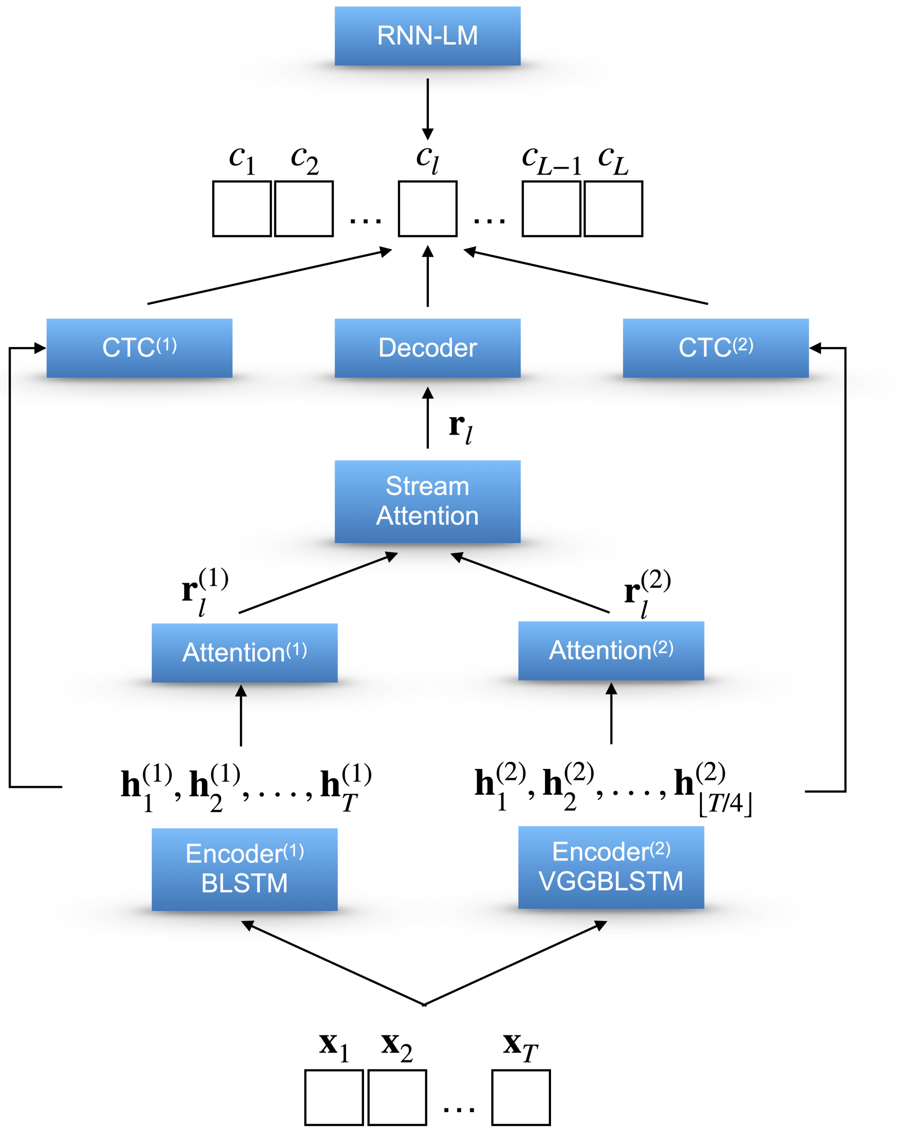


Figure 5.1. The Multi-Encoder Multi-Resolution (MEM-Res) architecture

without subsampling in any layer:

$$H^{(1)} = \text{Encoder}^{(1)}(X) \triangleq \text{BLSTM}(X), \tag{5.1}$$

where the BLSTM encoder is referred to as stream 1.

The combination of CNN and RNN allows the convolutional feature extractor applied on the input to reveal local correlations in both time and frequency dimensions. The RNN block on top of CNN makes it easier to learn temporal structure from the CNN output, to avoid modeling direct speech features with more underlying variations. The max-pooling layer is essential in CNN to reduce the spatial size of the representation to control over-fitting. In MEM-Res, we use the initial layers of the VGG net architecture followed by stacked BLSTM layers as VGGBLSTM decoder labeled as stream 2:

$$H^{(2)} = \text{Encoder}^{(2)}(X) \triangleq \text{VGGBLSTM}(X), \tag{5.2}$$

where $H^{(2)} = \{\mathbf{h}_1^{(2)}, ..., \mathbf{h}_{\lfloor T/4 \rfloor}^{(2)}\}$. The configuration of convolutional layers in VGG-BLSTM encoder is described in Table 4.2. Two max-pooling layers with $stride = 2$ downsample the input features by a factor of $s = 4$ in both temporal and spectral directions.

5.2.2 Hierarchical Attention

Since the encoders in MEM-Res describe the speech signal differently by catching acoustic knowledge in their own ways, encoder-level fusion is suitable to boost the network's ability to retrieve the relevant information. Following the general formation of multi-stream end-to-end ASR, the HAN component is used for information fusion. The decoder with the HAN component is trained to selectively attend to the appropriate encoder.

The letter-wise context vectors, $\mathbf{r}_l^1$ and $\mathbf{r}_l^2$, from individual encoders are computed as follows:

$$\mathbf{r}_l^{(1)} = \textstyle\sum_{t=1}^{T} a_{lt}^{(1)} \mathbf{h}_t^{(1)}, \tag{5.3}$$

$$\mathbf{r}_l^{(2)} = \textstyle\sum_{t=1}^{\lfloor T/4 \rfloor} a_{lt}^{(2)} \mathbf{h}_t^{(2)}, \tag{5.4}$$

where the attention weights are obtained using a content-based attention mechanism, as mentioned in Sec. 2.1.2.2. Note that since Encoder$^{(2)}$ (VGGBLSTM) performs downsampling by 4, the summation is till $\lfloor T/4 \rfloor$ in Eq. 5.4. The fusion context vector $\mathbf{r}_l$ is obtained as a convex combination of $\mathbf{r}_l^1$ and $\mathbf{r}_l^2$ as illustrated in the following:

$$\mathbf{r}_l = \beta_l^{(1)} \mathbf{r}_l^{(1)} + \beta_l^{(2)} \mathbf{r}_l^{(2)}, \tag{5.5}$$

The stream-level attention weights $\beta_l^{(1)}$ and $\beta_l^{(2)}$ are estimated using a content-based attention mechanism.

5.2.3 Per-encoder CTC

In the CTC/ATT model with a single encoder, the CTC objective serves as an auxiliary task to speed up the procedure of realizing monotonic alignment and providing a sequence-level objective. Similar to the general multi-stream framework, we introduce per-encoder CTC where a separate CTC mechanism is active for each encoder stream during training and decoding. In the case that both encoders are with different temporal resolutions and network architectures, per-encoder CTC can further align speech with labels in a monotonic order and customize the sequence modeling of individual streams.

During multi-task training and joint decoding, the objective functions are the same as the multi-stream model proposed in Chapter 3, with $N = 2$. Equal weight is assigned to each CTC component.

5.3 Data

We demonstrate our proposed MEM-Res model using two datasets: WSJ [98] (81 hours) and CHiME-4 [99] (18 hours). WSJ is a clean speech corpora including 81 hours of transcribed speech. In WSJ, we use the standard configuration: “si284” for training, “dev93” for validation, and “eval92” for test. The CHiME-4 dataset is a noisy speech corpus recorded or simulated using a tablet equipped with 6 microphones in four noisy environments: a cafe, a street junction, public transport, and a pedestrian area. For training, we use both “tr05_real” and “tr05_simu” with additional WSJ corpora to support end-to-end training. “dt05_multi_isolated_1ch_track” is used for validation. We evaluate the real recordings with 1, 2, 6-channel in the evaluation set. The BeamformIt method is applied to multi-channel evaluation.

5.4 Experiment Setup

Table 5.1 summarizes model configurations and training/decoding parameters used in the experiments. In all experiments, 80-dimensional mel-scale filterbank coefficients with additional 3-dimensional pitch features serve as the input features.

The Encoder$^{(1)}$ contains four BLSTM layers, in which each layer has 320 cells in both directions followed by a 320-unit linear projection layer. The Encoder$^{(2)}$ combines the convolution layers with an RNN-based network that has the same architecture as Encoder$^{(1)}$. A content-based attention mechanism with 320 attention units is used in encoder-level and frame-level attention mechanisms. The decoder is a one-layer unidirectional LSTM with 300 cells. We use 52 distinct labels including 26 English letters and other special tokens, i.e., punctuations and sos/eos.

We incorporate the look-ahead word-level RNN-LM of 1-layer LSTM with 1000 cells and 65K vocabulary, that is, 65K-dimensional output in Softmax layer. In addition to the original speech transcription, the WSJ text data with 37 million words from 1.6 million sentences is supplied as training data. RNN-LM is trained separately using Stochastic Gradient Descent (SGD) with learning rate = 0.5 for 60 epochs. Note that the RNN-LM is shared across two datasets.

The MEM-Res model is implemented using the Pytorch backend on ESPnet. Training procedure is operated using the AdaDelta algorithm with gradient clipping on single GPUs, "GTX 1080ti". The mini-batch size is set to be 15. We also apply a unigram label smoothing technique to avoid over-confidence predictions. The beam width is set to 30 for WSJ and 20 for CHiME-4 in decoding. For models jointly trained with CTC and attention objectives, $\lambda = 0.2$ is used for training, and $\lambda = 0.3$ for decoding. RNN-LM scaling factor γ is 1.0 for all experiments with the exception of using $\gamma = 0.1$ in decoding attention-only models.

5.5 Results and Analysis

5.5.1 Overall Results

The overall experimental results on WSJ and CHiME-4 are shown in Table 5.2. Compared to joint CTC/ATT single-encoder models, the proposed MEM-Res model with per-encoder CTC and HAN achieves relative improvements of 9.6% (29.2% $\rightarrow$ 26.4%) in CHiME-4 and 21.7% in WSJ (4.6% $\rightarrow$ 3.6%) in terms of WER. We compare the MEM-Res model with other end-to-end approaches, and it outperforms all of the systems from previous studies. We also conduct experiments using ROVER technique to fuse two single-encoder models in the word level, and our proposed models show substantial improvements. We design experiments with fixed encoder-level attention $\beta_l^1 = \beta_l^{(2)} = 0.5$. The MEM-Res model with the HAN component outperforms the

Table 5.1. Experimental configuration

Feature	
Each Stream	80-dim log-mel filter bank + 3-dim pitch
Number of Streams	2
Model	
Encoder type	*BLSTM* or *VGGBLSTM*
Encoder layers	*VGGBLSTM*: 6(VGG)+4(BLSTM)
	BLSTM: 4(BLSTM)
Encoder units	320 cells (BLSTM layers)
Subsampling	*BLSTM*: 1
	VGGBLSTM: 4
Frame-level Attention	320-cell Content-based
Stream Attention	320-cell Content-based
Decoder type	LSTM
Decoder layers	1
Decoder units	300 cells
Decoder Softmax	52 labels
	(26 English letters+punctuation+sos/eos)
Train and Decode	
Optimizer	AdaDelta
Batch size	15
Training Epoch	15 epochs
CTC weight λ (train)	0.2
CTC weight λ (decode)	0.3
Label Smoothing	Type: Unigram, Weight: 0.05
Beam size	WSJ:30; CHiME-4: 20
RNN-LM	
Type	Look-ahead Word-level RNN-LM
Size	1-Layer LSTM with 1,000 cells
Vocabulary	65,000
Train data	WSJ0-1+extra WSJ text
LM weight γ	0.5
Optimizer	Stochastic Gradient Descent
Batch size	300
Training Epoch	60
Learning Rate	0.5

ones without parameterized stream attention. Moreover, per-encoder CTC constantly enhances the performance with or without the HAN component. Especially in WSJ, the model shows a notable decrease (4.3% $\rightarrow$ 3.6%) in WER with per-encoder CTC. Our results further confirm the effectiveness of joint CTC/ATT architecture in comparison to models with either CTC or attention networks.

5.5.2 MEM-Res Model v.s. Single-Stream Model

For fair comparison, we increase the number of BLSTM layers from 4 to 8 in Encoder$^{(2)}$ to train a single-encoder model. In Table 5.3, the MEM-Res system outperforms the single-encoder model by a significant margin with similar amounts of parameters, 21.9 million versus 21.3 million. In CHiME-4, we evaluate the model using real test data from 1, 2, 6-channel resulting in an average of 19% relative improvement from all three setups. In WSJ, we achieved 3.6% WER in eval92 in our MEM-Res framework with relatively 32.1% improvement.

5.5.3 Analysis of Hierarchical Attention Mechanism

As shown in Table 5.4, We analyze the average stream-level attention weight for Encoder$^{(2)}$ when we gradually decrease the number of LSTM layers while keeping Encoder$^{(1)}$ with the original configuration. It aims to show that the HAN component is able to attend to the appropriate encoder seeking for the right knowledge. As suggested in the table, more attention goes to Encoder$^{(1)}$ from Encoder$^{(2)}$ as we intentionally make Encoder$^{(2)}$ weaker.

5.5.4 Effect of Multi-Resolution Configuration

Since the convolution layers of the VGGBLSTM encoder downsamples the input features by a factor of 4, there is no subsampling in the recurrent layers of Encoder$^{(2)}$.

Table 5.2. Comparison among single-encoder end-to-end models with BLSTM or VGGBSLTM as the encoder, the MEM-Res model and prior end-to-end models. (WER %)

Model	CHiME-4 et05_real_1ch	WSJ eval92
BLSTM (Single-Encoder)		
CTC	62.7	36.4
ATT	50.2	20.8
CTC+ATT	29.2	4.6
VGGBLSTM (Single-Encoder)		
CTC	50.6	19.1
ATT	42.2	17.2
CTC+ATT	29.6	5.6
BLSTM+VGGBLSTM (ROVER)		
CTC+ATT	30.8	5.9
BLSTM+VGGBLSTM (MEM-Res)		
CTC	49.1	15.2
ATT	44.3	18.9
CTC(shared)+ATT	26.8	4.4
CTC(shared)+ATT+HAN	26.9	4.3
CTC(per-enc)+ATT	26.6	4.1
CTC(per-enc)+ATT+HAN	**26.4**	**3.6**
Previous Studies		
RNN-CTC [6]	-	8.2
Eesen [7]	-	7.4
Temporal LS + Cov. [100]	-	6.7
E2E+regularization[101]	-	6.3
Scatt+pre-emp[102]	-	5.7
Joint e2e+look-ahead LM[24]	-	5.1
RCNN+BLSTM+CLDNN [75]	-	4.3
EE-LF-MMI [103]	-	4.1

Table 5.3. Comparison between the MEM-Res model and VGGBSLTM single-encoder model with similar network size (WER %)

Data	Single-Encoder (21.9M)	Proposed Model (21.3M)
CHiME-4		
et05_real_1ch	32.2	**26.4 (18.0%)**
et05_real_2ch	26.8	**21.9 (18.3%)**
et05_real_6ch	21.7	**17.2 (20.8%)**
WSJ		
eval92	5.3	**3.6 (32.1%)**

Table 5.4. Analysis of hierarchical attention mechanism when fixing $\mathrm{Encoder}^{(1)}$ and changing the number of LSTM layers in $\mathrm{Encoder}^{(2)}$ (WER %: CHiME-4)

# LSTM Layers in VGGBLSTM	Average Stream Attention for VGGBLSTM	WER %
0	0.27	30.6
1	0.52	29.8
2	0.75	28.9
3	0.82	27.8
4	0.81	26.4

Therefore, we experiment the effect of multi-resolution by changing the subsampling rate in the BLSTM encoder. The results in Table 5.5 show the contribution of multiple resolutions. When $s^{(1)} = 2$, we subsample every other output feature from the first BLSTM layer. In addition, the same subsampling strategy is operated on the second BLSTM layer as well for the case of $s^{(1)} = 4$. The WER goes up when increasing subsampling factor $s^{(1)}$ closer to $s^{(2)} = 4$ in both datasets. In other words, the fusion works better when two encoders are more heterogeneous which supports our hypothesis.

Table 5.5. Effect of multi-resolution configuration $(s^{(1)}, s^{(2)})$. $s^{(1)}$ and $s^{(2)}$ are subsampling factors for $\text{Encoder}^{(1)}$ and $\text{Encoder}^{(2)}$. (WER %)

Data	(4,4)	(2,4)	(1,4)
CHiME-4			
et05_real_1ch	29.1	27.0	**26.4**
WSJ			
eval92	4.5	4.2	**3.6**

5.6 Conclusion

In this work, we present our MEM-Res framework to build an end-to-end ASR system. Higher-level frame-wise acoustic features are carried out from RNN-based and CNN-RNN-based encoders with subsampling only in convolutional layers. Stream fusion selectively attends to each encoder via a content-based attention. We observe that assigning a CTC network to an individual encoder further enhances the heterogeneous configuration of encoders. To demonstrate the effectiveness of the proposed model, experiments are conducted on WSJ and CHiME-4, resulting in a relative WER reduction of $18.0 - 32.1\%$. Moreover, the proposed MEM-Res model achieves 3.6%

WER in the WSJ eval92 test set.

Chapter 6

Performance Monitoring (PM) for End-to-End Model

6.1 Introduction

In recent years, significant improvement for conventional ASR has been achieved via advancements with deep neural networks. The main paradigm for an ASR system is the so-called hybrid approach, which involves training a DNN to predict context dependent phoneme states (or senones) from the acoustic features. However, if test data comes from a very different domain than DNN training data, it is possible for the recognizer to fail without any warning, e.g., confident prediction of incorrect labels. Predicting these failures is the goal of the work that follows. Humans are often aware of the uncertainty of decisions they are making [104]. Performance Monitoring (PM) techniques aim for the same goal - to determine the quality of a system's output - based only on the behavior of the system and without any knowledge of the underlying truth. In conventional ASR systems, several PM techniques have been well-developed to monitor performance by looking at tri-phone posteriors or pre-softmax activations from neural network acoustic modeling. An effective PM measure could be useful in a number of applications, such as multi-stream selection scenario [55, 65, 68, 74, 105, 106] or semi-supervised training [107].

Unlike conventional ASR, end-to-end speech recognition approaches are designed to directly output word or character sequences from the input audio. This model subsumes several disjoint components in the hybrid ASR model (acoustic model, pronunciation model, language model) into a single neural network. As a result, all the components of an end-to-end model can be trained jointly to optimize a single objective. Three dominant E2E architectures for ASR are Connectionist Temporal Classification, attention-based encoder-decoder models and recurrent neural network transducers. A joint CTC/ATT framework is proposed to take advantage of both architectures within a multi-task scheme. Unlike hybrid ASR systems, strategies for monitoring more recently developed end-to-end ASR systems have not yet been explored, and that is the focus of this Chapter.

In the previous chapters, intelligence of each stream is examined using a hierarchical attention network in a multi-stream scenario. In this chapter, we propose several performance monitoring measures to evaluate the reliability of transcriptions from an end-to-end ASR model. In the hybrid approach, tri-phone posterior distributions and their corresponding pre-softmax activations are typically treated as PM features. Averaged entropy over temporal frames was proposed as a confidence measure in stream-selection [66, 105]. Mean temporal distance on posteriors estimates the performance by capturing the divergence of any two frames over several time spans [70, 108]. Reconstruction error of an auto-encoder trained on pre-softmax features was also used as the selection criterion in a multi-stream system [59, 64].

In the end-to-end setting, there are two levels of probability distributions: attention weights and decoder posteriors. Instead of temporal posteriors in the conventional case, each probability distribution corresponds to a character-level prediction. Therefore, we must adapt the techniques used for hybrid systems to the joint CTC/ATT model. Moreover, inspired by the success of discriminatively-trained DNNs, we propose using a RNN regression model trained to directly predict performance. Our analyses demonstrate strong correlations between PM measures and true performance, indicating that end-to-end ASR systems are indeed amenable to effective monitoring.

We present four different PM measures: Entropy, Mean Character Distance (MCD), Auto-Encoder, RNN-Predictor to assess the credibility of predictions from an joint CTC/ATT model. Three types of features are exploited to predict the CERs: attention distributions, decoder posterior distributions, and their pre-softmax activations.

6.2 Data

We conduct all our experiments based on the WSJ corpus and its variants with additional noises or reverberation conditions. Table 6.1 summarizes various databases that are used in our experiments. For end-to-end ASR, the clean WSJ SI-284 corpus and Aurora4 multi-condition training data are used in multi-style training. The Aurora4 set [109] contains simulated recordings of WSJ utterances in 14 different acoustic conditions, varying noise types and channel conditions. WSJ dev93 and Aurora4 dev set serve as the validation set for ASR training. We also use the same data configuration to train the auto-encoder. For the RNN predictor, in order to see data with a reasonable balance across different CERs, we use clean WSJ SI-84 together with its two artificially noise-corrupted versions, Aurora4 and CHiME-4-Sim. CHiME-4-Sim [99] consists of single-channel WSJ data with four additive noises.

To evaluate the predictability of each PM measure, a linear regression model is applied to map between PM scores and truth performance. The regression model is computed according to development sets from WSJ, Aurora4 and CHiME-4-Sim.

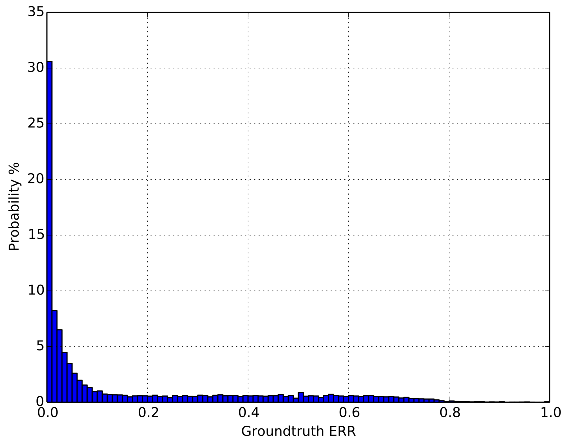
We refer to the data to train the ASR and linear regression model as *Train* and *Dev*, respectively. Fig. 6.1(a) and Fig. 6.1(b) show histograms of utterance CERs for both sets. Performance of *Dev* is more widely spread out than *Train*. We test the effectiveness of PM measures with data from two different domains. *Seen* test domains include evaluation sets from WSJ, Aurora4 and CHiME-4-Sim which are drawn from the same domains as ASR and PM training; *Unseen* test domains consist of evaluation sets from CHiME-4-Real [99] (real noisy recordings), Reverb-Sim [110] (simulated reverberation), DIRHA-Sim [87] (simulated reverberation) and DIRHA-Real [87] (real reverberated recordings). All the test data together are referenced as *Test*, with utterance CERs shown in Fig. 6.1(c).

Table 6.1. Datasets for experiments with CER (%) of test set

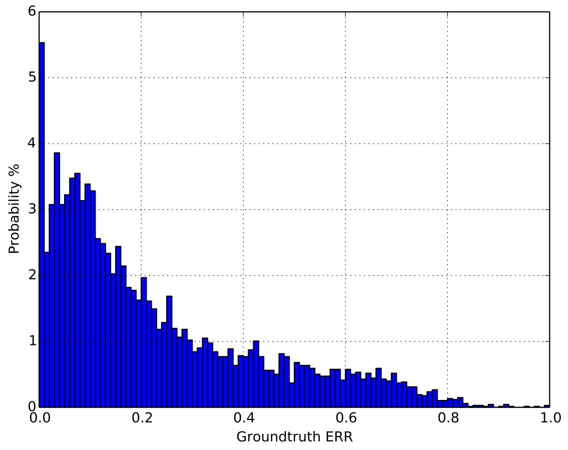
Task	Dataset	
Train		
ASR	WSJ(SI-284), Aurora4	
AE	WSJ(SI-284), Aurora4	
RNN	WSJ(SI-84), Aurora4, CHiME-4-Sim	
Dev		
Linear Regr	WSJ, Aurora4, CHiME-4-Sim	
Test		
All Tasks	WSJ (5.7%)	Aurora4 (14.5%)
	CHiME-4-Sim (52.2%)	DIRHA-Sim (68.2%)
	CHiME-4-Real (59.0%)	DIRHA-Real (70.7%)
	Reverb (41.6%)	

6.3 Experiment Setup

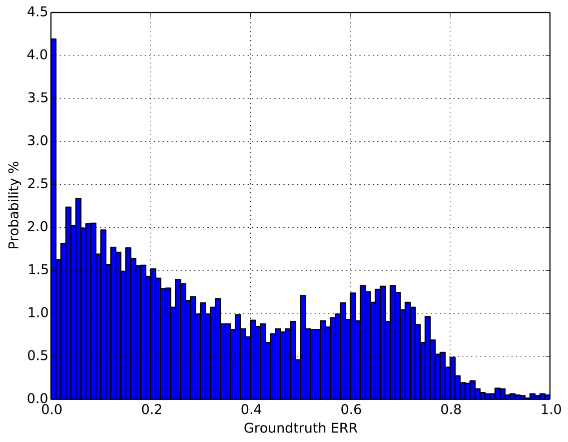
In the end-to-end model, the encoder contains four BLSTM layers, each with 320 cells in both directions, followed by a 320-unit linear projection layer. A content-based attention mechanism with 320 attention units follows. The decoder is a one-layer



(a) Train



(b) Dev



(c) Test

Figure 6.1. Histogram of CERs (%) in various datasets

unidirectional LSTM with 300 cells. We use 52 distinct labels at the decoder softmax layer, including 26 English letters and additional special tokens (i.e., punctuations and sos/eos).

The model is optimized using the AdaDelta algorithm with a mini-batch size of 15. We also apply a unigram label smoothing technique to avoid over-confident predictions. The beam width is set to 30 for all results. For joint training with CTC and attention objectives, $\lambda = 0.2$ is used for training, and $\lambda = 0.3$ for decoding. All results are reported as CER. In all experiments, 80-dimensional mel-scale filter-bank coefficients with additional 3-dimensional pitch features serve as the input features. Attention distributions, decoder posteriors, and pre-softmax features are extracted during joint decoding. Decoding results of each *Test* set are shown in Table. 6.1. No external language model is involved in this work. Model configurations are described in Table 6.2.

6.4 Performance Measures

6.4.1 Entropy

In the hybrid ASR framework, it was observed [66, 105] that the discriminative power of a clean phoneme classifier decreases for input speech with reduced signal-to-noise ratios. As the phoneme posteriors tend to be more uniformly distributed, entropy was proposed as a measure of uncertainty. In end-to-end ASR, which has no phoneme distributions, we investigate if entropy on either the attention probabilities or the decoder posteriors could be a reasonable indicator of model performance.

We denote a K-dimensional probability distribution vector associated with prediction c_l as $\mathbf{p}_l = [p_l^{(1)}, p_l^{(2)}, ..., p_l^{(K)}]$. $\mathbf{p}_l$ is either a frame-level attention probability vector $\mathbf{a}_l = [a_l^{(1)}, a_l^{(2)}, ..., a_l^{(T)}]$ or a decoder posterior distribution $\mathbf{o}_l = [o_l^{(1)}, o_l^{(2)}, ..., o_l^{(S)}]$.

Table 6.2. Experimental configuration

Feature	
Each Stream	80-dim log-mel filter bank + 3-dim pitch
Number of Streams	1
Model	
Encoder type	BLSTM
Encoder layers	4
Encoder units	320 cells
Encoder projection	320 cells
Subsampling	4
Frame-level Attention	320-cell Content-based
Decoder type	LSTM
Decoder layers	1
Decoder units	300 cells
Decoder Softmax	52 labels
	(26 English letters+punctuation+sos/eos)
Train and Decode	
Optimizer	AdaDelta
Batch size	15
Training Epoch	30 epochs (patience:3 epochs)
CTC weight λ (train)	0.2
CTC weight λ (decode)	0.3
Label Smoothing	Type: Unigram, Weight: 0.05
Beam size	30

Note that T is the number of input frames at the attention level. S is the Softmax dimension at the decoder output.

Entropy is first computed on the character-level distribution $\mathbf{p}_l$.

$$Entropy(\mathbf{p}_l) = -\sum_{k=0}^{K} p_l^{(k)} \log p_l^{(k)}, \tag{6.1}$$

where $\mathbf{p}_l$ could be $\mathbf{a}_l$ or $\mathbf{o}_l$. The utterance-level score is obtained by averaging entropy scores over all predictions.

$$E_{score} = \frac{1}{L}\sum_{l=0}^{L} Entropy(\mathbf{p}_l), \tag{6.2}$$

where L is the number of predictions. Note that the dimension of the attention distribution is equal to the number of time frames T at encoder output, which varies per utterance. Hence, we normalize the entropy by its upper bound $\log(T)$ for a consistent range $[0, 1]$.

6.4.2 M-Measure: Mean Character Distance

The Mean Temporal Distance (MTD) or M-Measure was proposed to show the mean distance of pairwise probability distributions from DNN outputs [70]. Symmetric Kullback–Leibler divergence was selected as a distance metric for distributions $\mathbf{p} = [p^{(1)}, p^{(2)}, ..., p^{(N)}]$ and $\mathbf{q} = [q^{(1)}, q^{(2)}, ..., q^{(N)}]$, which are each posteriors from different time frames.

$$\mathcal{D}(\mathbf{p}, \mathbf{q}) = \sum_{k=0}^{K} p^{(k)} \log \frac{p^{(k)}}{q^{(k)}} + \sum_{k=0}^{K} q^{(k)} \log \frac{q^{(k)}}{p^{(k)}} \tag{6.3}$$

A high MTD score indicates a greater difference between $\mathbf{p}$ and $\mathbf{q}$, meaning the model is choosing different output classes at different times. In noisy conditions or other cases with low model confidence, the distributions at different times should be more similar. In MTD, M-Measure often needs to sample frame pairs more than 200 ms apart due to phonetic coarticulation; for shorter time spans, small divergence could be caused by high confidence in the same phoneme.

In end-to-end framework, we propose Mean Character Distance (MCD), adapted from mean temporal distance. Since each probability estimate $\mathbf{p}_l$ in the attention or decoder posterior corresponds to a character prediction, the distance measure is suitable even for adjacent frames without concern for a co-articulation effect. So, we take the mean of distance over all pairs from various windows $\{\triangle l\} = \{1, 2, 3, 4, 5\}$.

$$\mathcal{M}_{score} = \frac{\sum_{\{\triangle l\}} \sum_{l=\triangle l}^{L} \mathcal{D}(\mathbf{p}_{l-\triangle l}, \mathbf{p}_l)}{\prod_{\{\triangle l\}} (L - \triangle l)} \tag{6.4}$$

Equal weights are applied to all pairs, instead of assigning higher weights for more distant pairs, as with MTD.

6.4.3 Auto-Encoder

Mean squared error (MSE) of auto-encoder outputs was proposed in [59] to measure the mismatch between train and test data as an indicator of DNN performance. The auto-encoder is trained to minimize the reconstruction error of the DNN pre-softmax activations from training data. The previous study illustrated that if a data vector is sampled from training data distribution, the corresponding reconstruction error should be low, while a high error could be observed in a mismatched condition.

In the end-to-end network, it is natural to apply this technique to 52-dimensional decoder pre-softmax activations. The auto-encoder used here is a five-layer 512-unit feed-forward neural network, including a 24-dimensional bottleneck layer in the middle as shown in Fig. 6.2. PM score per utterance is derived as average MSE across all frames.

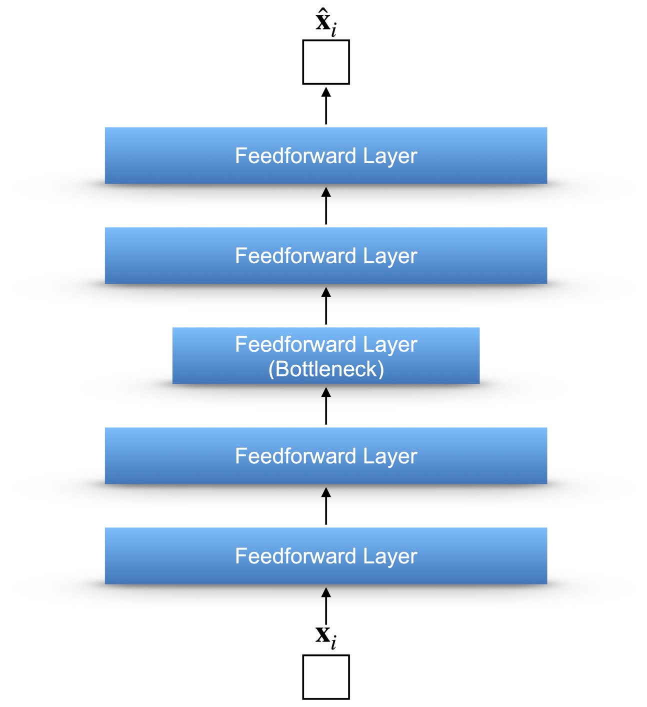


Figure 6.2. Configurations of auto-encoder PM techniques

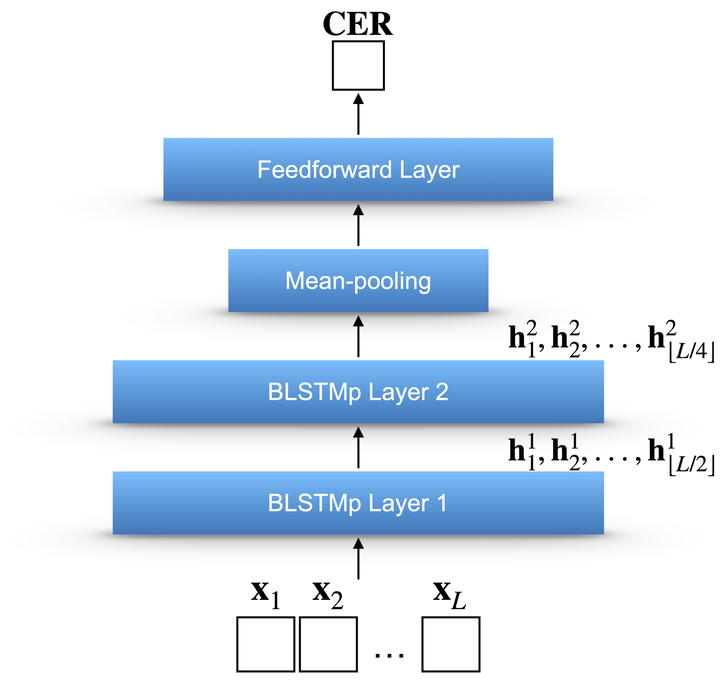

CER
Feedforward Layer
Mean-pooling
$\mathbf{h}^2_1, \mathbf{h}^2_2, \ldots, \mathbf{h}^2_{\lfloor L/4 \rfloor}$
BLSTMp Layer 2
$\mathbf{h}^1_1, \mathbf{h}^1_2, \ldots, \mathbf{h}^1_{\lfloor L/2 \rfloor}$
BLSTMp Layer 1
$\mathbf{x}_1$ $\mathbf{x}_2$ $\mathbf{x}_L$
…

Figure 6.3. Configurations of RNN predictor

6.4.4 RNN Predictor

In this work, we propose an RNN-based regression model which directly maps pre-softmax features of one utterance into error rates in the range of CER $[0, +\infty]$. The model is depicted in Fig. 6.3. Two BLSTM layers of 320 units are employed to handle temporal dependencies of inputs. Each layer subsamples every other output frame. A mean-pooling layer is then used on top of BLSTM outputs to formulate one summary vector per utterance, which is fed into a linear layer of 300 units and an output layer with one Rectified Linear Unit (ReLU). The model is optimized with MSE loss between predictions and truth CERs. The PM score is derived from the model output.

6.5 Results and Analysis

6.5.1 PM Score versus Truth Performance

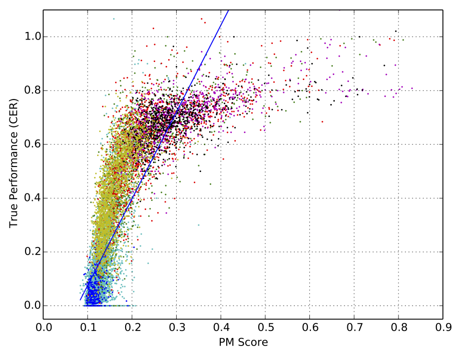
Fig. 6.4 shows scatter plots of PM scores versus truth performance (CER %) on *Test*, where each point in the plot represents one utterance. A linear model $CER = a*PM+b$ learned to minimize the mean squared error over *Dev* is also shown for each measure. In Fig. 6.4(a)(d) , entropy scores on the decoder output clearly demonstrate a linear relationship with true performance, while linear correlation for attention-level distributions holds only for error rates less than 0.5, resulting in larger prediction error overall. As shown in Fig. 6.4(b)(e) for MCD, similar to the observations for entropy measures, PM on decoder posteriors is better for predicting CERs than PM with attention probabilities. Furthermore, MCD is more linearly correlated with CERs than seen with entropy at attention level. In Fig. 6.4(c) for the measure using auto-encoder , reconstruction error prediction is consistent with the fitted line for CERs lower than 0.4, while the prediction error diverges for utterances with higher CERs. It might be the case that since the auto-encoder only sees the “good” data in training, there is lack of knowledge of how “bad” data looks. The scatter plot in Fig. 6.4(f) shows that the predictions of RNN predictor are well-aligned with the fitted line from linear regression. Since it is a direct estimation of CER, the ideal fit should be $CER = PM$. The derived model using *Dev* is $CER = 0.98 * PM + 0.06$, which is slightly offset and tilted from the ideal case.

6.5.2 Linear Regression Models on PM Measures

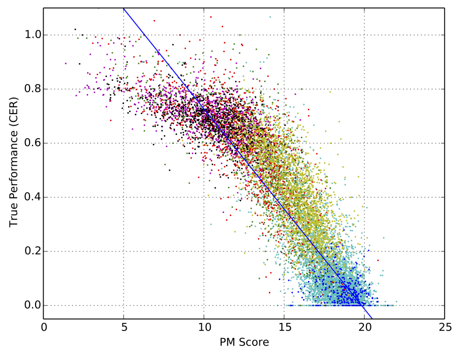
Table 6.3 summarizes MSEs of linear regression models trained on various PM techniques across different *Test* sets. It is worth noting that decoder features work more effectively than attention probabilities in all cases for predicting CERs. Entropy gives the lowest MSEs for WSJ and Aurora4, domains which have been seen in ASR training. The RNN predictor achieves best performance in CHiME-4-Sim (not surprising, since this domain was seen in RNN predictor training) and real recordings from CHiME-4-Real and DIRHA-Real, domains not seen in any training stage at all. MCD measure performs the best at the two unseen simulated domains with reverberant conditions. Overall, entropy, MCD, and RNN prediction all provide reasonably good CER predictions, with average prediction errors (square root of MSE) of 10.1%, 8.8% and 10.1%, respectively, where MCD outperforms the rest of PM measures across all test set.

6.5.3 PM Measures on Multi-Stream Model

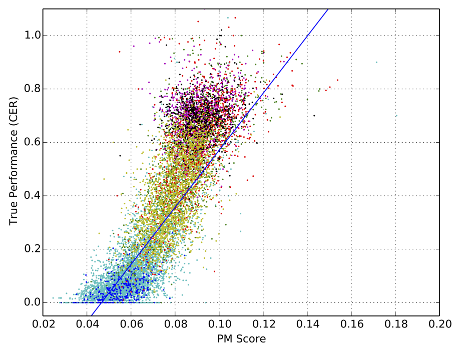
In this section, we discuss the effectiveness and limitation of PM measures on a multi-stream end-to-end model. In Sec. 6.5.2, MCD shows the best overall prediction power among four PM measures on a single-stream model. Therefore, MCD on decoder posteriors is chosen to be evaluated on a multi-stream model.



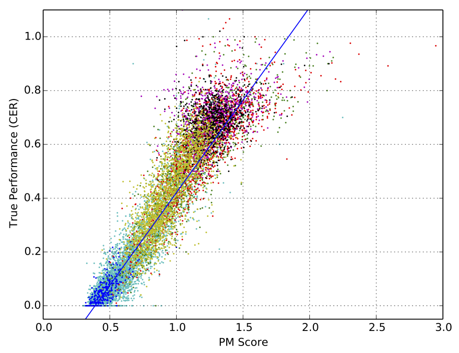
(a) Entropy (Attention Posteriors)



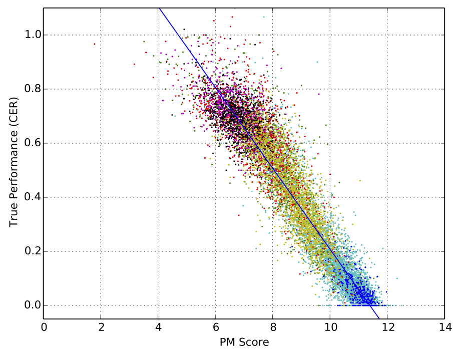
(b) MCD (Attention Posteriors)



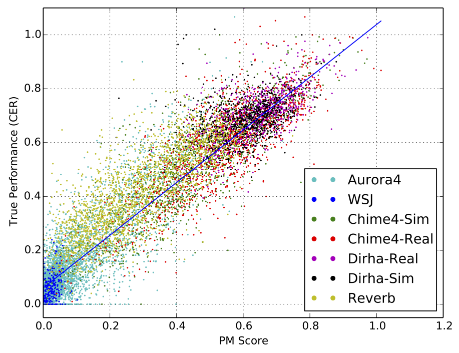
(c) Auto-encoder (Decoder Pre-softmax)



(d) Entropy (Decoder Posteriors)

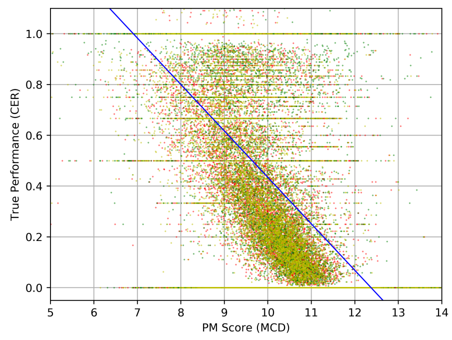


(e) MCD (Decoder Posteriors)

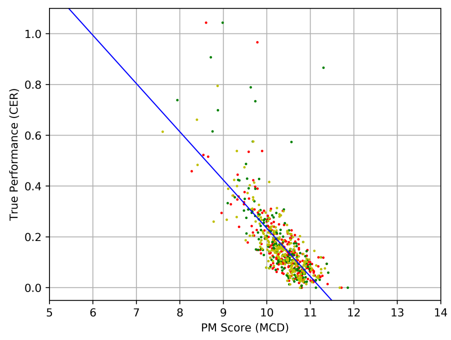


(f) RNN (Decoder Pre-softmax)

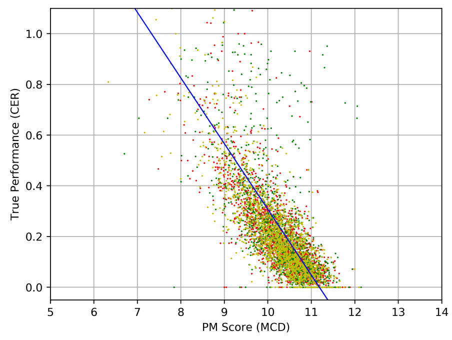
Figure 6.4. Performance monitoring score versus truth performance (CER %)



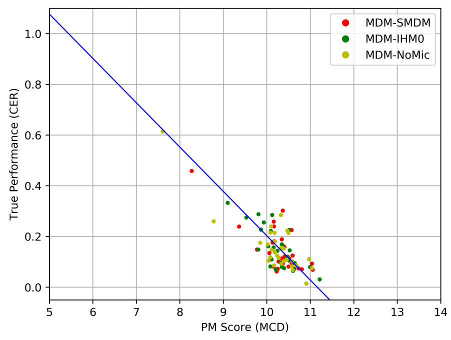
(a) $K = 0$



(c) $K = 10$



(b) $K = 5$



(d) $K = 15$

Figure 6.5. Discussion on minimal duration K in scatter plots of MCD scores versus true performance (CER %)

Table 6.3. Mean Square Error ($\times 10^{-2}$) of linear regression trained on performance monitoring techniques. All results are reported on test sets.

	Entropy		MCD			
Dataset/Domain	Dec	Att	Dec	Att	AE	RNN
Seen (ASR, PM)						
WSJ	**0.17**	0.88	0.25	0.77	0.82	0.29
Aurora4	**0.44**	1.43	0.47	1.14	1.12	0.74
Seen (PM only)						
CHiME-4-Sim	1.13	5.11	1.07	1.87	2.68	**1.01**
Unseen						
CHiME-4-Real	1.50	5.77	1.24	2.02	3.62	**1.05**
Reverb-Sim	0.94	3.42	**0.78**	2.20	1.88	1.50
DIRHA-Sim	2.23	6.75	**1.16**	2.05	6.07	1.43
DIRHA-Real	2.77	11.80	1.26	2.39	6.97	**1.25**
All Together(deg=1)	1.02	3.83	**0.79**	1.67	2.49	1.02
All Together(deg=3)	0.99	1.66	**0.74**	1.63	2.20	0.99
All Together(deg=5)	2.44	5.03	**0.77**	1.35	3.36	0.99

We select the MEM-Array model from Table 4.22, which is the best model on the AMI dataset. Similar to single-stream scenario, a linear regression model is trained on AMI *Dev* set and tested on three *Eval* sets: *MDM-SMDM* (seen condition), *MDM-NoMic* and *MDM-IHM0* (unseen conditions). The CER results with no additional RNN-LM are 28.4% (*MDM-SMDM*), 29.9% (*MDM-NoMic*), and 29.1%(*MDM-IHM0*).

Fig. 6.5 (a) shows the scatter plot of MCD scores versus performance (CER) on three *Eval* sets. We observe that a large amount of short utterances form horizontal lines on the scatter plot, which indicates that the PM measure could not well predict performances for these utterances. This is expected since MCD measure is computed as a sample mean of distribution distance (symmetric KL divergence). Lack of collected samples in short utterances may affect the predictive power of MCD. Hence, we created subsets from both *Dev* and *Eval* , where utterances only with duration longer than K

seconds are kept. Fig. 6.5 (b-d) illustrate the scatter plots using subsets generated with $K = 5, 10, 15$. Improved predictive power is reported on utterances with longer duration. Fig. 6.6 shows that the average prediction error decreases to below 20% with duration requirement $K \geq 10$, further supporting our hypothesis. Similar trend is also observed in entropy for the similar reason.

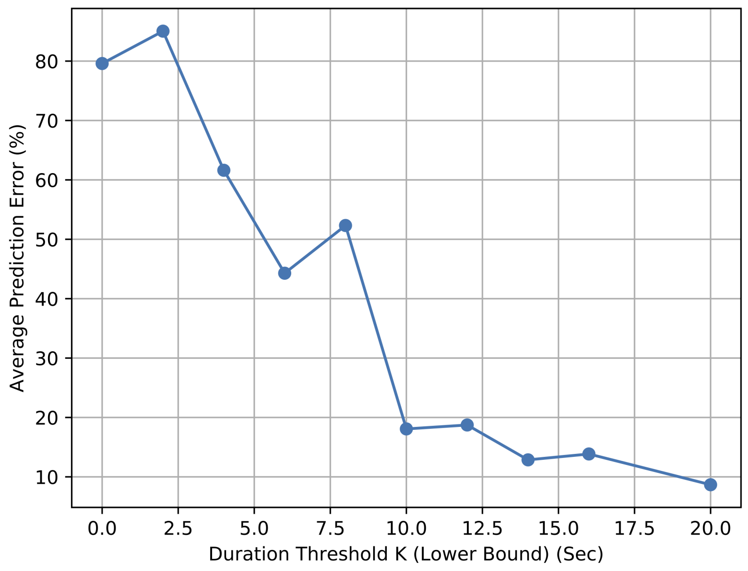


Figure 6.6. Average prediction error versus minimal duration K

6.6 Conclusion

In this chapter, we adapt previous PM measures (Entropy, M-measure and Auto-encoder) and apply our proposed RNN predictor in the end-to-end setting. These measures utilize the decoder output layer and attention probability vectors, and their predictive power is measured with simple linear models. Our findings suggest that decoder-level features are more feasible and informative than attention-level probabilities for PM measures. Entropy and MCD are very simple, effective measures where MCD shows the overall best performance with an average prediction error 8.8%.

Analysis on a multi-stream model suggests that both measures demonstrate better predictive power for long utterances. While auto-encoder methods may be suitable to handle mismatched conditions within a certain level of data corruption, an RNN-based regression model shows potential in the direction of performance estimation using deep neural networks, especially for unseen conditions. Overall, these results show great promise for performance prediction of end-to-end ASR models.

Chapter 7

Conclusion

7.1 Contributions

In this thesis, we focus on building an effective and robust multi-stream system within end-to-end ASR. The proposed framework is applied to two multi-stream scenarios with substantial improvements compared to best single streams and conventional fusion methods.

In Chapter 3, a general form of multi-stream E2E framework is proposed based on the joint CTC/ATT model. Parallel inputs are processed by separate encoders, which are further customized by individual CTC and attention networks. With flexibility for diverse encoder configurations, the proposed model is able to adapt to various multi-stream situations. The hierarchical attention network acts as an embedded performance monitor to dynamically guide the system towards more informative streams. Two representative frameworks targeting different applications, MEM-Array and MEM-Res, are present in Chapter 4 and Chapter 5, respectively.

Chapter 4 introduces the MEM-Array model addressing scenarios with distributed microphone arrays. Each encoder takes an array signal as input. Two levels of attention mechanisms combine knowledge in the output level, where frame synchronization across arrays is not required. Sec. 4.1 presents a direct adaptation of the general multi-stream form to multi-array applications. The experimental results show noticeable performance improvements over multiple single-stream models and conventional fusion strategies. A toy example with one-stream noise injection illustrates robust selectivity of the HAN component. Furthermore, a practical two-stage training strategy in Sec. 4.2 offers a framework for effective training with less computational cost while substantially improving performance. Stage-1 concentrates on optimizing a universal feature extractor to benefit from all non-parallel data, whereas Stage-2 greatly scales the training with focus solely on the multi-stream fusion. In our experiments, this strategy achieves relative WER reductions of $8.2 - 32.4\%$ compared with our previous method. To make systems robust against various environmental distortions, such as background noises and reverberation, we further propose a two-stage augmentation scheme and adaptive CTC fusion in Sec. 4.3: Stage-1 Augmentation intends to handle unseen single-stream conditions via online and offline augmentations; Stage-2 Time Masking mimics random microphone dropout to deal with inter-stream dynamics; Adaptive CTC takes advantage of stream attention vectors without using predefined knowledge for CTC fusion. Substantial improvements are reported with relative WER reductions of $29.7 - 59.3\%$ across several unseen stream combinations. Our analysis also shows that a smaller amount of parallel data is sufficient for an improved MEM-Array model.

In Chapter 5, the MEM-Res model is a multi-stream formulation based on a single input. Parallel encoders with different configurations and temporal resolutions aim to represent diverse information from the same acoustic. Hierarchical attention then combines these complementary knowledge dynamically. Our investigation of multi-resolution demonstrates that diverse temporal resolution enhances knowledge fusion, supporting our hypothesis. By matching single-stream models with similar parameters for fair comparison, this multi-stream realization still achieves lower WERs. The MEM-Res model also beats multi-stream models without the HAN component illustrating the fusion power of the framework. Especially, the result on WSJ is reported with 3.6% WER outperforming several previous studies.

It is beneficial in many scenarios to inform users for potential system failure without knowing ground-truth beforehand. In Chapter 6, we develop four performance monitoring measures customized for the E2E setting: Entropy, Mean Character Distance, Auto-encoder, and RNN predictor. Our results show that PM measures on decoder features are more effective for predicting true error rates than PM measures on attention probabilities. Entropy and MCD are very simple, effective measures where MCD shows the overall best performance. While auto-encoder methods might be suitable to handle mismatched conditions within a certain level of data corruption, an RNN-based regression model shows potential in the direction of performance estimation using DNNs. We also discuss the statistical nature of MCD and Entropy measures in which both measures tend to perform more effectively with long duration utterances.

The proposed multi-stream architecture has been implemented in ESPnet [95], an open source toolkit mainly focusing on end-to-end speech recognition and end-to-end text-to-speech.

7.2 Future Directions

We outline a number of promising future directions based on the approaches described in the preceding chapters.

Strategy when adding extra streams. With increasing use of voice-controlled devices in domestic settings, it is practical to consider the scenario when adding an extra stream (microphone array) into a well-established system. It is fairly expensive and unrealistic to retrain the whole giant network when adding one stream every time. The proposed two-stage training strategy better defines a scheme for additional stream: Directly utilizing the UFE from Stage-1 for the new stream, Stage-2 optimizes the stream fusion integrating with the additional stream. The future direction remains to demonstrate the efficiency of this approach.

Sophisticated stream fusion. In the setup in Sec. 4.1, due to the memory issue, the stream fusion module only uses a fairly simple content-based attention mechanism, which is just a one-layer neural network. Later on, the proposed two-stage training alleviates the burden of excessive memory use during training, which makes it possible to explore more complex stream fusion mechanisms for noise robustness. There are two proposals in this direction:

1. **Provide information across streams for stream fusion**. Currently, stream weight is computed using features from each stream independently, followed by a Softmax module to normalize weights across all streams. Intuitively, it would be helpful to feed information from other streams to well model the hierarchical attention across streams. Applying self-attention across all contextual vectors is one possible approach; Averaging the contextual vectors from other streams could provide an additional input to compute stream attention.

2. **Add stream specific layer for training**. In the current setting, the stream fusion module does not distinguish stream-specific characteristics. In most of the cases, when an array is placed in a home domestic scenario, the acoustic condition in its surroundings stays similar. Injecting such information in a multi-

stream framework could be beneficial for robustness. In Stage-2 of two-stage strategy, adding trainable parameters for each stream is worth investigating.

CTC influence in multi-stream. In the CTC/ATT model with a single encoder, the CTC objective serves as an auxiliary task to speed up the procedure of realizing monotonic alignment. In multi-stream framework, per-encoder CTC is introduced where a separate CTC mechanism is active for each encoder stream during training and decoding. Instead of assigning stream weights for each CTC, another approach could be decreasing the effect of CTC and replying on attention-based modules for inference. In the two-stage strategy, Stage-2 focuses on fusion module training. Training with CTCs in Stage-2 is redundant since monotonic behavior is the focus of Stage-1 training. Hence, methods to eliminate influence from CTC are worth exploring for a simplified system.

Bibliography

1. Bahl, L. R., Jelinek, F. & Mercer, R. L. A maximum likelihood approach to continuous speech recognition. *IEEE transactions on pattern analysis and machine intelligence,* 179–190 (1983).
2. Jelinek, F., Bahl, L. & Mercer, R. Design of a linguistic statistical decoder for the recognition of continuous speech. *IEEE Transactions on Information Theory* **21,** 250–256 (1975).
3. Bourlard, H. & Wellekens, C. J. Links between Markov models and multilayer perceptrons. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **12,** 1167–1178 (1990).
4. Rabiner, L. R. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE* **77,** 257–286 (1989).
5. Graves, A., Fernandez, S., Gomez, F. & Schmidhuber, J. *Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks* in *International Conference on Machine learning (ICML)* (2006), 369–376.
6. Graves, A. & Jaitly, N. *Towards end-to-end speech recognition with recurrent neural networks* in *International Conference on Machine Learning (ICML)* (2014), 1764–1772.
7. Miao, Y., Gowayyed, M. & Metze, F. *EESEN: End-to-end speech recognition using deep RNN models and WFST-based decoding* in *IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)* (2015), 167–174.
8. Chorowski, J. K., Bahdanau, D., Serdyuk, D., Cho, K. & Bengio, Y. *Attention-based models for speech recognition* in *Advances in Neural Information Processing Systems (NIPS)* (2015), 577–585.
9. Chan, W., Jaitly, N., Le, Q. V. & Vinyals, O. *Listen, Attend and Spell: A Neural Network for Large Vocabulary Conversational Speech Recognition* in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (2015).
10. Graves, A. *Sequence Transduction with Recurrent Neural Networks* in *Proc. of ICML Workshop on Representation Learning* (2012).
11. Graves, A., Mohamed, A.-r. & Hinton, G. *Speech recognition with deep recurrent neural networks* in *Proc. of ICASSP* (2013), 6645–6649.

12. Kim, S., Hori, T. & Watanabe, S. *Joint CTC-attention based end-to-end speech recognition using multi-task learning* in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (2017), 4835–4839.

13. Hori, T., Watanabe, S., Zhang, Y. & Chan, W. *Advances in Joint CTC-Attention based End-to-End Speech Recognition with a Deep CNN Encoder and RNN-LM* in *INTERSPEECH* (2017).

14. Watanabe, S., Hori, T., Kim, S., Hershey, J. R. & Hayashi, T. Hybrid CTC/attention architecture for end-to-end speech recognition. *IEEE Journal of Selected Topics in Signal Processing* **11,** 1240–1253 (2017).

15. Hochreiter, S. & Schmidhuber, J. Long short-term memory. *Neural computation* **9,** 1735–1780 (1997).

16. Graves, A., Jaitly, N. & Mohamed, A.-r. *Hybrid speech recognition with deep bidirectional LSTM* in *2013 IEEE workshop on automatic speech recognition and understanding* (2013), 273–278.

17. Luong, T., Pham, H. & Manning, C. D. *Effective Approaches to Attention-based Neural Machine Translation* in *Proc. of EMNLP* (Association for Computational Linguistics, Lisbon, Portugal, Sept. 2015), 1412–1421.

18. Hou, J., Zhang, S. & Dai, L.-R. *Gaussian Prediction Based Attention for Online End-to-End Speech Recognition* in *Proc. of INTERSPEECH* (2017).

19. Tjandra, A., Sakti, S. & Nakamura, S. *Local Monotonic Attention Mechanism for End-to-End Speech And Language Processing* in *IJCNLP* (2017).

20. Raffel, C., Luong, M.-T., Liu, P. J., Weiss, R. J. & Eck, D. *Online and linear-time attention by enforcing monotonic alignments* in *Proc. of ICML* (2017), 2837–2846.

21. Chiu, C.-C. & Raffel, C. *Monotonic chunkwise attention* in *Proc. of ICLR* (2018).

22. Graves, A. *Supervised Sequence Labelling with Recurrent Neural Networks* PhD thesis (Universitat Munchen, 2008).

23. Hori, T., Watanabe, S. & Hershey, J. *Joint CTC/attention decoding for end-to-end speech recognition* in *Proc. of ACL* (2017), 518–529.

24. Hori, T., Cho, J. & Watanabe, S. *End-to-end speech recognition with word-based RNN language models* in *Proc. of SLT* (2018), 389–396.

25. Kinoshita, K. *et al.* A summary of the REVERB challenge: state-of-the-art and remaining challenges in reverberant speech processing research. *EURASIP Journal on Advances in Signal Processing* **2016,** 7 (2016).

26. Barker, J., Marxer, R., Vincent, E. & Watanabe, S. The third 'CHiME'speech separation and recognition challenge: Analysis and outcomes. *Computer Speech & Language* **46,** 605–626 (2017).

27. Benesty, J., Chen, J. & Huang, Y. *Microphone array signal processing* (Springer Science & Business Media, 2008).

28. Van Veen, B. D. & Buckley, K. M. Beamforming: A versatile approach to spatial filtering. *IEEE assp magazine* **5,** 4–24 (1988).

29. Wolfel, M. & McDonough, J. *Distant speech recognition* (John Wiley & Sons, 2009).

30. Higuchi, T., Ito, N., Yoshioka, T. & Nakatani, T. *Robust MVDR beamforming using time-frequency masks for online/offline ASR in noise* in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (2016), 5210–5214.

31. Tu, Y., Du, J., Sun, L., Ma, F. & Lee, C.-H. *On Design of Robust Deep Models for CHiME-4 Multi-Channel Speech Recognition with Multiple Configurations of Array Microphones.* in *INTERSPEECH* (2017), 394–398.

32. Menne, T. *et al. The RWTH/UPB/FORTH system combination for the 4th CHiME challenge evaluation* in *CHiME-4 workshop* (2016).

33. Xiao, X. *et al. Deep beamforming networks for multi-channel speech recognition* in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (2016), 5745–5749.

34. Li, B., Sainath, T. N., Weiss, R. J., Wilson, K. W. & Bacchiani, M. Neural network adaptive beamforming for robust multichannel speech recognition (2016).

35. Meng, Z., Watanabe, S., Hershey, J. R. & Erdogan, H. *Deep long short-term memory adaptive beamforming networks for multichannel robust speech recognition* in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (2017), 271–275.

36. Heymann, J., Drude, L. & Haeb-Umbach, R. *Neural network based spectral mask estimation for acoustic beamforming* in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (2016), 196–200.

37. Xiao, X. *et al. A study of learning based beamforming methods for speech recognition* in *CHiME 2016 workshop* (2016), 26–31.

38. Erdogan, H. *et al. Multi-channel speech recognition: Lstms all the way through* in *CHiME-4 workshop* (2016), 1–4.

39. Erdogan, H., Hershey, J. R., Watanabe, S., Mandel, M. I. & Le Roux, J. *Improved mvdr beamforming using single-channel mask prediction networks.* in *Interspeech* (2016), 1981–1985.

40. Heymann, J., Drude, L., Boeddeker, C., Hanebrink, P. & Haeb-Umbach, R. *Beamnet: End-to-end training of a beamformer-supported multi-channel ASR system* in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (2017), 5325–5329.

41. Xiao, X., Zhao, S., Jones, D. L., Chng, E. S. & Li, H. *On time-frequency mask estimation for MVDR beamforming with application in robust speech recognition* in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (2017), 3246–3250.

42. Nakatani, T., Ito, N., Higuchi, T., Araki, S. & Kinoshita, K. *Integrating DNN-based and spatial clustering-based mask estimation for robust MVDR beamforming* in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (2017), 286–290.

43. Pfeifenberger, L., Zohrer, M. & Pernkopf, F. *DNN-based speech mask estimation for eigenvector beamforming* in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (2017), 66–70.

44. Ochiai, T., Watanabe, S., Hori, T. & Hershey, J. R. *Multichannel end-to-end speech recognition* in *Proc. of ICML* (2017), 2632–2641.

45. Ochiai, T., Watanabe, S., Hori, T., Hershey, J. R. & Xiao, X. Unified architecture for multichannel end-to-end speech recognition with neural beamforming. *IEEE Journal of Selected Topics in Signal Processing* **11,** 1274–1288 (2017).

46. Subramanian, A. S. *et al. Speech enhancement using end-to-end speech recognition objectives* in *2019 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)* (2019), 234–238.

47. Kim, S., Lane, I., Kim, S. & Lane, I. *End-to-End Speech Recognition with Auditory Attention for Multi-Microphone Distance Speech Recognition.* in *Interspeech* (2017), 3867–3871.

48. Braun, S., Neil, D., Anumula, J., Ceolini, E. & Liu, S.-C. *Multi-channel Attention for End-to-End Speech Recognition* in *Proc. Interspeech 2018* (2018), 17–21.

49. Araki, S., Ono, N., Kinoshita, K. & Delcroix, M. *Meeting recognition with asynchronous distributed microphone array* in *2017 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)* (2017), 32–39.

50. Araki, S., Ono, N., Kinoshita, K. & Delcroix, M. *Meeting recognition with asynchronous distributed microphone array using block-wise refinement of mask-based MVDR beamformer* in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (2018), 5694–5698.

51. Yoshioka, T. *et al. Meeting Transcription Using Asynchronous Distant Microphones.* in *INTERSPEECH* (2019), 2968–2972.

52. Du, J. *et al. The USTC-iFlytek Systems for CHiME-5 Challenge* in *CHiME-5* (2018).

53. Barker, J., Watanabe, S., Vincent, E. & Trmal, J. *The Fifth 'CHiME' Speech Separation and Recognition Challenge: Dataset, Task and Baselines* in *Interspeech 2018* (2018), 1561–1565.

54. Fletcher, H. Speech and hearing in communication (1953).

55. Hermansky, H. Multistream recognition of speech: Dealing with unknown unknowns. *Proceedings of the IEEE* **101,** 1076–1088 (2013).

56. Tibrewala, S. & Hermansky, H. *Sub-band based recognition of noisy speech* in *1997 IEEE International Conference on Acoustics, Speech, and Signal Processing* **2** (1997), 1255–1258.

57. Sharma, S. R. *et al. Multi-stream approach to robust speech recognition* PhD thesis (Ph. D.), Oregon Graduate Institute, 1999).

58. Ogawa, T., Li, F. & Hermansky, H. *Stream selection and integration in multistream ASR using GMM-based performance monitoring.* in *INTERSPEECH* (2013), 3332–3336.

59. Mallidi, S. H., Ogawa, T., Vesely, K., Nidadavolu, P. S. & Hermansky, H. *Autoencoder based multi-stream combination for noise robust speech recognition* in *Sixteenth Annual Conference of the International Speech Communication Association* (2015).

60. Mallidi, S. H. & Hermansky, H. *Novel neural network based fusion for multistream ASR* in *Acoustics, Speech and Signal Processing (ICASSP), 2016 IEEE International Conference on* (2016), 5680–5684.

61. Mallidi, S. H. R. *et al. A practical and efficient multistream framework for noise robust speech recognition* PhD thesis (Johns Hopkins University, 2018).

62. Hermansky, H. Coding and decoding of messages in human speech communication: Implications for machine recognition of speech. *Speech Communication* (2018).

63. Mesgarani, N., Thomas, S. & Hermansky, H. Toward optimizing stream fusion in multistream recognition of speech. *The Journal of the Acoustical Society of America* **130,** EL14–EL18 (2011).

64. Mallidi, S. H., Ogawa, T. & Hermansky, H. *Uncertainty estimation of DNN classifiers* in *Proc. of ASRU* (2015), 283–288.

65. Mallidi, S. H. R. & Hermansky, H. *A framework for practical multistream asr.* in *INTERSPEECH* (2016), 3474–3478.

66. Misra, H., Bourlard, H. & Tyagi, V. *New entropy based combination rules in HMM/ANN multi-stream ASR* in *Proc. of ICASSP* **2** (2003), II–741.

67. Wang, X., Yan, Y. & Hermansky, H. Stream Attention for far-field multi-microphone ASR. *arXiv preprint arXiv:1711.11141* (2017).

68. Wang, X., Li, R. & Hermansky, H. Stream Attention for Distributed Multi-Microphone Speech Recognition. *Proc. Interspeech 2018,* 3033–3037 (2018).

69. Xiong, F. *et al. Channel Selection using Neural Network Posterior Probability for Speech Recognition with Distributed Microphone Arrays in Everyday Environments* in *CHiME-5* (2018).

70. Hermansky, H., Variani, E. & Peddinti, V. *Mean temporal distance: Predicting ASR error from temporal properties of speech signal* in *2013 IEEE International Conference on Acoustics, Speech and Signal Processing* (2013), 7423–7426.

71. Kanda, N. *et al. The Hitachi/JHU CHiME-5 system: Advances in speech recognition for everyday home environments using multiple microphone arrays* in *CHiME-5* (2018).

72. Fiscus, J. G. *A post-processing system to yield reduced word error rates: Recognizer output voting error reduction (ROVER)* in *Proc. of ASRU* (1997), 347–354.

73. Xu, H., Povey, D., Mangu, L. & Zhu, J. Minimum bayes risk decoding and system combination based on a recursion for edit distance. *Computer Speech & Language* **25,** 802–828 (2011).

74. Meyer, B. T. *et al. Performance monitoring for automatic speech recognition in noisy multi-channel environments* in *Proc. of SLT* (2016), 50–56.

75. Wang, Y., Deng, X., Pu, S. & Huang, Z. Residual convolutional CTC networks for automatic speech recognition. *arXiv preprint arXiv:1702.07793* (2017).

76. Palaskar, S., Sanabria, R. & Metze, F. *End-to-end multimodal speech recognition* in *Proc. of ICASSP* (2018), 5774–5778.

77. Renduchintala, A., Ding, S., Wiesner, M. & Watanabe, S. Multi-Modal Data Augmentation for End-to-end ASR. *Proc. Interspeech 2018,* 2394–2398 (2018).

78. Yang, Z. *et al. Hierarchical attention networks for document classification* in *NAACL HLT* (2016), 1480–1489.

79. Hori, C. *et al. Attention-based multimodal fusion for video description* in *ICCV 2017* (2017), 4203–4212.

80. Libovicky, J. & Helcl, J. *Attention Strategies for Multi-Source Sequence-to-Sequence Learning* in *ACL 2017* **2** (2017), 196–202.

81. Vaswani, A. *et al. Attention is all you need* in *Advances in Neural Information Processing Systems* (2017), 5998–6008.

82. Hayashi, T., Watanabe, S., Toda, T. & Takeda, K. *Multi-Head Decoder for End-to-End Speech Recognition* in *Proc. Interspeech 2018* (2018), 801–805.

83. Chiu, C.-C. *et al. State-of-the-art speech recognition with sequence-to-sequence models* in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (2018), 4774–4778.

84. Pundak, G., Sainath, T. N., Prabhavalkar, R., Kannan, A. & Zhao, D. *Deep context: end-to-end contextual speech recognition* in *Proc. of SLT* (2018), 418–425.

85. Kim, S. & Metze, F. *Dialog-context aware end-to-end speech recognition* in *Proc. of SLT* (2018), 434–440.

86. Carletta, J. *et al. The AMI meeting corpus: A pre-announcement* in *International Workshop on Machine Learning for Multimodal Interaction* (2005), 28–39.

87. Ravanelli, M., Svaizer, P. & Omologo, M. *Realistic Multi-Microphone Data Simulation for Distant Speech Recognition* in *Proc. of INTERSPEECH* (2016).

88. Vincent, E., Watanabe, S., Nugraha, A. A., Barker, J. & Marxer, R. An analysis of environment, microphone and data simulation mismatches in robust speech recognition. *Computer Speech & Language* **46,** 535–557 (2017).

89. Wang, Z., Wang, X., Li, X., Fu, Q. & Yan, Y. *Oracle performance investigation of the ideal masks* in *IWAENC 2016* (2016), 1–5.

90. Markovich-Golan, S., Bertrand, A., Moonen, M. & Gannot, S. Optimal distributed minimum-variance beamforming approaches for speech enhancement in wireless acoustic sensor networks. *Signal Processing* **107,** 4–20 (2015).

91. Zhang, Y., Chan, W. & Jaitly, N. *Very Deep Convolutional Networks for End-to-End Speech Recognition* in *IEEE International Conference on Acoustics, Speech and Signal Processing* (2017).

92. Cho, J. *et al. Multilingual sequence-to-sequence speech recognition: architecture, transfer learning, and language modeling* in *SLT 2018* (2018).

93. Anguera, X., Wooters, C. & Hernando, J. Acoustic beamforming for speaker diarization of meetings. *IEEE Transactions on Audio, Speech, and Language Processing* **15,** 2011–2022 (2007).

94. Simonyan, K. & Zisserman, A. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556* (2014).

95. Watanabe, S. *et al. ESPnet: End-to-End Speech Processing Toolkit* in *Interspeech 2018* (2018), 2207–2211.

96. Pereyra, G., Tucker, G., Chorowski, J., Kaiser, Ł. & Hinton, G. *Regularizing Neural Networks by Penalizing Confident Output Distributions* 2017. arXiv: 1701.06548 [cs.NE].

97. Park, D. S. *et al.* SpecAugment: A Simple Data Augmentation Method for Automatic Speech Recognition. *Proc. of INTERSPEECH* (2019).

98. Consortium, L. D. CSR-II (WSJ1) Complete. *Linguistic Data Consortium, Philadelphia* **LDC94S13A** (1994).

99. Vincent, E., Watanabe, S., Barker, J. & Marxer, R. *The 4th CHiME speech separation and recognition challenge* 2016.

100. Chorowski, J. & Jaitly, N. Towards better decoding and language model integration in sequence to sequence models. *arXiv preprint arXiv:1612.02695* (2016).

101. Zhou, Y., Xiong, C. & Socher, R. Improved Regularization Techniques for End-to-End Speech Recognition. *arXiv preprint arXiv:1712.07108* (2017).

102. Zeghidour, N., Usunier, N., Synnaeve, G., Collobert, R. & Dupoux, E. End-to-End Speech Recognition From the Raw Waveform. *arXiv preprint arXiv:1806.07098* (2018).

103. Hadian, H., Sameti, H., Povey, D. & Khudanpur, S. End-to-end speech recognition using lattice-free MMI. *Proc. Interspeech 2018,* 12–16 (2018).

104. K. Scheffers, M. & Coles, M. Performance monitoring in a confusing world: Error-related brain activity, judgments of response accuracy, and types of errors. *Journal of Experimental Psychology: Human Perception and Performance* **26,** 141–151 (Feb. 2000).

105. Okawa, S., Bocchieri, E. & Potamianos, A. *Multi-band speech recognition in noisy environments* in *Proceedings of the 1998 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP'98 (Cat. No. 98CH36181)* **2** (1998), 641–644.

106. Martinez, A. M. C. *et al.* DNN-based performance measures for predicting error rates in automatic speech recognition and optimizing hearing aid parameters. *Speech Communication* **106,** 44–56 (2019).

107. Ganapathy, S., Omar, M. & Pelecanos, J. *Unsupervised channel adaptation for language identification using co-training* in *2013 IEEE International Conference on Acoustics, Speech and Signal Processing* (2013), 6857–6861.

108. Variani, E., Li, F. & Hermansky, H. *Multi-stream recognition of noisy speech with performance monitoring.* in *Interspeech* (2013), 2978–2981.

109. Pearce, D. & Picone, J. Aurora working group: DSR front end LVCSR evaluation AU/384/02. *Inst. for Signal & Inform. Process., Mississippi State Univ., Tech. Rep* (2002).

110. Kinoshita, K. *et al. The reverb challenge: A common evaluation framework for dereverberation and recognition of reverberant speech* in *2013 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics* (Oct. 2013), 1–4.

Vita

Ruizhi Li received his Bachelor of Engineering in Automation from Beijing University of Chemical Technology, Beijing, China in 2012, and Master of Science in Electrical Engineering from Washington University in St. Louis, St. Louis, MO in 2014. He joined the Ph.D. program in the Department of Electrical Engineering at the Johns Hopkins University, Baltimore, MD in August 2014. He worked with Dr. Hynek Hermansky at the Center for Language and Speech Processing, focusing on multi-stream scenarios in end-to-end speech recognition. His research interests include machine learning, speech recognition, speaker recognition, and language identification. In the past, he has been part of several summer workshops at CLSP, and has also worked at Brno University of Technology, Brno, Czech Republic, Amazon Lab 126, Sunnyvale, CA, and Amazon, Seattle, WA.

In October 2020, Ruizhi started as a Senior Applied Scientist at Microsoft in Bellevue, WA.