Hybrid CTC/Attention Architecture for End-to-End Speech Recognition

Shinji Watanabe, *Senior Member, IEEE*, Takaaki Hori, *Senior Member, IEEE*, Suyoun Kim, *Student Member, IEEE*, John R. Hershey, *Senior Member, IEEE*, and Tomoki Hayashi, *Student Member, IEEE*

***Abstract*—Conventional automatic speech recognition (ASR) based on a hidden Markov model (HMM)/deep neural network (DNN) is a very complicated system consisting of various modules such as acoustic, lexicon, and language models. It also requires linguistic resources, such as a pronunciation dictionary, tokenization, and phonetic context-dependency trees. On the other hand, end-to-end ASR has become a popular alternative to greatly simplify the model-building process of conventional ASR systems by representing complicated modules with a single deep network architecture, and by replacing the use of linguistic resources with a data-driven learning method. There are two major types of end-to-end architectures for ASR; attention-based methods use an attention mechanism to perform alignment between acoustic frames and recognized symbols, and connectionist temporal classification (CTC) uses Markov assumptions to efficiently solve sequential problems by dynamic programming. This paper proposes hybrid CTC/attention end-to-end ASR, which effectively utilizes the advantages of both architectures in training and decoding. During training, we employ the multiobjective learning framework to improve robustness and achieve fast convergence. During decoding, we perform joint decoding by combining both attention-based and CTC scores in a one-pass beam search algorithm to further eliminate irregular alignments. Experiments with English (WSJ and CHiME-4) tasks demonstrate the effectiveness of the proposed multiobjective learning over both the CTC and attention-based encoder–decoder baselines. Moreover, the proposed method is applied to two large-scale ASR benchmarks (spontaneous Japanese and Mandarin Chinese), and exhibits performance that is comparable to conventional DNN/HMM ASR systems based on the advantages of both multiobjective learning and joint decoding without linguistic resources.**

***Index Terms*—Automatic speech recognition, end-to-end, connectionist temporal classification, attention mechanism, hybrid CTC/attention.**

I. Introduction

AUTOMATIC speech recognition (ASR) is an essential technology for realizing natural human–machine interfaces. It has become a mature set of technologies that have been widely deployed, resulting in great success in interface applications such as voice search. A typical ASR system is factorized into several modules including acoustic, lexicon, and language models based on a probabilistic noisy channel model [1]. Over the last decade, dramatic improvements in acoustic and language models have been driven by machine learning techniques known as deep learning [2]. However, current systems lean heavily on the scaffolding of complicated legacy architectures that developed around traditional techniques. They present the following problems that we may seek to eliminate.

1) *Stepwise refinement:* Many module-specific processes are required to build an accurate module. For example, when we build an acoustic model from scratch, we have to first build a hidden Markov model (HMM) and Gaussian mixture models (GMMs) to obtain the tied-state HMM structure and phonetic alignments, before we can train deep neural networks (DNNs).
2) *Linguistic information:* To factorize acoustic and language models well, we need to have a lexicon model, which is usually based on a handcrafted pronunciation dictionary to map word to phoneme sequences. Since phonemes are designed using linguistic knowledge, they are subject to human error that a fully data-driven system might avoid. Finally, some languages do not explicitly have a word boundary and need tokenization modules [3], [4].
3) *Conditional independence assumptions:* The current ASR systems often use conditional independence assumptions (especially Markov assumptions) during the above factorization and to make use of GMM, DNN, and n-gram models. Real-world data do not necessarily follow such assumptions leading to model misspecification.
4) *Complex decoding:* Inference/decoding has to be performed by integrating all modules. Although this integration is often efficiently handled by finite state transducers, the construction and implementation of well-optimized transducers are very complicated [5], [6].
5) *Incoherence in optimization:* The above modules are optimized separately with different objectives, which may result in incoherence in optimization, where each module is not trained to match the other modules.

Consequently, it is quite difficult for nonexperts to use/develop ASR systems for new applications, especially for new languages.

Manuscript received April 1, 2017; revised October 10, 2017; accepted October 11, 2017. Date of publication October 16, 2017; date of current version November 16, 2017. The guest editor coordinating the review of this paper and approving it for publication was Dr. Kate Knill. *(Corresponding author: Shinji Watanabe.)*

S. Watanabe, T. Hori, and J. R. Hershey are with Mitsubishi Electric Research Laboratories, Cambridge, MA 02139 USA (e-mail: shinjiw@ieee.org; thori@merl.com; hershey@merl.com).

S. Kim is with Carnegie Mellon University, Pittsburgh, PA 15213 USA (e-mail: suyoun@cmu.edu).

T. Hayashi is with Nagoya University, Nagoya 464-8603, Japan (e-mail: hayashi.tomoki@g.sp.m.is.nagoya-u.ac.jp).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/JSTSP.2017.2763455

End-to-end ASR has the goal of simplifying the above module-based architecture into a single-network architecture within a deep learning framework in order to address the above issues. There are two major types of end-to-end architectures for ASR; attention-based methods use an attention mechanism to perform alignment between acoustic frames and recognized symbols, and connectionist temporal classification (CTC) uses Markov assumptions to efficiently solve sequential problems by dynamic programming [7], [8].

All ASR models aim to elucidate the posterior distribution, $p(W|X)$, of a word sequence, W, given a speech feature sequence X. End-to-end methods directly carry this out whereas conventional models factorize $p(W|X)$ into modules such as the language model, $p(W)$, which can be trained on pure language data, and an acoustic model likelihood, $p(X|W)$, which is trained on acoustic data with the corresponding language labels. End-to-end ASR methods typically rely only on paired acoustic and language data. Without the additional language data, they can suffer from data sparseness or out-of-vocabulary issues. To improve generalization, and handle out-of-vocabulary problems, it is typical to use the letter representation rather than the word representation for the language output sequence, which we adopt in the descriptions below.

The attention-based end-to-end method solves the ASR problem as a sequence mapping from speech feature sequences to text by using an encoder–decoder architecture. The decoder network uses an attention mechanism to find an alignment between each element of the output sequence and the hidden states generated by the acoustic encoder network for each frame of acoustic input [7], [9]–[11]. At each output position, the decoder network computes a matching score between its hidden state and the states of the encoder network at each input time, to form a temporal alignment distribution, which is then used to extract an average of the corresponding encoder states.

This basic temporal attention mechanism is too flexible in the sense that it allows extremely nonsequential alignments. This may be fine for applications such as machine translation where the input and output word orders are different [12], [13]. However in speech recognition, the feature inputs and corresponding letter outputs generally proceed in the same order with only small within-word deviations (e.g., the word “iron,” which transposes the sounds for “r” and “o”). Another problem is that the input and output sequences in ASR can have very different lengths, and they vary greatly from case to case, depending on the speaking rate and writing system, making it more difficult to track the alignment.

However, an advantage is that the attention mechanism does not require any conditional independence assumptions, and could address all of the problems cited above. Although the alignment problems of attention-based mechanisms have been partially addressed in [7], [14] using various mechanisms, here we propose more rigorous constraints by using CTC-based alignment to guide the training.

CTC permits the efficient computation of a strictly monotonic alignment using dynamic programming [8], [15] although it requires separate language models and graph-based decoding [16], except in the case of huge training data [17], [18]. We propose to take advantage of the constrained CTC alignment in a hybrid CTC/attention-based system. During training, we propose a multiobjective learning method by attaching a CTC objective to an attention-based encoder network as a regularization [19]. This greatly reduces the number of irregularly aligned utterances without any heuristic search techniques. During decoding, we propose a joint decoding approach, which combines both attention-based and CTC scores in a rescoring/one-pass beam search algorithm to eliminate the irregular alignments [20].

The proposed method is first applied to English-read-speech ASR tasks to mainly show the effectiveness of the multiobjective learning of our hybrid CTC/attention architecture. Then, the method is further applied to Japanese and Mandarin ASR tasks, which require extra linguistic resources including a morphological analyzer [3] or word segmentation [21] in addition to a pronunciation dictionary to provide accurate lexicon and language models in conventional DNN/HMM ASR. Surprisingly, the method achieved performance comparable to, and in some cases superior to, several state-of-the-art HMM/DNN ASR systems, without using the above linguistic resources, when both multiobjective learning and joint decoding are used.

This paper summarizes our previous studies of the hybrid CTC/attention architecture [19], [20], which focus on its training and decoding functions, respectively. The paper extends [19] and [20] by providing more detailed formulations from conventional HMM/DNN systems to current end-to-end ASR systems (Section II), a consistent formulation of the hybrid CTC/attention architecture for training and decoding with precise implementations (Section III), and more experimental discussions (Section IV).

II. From HMM/DNN to End-to-End ASR

This section provides a formulation of conventional HMM/DNN ASR and CTC or attention-based end-to-end ASR. The formulation is intended to clarify the probabilistic factorizations and conditional independence assumptions (Markov assumptions), which are important properties to characterize these three methods.

A. *HMM/DNN*

ASR deals with a sequence mapping from a T-length speech feature sequence, $X = \{\mathbf{x}_t \in \mathbb{R}^D | t = 1, \dots, T\}$, to an N-length word sequence, $W = \{w_n \in \mathcal{V} | n = 1, \dots, N\}$. Here, $\mathbf{x}_t$ is a D-dimensional speech feature vector (e.g., log Mel filterbanks) at frame t, and w_n is a word at position n in the vocabulary, $\mathcal{V}$.

ASR is mathematically formulated with Bayes decision theory, where the most probable word sequence, $\hat{W}$, is estimated among all possible word sequences, $\mathcal{V}*$, as follows:

$$\hat{W} = \arg \max_{W \in \mathcal{V}*} p(W|X). \tag{1}$$

Therefore, the main problem of ASR is how to obtain the posterior distribution $p(W|X)$.

The current main stream of ASR is based on a hybrid HMM/DNN [22], which uses Bayes' theorem and introduces the HMM state sequence, $S = \{s_t \in \{1, \ldots, J\} | t = 1, \ldots, T\}$, to factorize $p(W|X)$ into the following three distributions:

$$\arg\max_{W \in \mathcal{V}*} p(W|X)$$

$$= \arg\max_{W \in \mathcal{V}*} \sum_S p(X|S, W)p(S|W)p(W) \quad (2)$$

$$\approx \arg\max_{W \in \mathcal{V}*} \sum_S p(X|S)p(S|W)p(W). \quad (3)$$

The three factors, $p(X|S)$, $p(S|W)$, and $p(W)$, are the acoustic, lexicon, and language models, respectively. Eq. (3) is obtained by a conditional independence assumption (i.e., $p(X|S, W) \approx p(X|S)$), which is a reasonable assumption to simplify the dependency of the acoustic model.

1) Acoustic Model $p(X|S)$*:* $p(X|S)$ is further factorized by using a probabilistic chain rule and conditional independence assumption as follows:

$$p(X|S) = \prod_{t=1}^{T} p(\mathbf{x}_t|\mathbf{x}_1, , \ldots, \mathbf{x}_{t-1}, S) \quad (4)$$

$$\approx \prod_{t=1}^{T} p(\mathbf{x}_t|s_t) \propto \prod_{t=1}^{T} \frac{p(s_t|\mathbf{x}_t)}{p(s_t)}, \quad (5)$$

where the framewise likelihood function $p(\mathbf{x}_t|s_t)$ is replaced with the framewise posterior distribution $p(s_t|\mathbf{x}_t)/p(s_t)$ computed by powerful DNN classifiers by using the so-called pseudo-likelihood trick [22]. The conditional independence assumption in (5) is often regarded as too strong, since it does not consider any input and hidden state contexts. Therefore DNNs with long context features or recurrent neural networks are often used to mitigate this issue. To train the framewise posterior, we also require the provision of a framewise state alignment, s_t, as a target, which is often provided by an HMM/GMM system.

2) Lexicon Model $p(S|W)$*:* $p(S|W)$ is also factorized by using a probabilistic chain rule and conditional independence assumption (1st-order Markov assumption) as follows:

$$p(S|W) = \prod_{t=1}^{T} p(s_t|s_1, \ldots, s_{t-1}, W) \quad (6)$$

$$\approx \prod_{t=1}^{T} p(s_t|s_{t-1}, W). \quad (7)$$

This probability is represented by an HMM state transition given W. The conversion from W to HMM states is deterministically performed by using a pronunciation dictionary through a phoneme representation.

3) Language Model $p(W)$*:* Similarly, $p(W)$ is factorized by using a probabilistic chain rule and conditional independence assumption ($(m-1)$th-order Markov assumption) as an m-gram model, i.e.,

$$p(W) = \prod_{n=1}^{N} p(w_n|w_1, \ldots, w_{n-1}) \quad (8)$$

$$\approx \prod_{n=1}^{N} p(w_n|w_{n-m-1}, \ldots, w_{n-1}). \quad (9)$$

Although recurrent neural network language models (RNNLMs) can avoid this conditional independence assumption issue [23], it makes the decoding complex, and RNNLMs are often combined with m-gram language models based on a rescoring technique.

Thus, conventional HMM/DNN systems make the ASR problem formulated in (1) feasible by using factorization and conditional independence assumptions, at the cost of the five problems discussed in Section I.

B. Connectionist Temporal Classification (CTC)

The CTC formulation also follows from Bayes decision theory (Eq. (1)). Note that the CTC formulation uses an L-length letter sequence, $C = \{c_l \in \mathcal{U} | l = 1, \ldots, L\}$, with a set of distinct letters, $\mathcal{U}$. In addition, CTC additionally uses a "blank symbol," which explicitly denotes the letter boundary to handle the repetition of letter symbols. With the blank symbol, an augmented letter sequence, C', is defined as

$$C' = \{\texttt{<b>}, c_1, \texttt{<b>}, c_2, \texttt{<b>}, \ldots, c_L, \texttt{<b>}\} \quad (10)$$

$$= \{c'_l \in \mathcal{U} \cup \{\texttt{<b>}\} | l = 1, \ldots, 2L + 1\}. \quad (11)$$

In C', the augmented letter, c'_l, is always blank "<b>" when l is an odd number, whereas it is always a letter when l is an even number

Similar to Section II-A, by introducing a framewise letter sequence with an additional blank symbol, $Z = \{z_t \in \mathcal{U} \cup \{\texttt{<b>}\} | t = 1, \ldots, T\}$,[1] the posterior distribution, $p(C|X)$, is factorized as follows:

$$p(C|X) = \sum_Z p(C|Z, X)p(Z|X) \quad (12)$$

$$\approx \sum_Z p(C|Z)p(Z|X). \quad (13)$$

Similar to (3), CTC uses a conditional independence assumption to obtain (13) (i.e., $p(C|Z, X) \approx p(C|Z)$), which is a reasonable assumption to simplify the dependency of the CTC acoustic model, $p(Z|X)$, and CTC letter model, $p(C|Z)$.

1) CTC Acoustic Model: Similar to Section II-A1, $p(Z|X)$ is further factorized by using a probabilistic chain rule and conditional independence assumption as follows:

$$p(Z|X) = \prod_{t=1}^{T} p(z_t|z_1, \ldots, z_{t-1}, X) \quad (14)$$

$$\approx \prod_{t=1}^{T} p(z_t|X). \quad (15)$$

The framewise posterior distribution, $p(z_t|X)$, is conditioned on all inputs, X, and it is straightforward to be modeled by using

[1]In CTC and attention-based approaches, the sequence length of hidden states would be shorter than the original input sequence length (i.e., $|Z| < T$ in the CTC case) owing to the subsampling technique [10], [24]. However, the formulation in this paper retains the same index t and length T for simplicity.

bidirectional long short-term memory (BLSTM) [25], [26]:

$$p(z_t|X) = \text{Softmax}(\text{LinB}(\mathbf{h}_t)), \tag{16}$$

$$\mathbf{h}_t = \text{BLSTM}_t(X). \tag{17}$$

$\text{Softmax}(\cdot)$ is a sofmax activation function, and $\text{LinB}(\cdot)$ is a linear layer to convert the hidden vector, $\mathbf{h}_t$, to a $(|\mathcal{U}|+1)$ dimensional vector ($+1$ means a blank symbol introduced in CTC) with learnable matrix and bias vector parameters. $\text{BLSTM}_t(\cdot)$ accepts the full input sequence and output hidden vector at t.

2) CTC Letter Model: $p(Z|X)$ is rewritten by using Bayes' rule, a probabilistic chain rule, and a conditional independence assumption as follows:

$$p(C|Z) = \frac{p(Z|C)p(C)}{p(Z)} \tag{18}$$

$$= \prod_{t=1}^{T} p(z_t|z_1, \ldots, z_{t-1}, C)\frac{p(C)}{p(Z)} \tag{19}$$

$$\approx \prod_{t=1}^{T} p(z_t|z_{t-1}, C)\frac{p(C)}{p(Z)}, \tag{20}$$

where $p(z_t|z_{t-1}, C)$, $p(C)$, and $p(Z)$ are the state transition probability, letter-based language model, and state prior probability, respectively. CTC has a letter-based language model, $p(C)$, and by using a letter-to-word finite state transducer, we can also incorporate a word-based language model in CTC during decoding [16]. $\frac{1}{p(Z)}$ is not introduced in the original CTC formulation [15]. However, the theoretical justification and experimental effectiveness of this factor are shown in [27].

The state transition probability, $p(z_t|z_{t-1}, C)$, is represented with the augmented letter c'_l in (11) as follows:

$$p(z_t|z_{t-1}, C) \propto \begin{cases} 1 & z_t = c'_l \text{ and } z_{t-1} = c'_l \text{ for all possible } l \\ 1 & z_t = c'_l \text{ and } z_{t-1} = c'_{l-1} \text{ for all possible } l \\ 1 & z_t = c'_l \text{ and } z_{t-1} = c'_{l-2} \text{ for all possible even } l \\ 0 & \text{otherwise} \end{cases}. \tag{21}$$

In (21), the first case denotes the self transition, while the second case denotes the state transition. The third case is a special state transition from letter c'_{l-2} to c'_l by skipping "blank," where l is an even number, and c'_{l-2} and c'_l always denote a letter, as shown in (10). Note that in the implementation, these transition values are not normalized over z_t (i.e., not a probabilistic value) [16], [28], similar to the HMM state transition implementation [29].

With the state transition form in (21), it is obvious that CTC has the monotonic alignment property, i.e.,

$$\text{When } z_{t-1} = c'_m, \text{ Then } z_t = c'_l \text{ where } l \geq m. \tag{22}$$

This property is an important constraint for ASR, since the ASR sequence-to-sequence mapping must follow the monotonic alignment unlike machine translation. An HMM/DNN also satisfies this monotonic alignment property.

3) Objective: With Eqs. (15) and (20), the posterior, $p(C|X)$, is finally represented as

$$p(C|X) \approx \underbrace{\sum_Z \prod_{t=1}^{T} p(z_t|z_{t-1}, C)p(z_t|X)}_{\triangleq p_{\text{ctc}}(C|X)} \frac{p(C)}{p(Z)}. \tag{23}$$

Although (23) has to deal with a summation over all possible Z, it is efficiently computed by using dynamic programming (Viterbi/forward–backward algorithm) thanks to the Markov property. We also define the CTC objective function, $p_{\text{ctc}}(C|X)$, used in the later formulation, which does not usually include $p(C)/p(Z)$.

The CTC formulation is similar to that of an HMM/DNN, except that it applies Bayes' rule to $p(C|Z)$ instead of $p(W|X)$. As a result, CTC has three distribution components similar to the HMM/DNN case, i.e., the framewise posterior distribution, $p(z_t|X)$, transition probability, $p(z_t|z_{t-1}, C)$, and (letter-based) language model, $p(C)$. CTC also uses several conditional independence assumptions (Markov assumptions), and does not fully utilize the benefits of end-to-end ASR, as discussed in Section I. However, compared with HMM/DNN systems, CTC with the character output reprsentation still possesses the end-to-end benefits that it does not require pronunciation dictionaries and omits an HMM/GMM construction step.

C. Attention Mechanism

Compared with the HMM/DNN and CTC approaches, *the attention-based approach does not make any conditional independence assumptions*, and directly estimates the posterior, $p(C|X)$, on the basis of a probabilistic chain rule, as follows:

$$p(C|X) = \underbrace{\prod_{l=1}^{L} p(c_l|c_1, \ldots, c_{l-1}, X)}_{\triangleq p_{\text{att}}(C|X)}, \tag{24}$$

where $p_{\text{att}}(C|X)$ is an attention-based objective function. $p(c_l|c_1, \ldots, c_{l-1}, X)$ is obtained by

$$\mathbf{h}_t = \text{Encoder}(X), \tag{25}$$

$$a_{lt} = \begin{cases} \text{ContentAttention}(\mathbf{q}_{l-1}, \mathbf{h}_t) \\ \text{LocationAttention}(\{a_{l-1}\}_{t=1}^{T}, \mathbf{q}_{l-1}, \mathbf{h}_t) \end{cases}, \tag{26}$$

$$\mathbf{r}_l = \sum_{t=1}^{T} a_{lt}\mathbf{h}_t, \tag{27}$$

$$p(c_l|c_1, \ldots, c_{l-1}, X) = \text{Decoder}(\mathbf{r}_l, \mathbf{q}_{l-1}, c_{l-1}). \tag{28}$$

Eqs. (25) and (28) are encoder and decoder networks, respectively. a_{lt} in (26) is an attention weight, and represents the soft alignment of the hidden vector, $\mathbf{h}_t$, for each output, c_l, based on the weighted summation of hidden vectors to form the letter-wise hidden vector $\mathbf{r}_l$ in (27). $\text{ContentAttention}(\cdot)$ and $\text{LocationAttention}(\cdot)$ in (26) are based on a content-based attention mechanism with and without convolutional features [9], respectively. We will explain each module in more detail below.

1) Encoder Network: Eq. (25) converts the input feature vectors, X, into a framewise hidden vector, $\mathbf{h}_t$, and BLSTM is often used as an encoder network, i.e.,

$$\text{Encoder}(X) \triangleq \text{BLSTM}_t(X). \tag{29}$$

Note that the outputs are often subsampled to reduce the computational complexity of the encoder network [9], [10].

2) Content-Based Attention Mechanism: In (26), ContentAttention$(\cdot)$ is represented as follows:

$$e_{lt} = \mathbf{g}^\top \tanh(\text{Lin}(\mathbf{q}_{l-1}) + \text{LinB}(\mathbf{h}_t)), \tag{30}$$

$$a_{lt} = \text{Softmax}(\{e_{lt}\}_{t=1}^T). \tag{31}$$

$\mathbf{g}$ is a learnable vector parameter. $\{e_{lt}\}_{t=1}^T$ is a T-dimensional vector, i.e., $\{e_{lt}\}_{t=1}^T = [e_{l1}, e_{l2}, \ldots, e_{lT}]^\top$. $\tanh(\cdot)$ is a hyperbolic tangent activation function, and $\text{Lin}(\cdot)$ is a linear layer with learnable matrix parameters, but without bias vector parameters.

3) Location-aware Attention Mechanism: The content-based attention mechanism is extended to deal with a convolution (location-aware attention). When we use $\mathbf{a}_{l-1} = \{a_{l-1}\}_{t=1}^T = [a_{l-1,1}, \ldots, a_{l-1,T}]^\top$, LocationAttention$(\cdot)$ in (26) is represented as follows:

$$\{\mathbf{f}_t\}_{t=1}^T = \mathbf{K} * \mathbf{a}_{l-1}, \tag{32}$$

$$e_{lt} = \mathbf{g}^\top \tanh(\text{Lin}(\mathbf{q}_{l-1}) + \text{Lin}(\mathbf{h}_t) + \text{LinB}(\mathbf{f}_t)), \tag{33}$$

$$a_{lt} = \text{Softmax}(\{e_{lt}\}_{t=1}^T). \tag{34}$$

$*$ denotes one-dimensional convolution along the input feature axis, t, with the convolution parameter, $\mathbf{K}$, to produce the set of T features $\{\mathbf{f}_t\}_{t=1}^T$.

4) Decoder Network: The decoder network in (28) is another recurrent network conditioned on the previous output c_{l-1} and hidden vector $\mathbf{q}_{l-1}$, similar to an RNNLM, in addition to the letter-wise hidden vector, $\mathbf{r}_l$. We use the following unidirectional LSTM:

$$\text{Decoder}(\cdot) \triangleq \text{Softmax}(\text{LinB}(\text{LSTM}_l(\cdot))). \tag{35}$$

$\text{LSTM}_l(\cdot)$ is a unidirectional LSTM unit, which outputs the hidden vector $\mathbf{q}_l$ as follows:

$$\mathbf{q}_l = \text{LSTM}_l(\mathbf{r}_l, \mathbf{q}_{l-1}, c_{l-1}). \tag{36}$$

This LSTM accepts the concatenated vector of the letter-wise hidden vector, $\mathbf{r}_l$, and the one-hot representation of the previous output, c_{l-1}, as an input.

5) Objective: The training objective of the attention model is approximately computed from the sequence posterior $p_{\text{att}}(C|X)$ in (24) as follows:

$$p_{\text{att}}(C|X) \approx \prod_{l=1}^{L} p(c_l|c_1^*, \ldots, c_{l-1}^*, X) \triangleq p_{\text{att}}^*(C|X), \tag{37}$$

where c_l^* is the ground truth of the previous characters. This is the strong assumption of the attention-based approach that (37) corresponds to a combination of letter-wise objectives based on a simple multiclass classification with the conditional ground truth history $c_l^*, \ldots, c_{l-1}^*$ in each output, l, and does not fully consider a sequence-level objective, as pointed out by [10].

In summary, attention-based ASR does not explicitly separate each module, and potentially handles the all five issues presented in Section I. It implicitly combines acoustic, lexicon, and language models as encoder, attention, and decoder networks, which can be jointly trained as a single network. However, compared with an HMM/DNN and CTC, which has a reasonable monotonic alignment property, as discussed in Section II-B2, the attention mechanism does not maintain this constraint. The alignment is represented by a weighted sum over all frames, as shown in (27), and often provides irregular alignments. A major focus of this paper is to address this problem by proposing hybrid CTC/attention architectures.

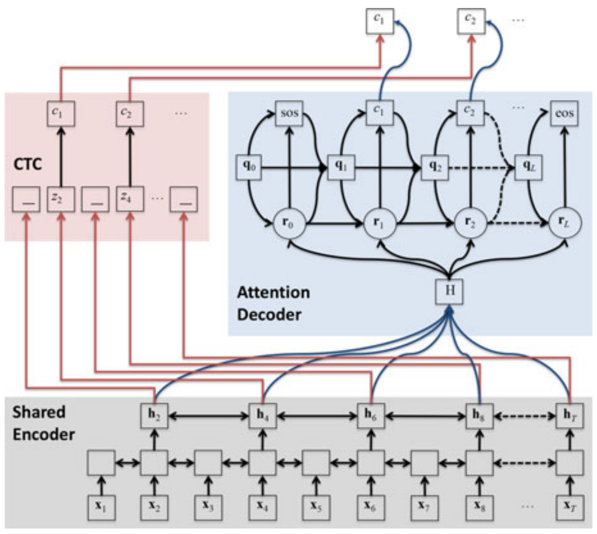

Fig. 1. Hybrid CTC/attention-based end-to-end architecture. The shared encoder is trained by both CTC and attention model objectives simultaneously. The shared encoder transforms our input sequence, $\{\mathbf{x}_t \cdots \mathbf{x}_T\}$, into the high level features, $H = \{\mathbf{h}_t \cdots \mathbf{h}_T\}$, and the attention decoder generates the letter sequence, $\{c_1 \cdots c_L\}$.

III. Hybrid CTC/Attention

This section explains our CTC/attention architecture, which utilizes both benefits of CTC and attention during the training and decoding steps in ASR.

A. Multiobjective Learning

The proposed training method uses a CTC objective function as an auxiliary task to train the attention model encoder within the multiobjective learning (MOL) framework [19]. Fig. 1 illustrates the overall architecture of the framework, where the same BLSTM is shared with the CTC and attention encoder networks (i.e., Eqs. (17) and (29), respectively). Unlike the sole attention model, the forward–backward algorithm of CTC can enforce a monotonic alignment between speech and label sequences during training. That is, rather than solely depending on data-driven attention methods to estimate the desired alignments in long sequences, the forward–backward algorithm in CTC helps to speed up the process of estimating the desired alignment. The objective to be maximized is a logarithmic linear combination of the CTC and attention objectives, i.e., $p_{\text{ctc}}(C|X)$ in (23) and $p_{\text{att}}^*(C|X)$ in (37):

$$\mathcal{L}_{\text{MOL}} = \lambda \log p_{\text{ctc}}(C|X) + (1-\lambda) \log p_{\text{att}}^*(C|X), \tag{38}$$

where the tunable parameter, λ, satisfies $0 \leq \lambda \leq 1$. Another advantage of (38) is that the attention objective is an approximated letter-wise objective, as discussed in Section II-C5, whereas the CTC objective is a sequence-level objective. Therefore, this multiobjective learning could also mitigate this approximation with the sequence-level CTC objective, in addition to helping the process of estimating the desired alignment. This multiobjective learning strategy in end-to-end ASR is also presented in [30], which combines segmental conditional random field (CRF) and CTC.

B. Joint Decoding

The inference step of our hybrid CTC/attention-based end-to-end speech recognition is performed by label synchronous decoding with a beam search similar to conventional attention-based ASR. However, we take the CTC probabilities into account to find a hypothesis that is better aligned to the input speech, as shown in Fig. 1. Hereafter, we describe the general attention-based decoding and conventional techniques to mitigate the alignment problem. Then, we propose joint decoding methods with a hybrid CTC/attention architecture.

1) Attention-Based Decoding in General: End-to-end speech recognition inference is generally defined as a problem to find the most probable letter sequence $\hat{C}$ given the speech input X, i.e.

$$\hat{C} = \arg\max_{C \in \mathcal{U}^*} \log p(C|X). \tag{39}$$

In attention-based ASR, $p(C|X)$ is computed by (24), and $\hat{C}$ is found by a beam search technique.

Let Ω_l be a set of partial hypotheses of the length l. At the beginning of the beam search, Ω_0 contains only one hypothesis with the starting symbol, <sos>. For $l = 1$ to L_{max}, each partial hypothesis in Ω_{l-1} is expanded by appending possible single letters, and the new hypotheses are stored in Ω_l, where L_{max} is the maximum length of the hypotheses to be searched. The score of each new hypothesis is computed in the log domain as

$$\alpha(h, X) = \alpha(g, X) + \log p(c|g_{l-1}, X), \tag{40}$$

where g is a partial hypothesis in Ω_{l-1}, c is a letter appended to g, and h is the new hypothesis such that $h = g \cdot c$. If c is a special symbol that represents the end of a sequence, <eos>, h is added to $\hat{\Omega}$ but not Ω_l, where $\hat{\Omega}$ denotes a set of complete hypotheses. Finally, $\hat{C}$ is obtained by

$$\hat{C} = \arg\max_{h \in \hat{\Omega}} \alpha(h, X). \tag{41}$$

In the beam search process, Ω_l is allowed to hold only a limited number of hypotheses with higher scores to improve the search efficiency.

Attention-based ASR, however, may be prone to include deletion and insertion errors (see Fig. 3 and related discussions) because of its flexible alignment property, which can attend to any portion of the encoder state sequence to predict the next label, as discussed in Section II-C. Since attention is generated by the decoder network, it may prematurely predict the end-of-sequence label, even when it has not attended to all of the encoder frames, making the hypothesis too short. On the other hand, it may predict the next label with a high probability by attending to the same portions as those attended to before. In this case, the hypothesis becomes very long and includes repetitions of the same label sequence.

2) Conventional Decoding Techniques: To alleviate the alignment problem, a length penalty term is commonly used to control the hypothesis length to be selected [9], [31]. With the length penalty, the decoding objective in (39) is changed to

$$\hat{C} = \arg\max_{C \in \mathcal{U}^*} \left\{ \log p(C|X) + \gamma |C| \right\}, \tag{42}$$

where $|C|$ is the length of sequence C, and γ is a tunable parameter. However, it is actually difficult to completely exclude hypotheses that are too long or too short even if γ is carefully tuned. It is also effective to control the hypothesis length by the minimum and maximum lengths to some extent, where the minimum and maximum are selected as fixed ratios to the length of the input speech. However, since there are exceptionally long or short transcripts compared to the input speech, it is difficult to balance saving such exceptional transcripts and preventing hypotheses with irrelevant lengths.

Another approach is the *coverage* term recently proposed in [14], which is incorporated in the decoding objective in (42) as

$$\hat{C} = \arg\max_{C \in \mathcal{U}^*} \left\{ \log p(C|X) + \gamma |C| + \eta \cdot \text{coverage}(C|X) \right\}, \tag{43}$$

where the coverage term is computed by

$$\text{coverage}(C|X) = \sum_{t=1}^{T} \left[\sum_{l=1}^{L} a_{lt} > \tau \right]. \tag{44}$$

η and τ are tunable parameters. The coverage term represents the number of frames that have received a cumulative attention greater than τ. Accordingly, it increases when paying close attention to some frames for the first time, but does not increase when paying attention again to the same frames. This property is effective for avoiding looping of the same label sequence within a hypothesis. However, the coverage term has no explicit mechanism for avoiding premature prediction of the end-of-sequence label, which makes the hypothesis too short and causes a lot of deletion errors. Moreover, it is still difficult to obtain a common parameter setting for γ, η, τ, and the optional min/max lengths so that they are appropriate for any speech data from different tasks.

3) Joint Decoding: Our hybrid CTC/attention approach combines the CTC and attention-based sequence probabilities in the inference step, as well as the training step. Suppose $p_{\text{ctc}}(C|X)$ in (23) and $p_{\text{att}}(C|X)$ in (24) are the sequence probabilities given by CTC and the attention model, respectively. The decoding objective is defined similarly to (38) as

$$\hat{C} = \arg\max_{C \in \mathcal{U}^*} \left\{ \lambda \log p_{\text{ctc}}(C|X) + (1 - \lambda) \log p_{\text{att}}(C|X) \right\}. \tag{45}$$

The CTC probability enforces a monotonic alignment that does not allow large jumps or looping of the same frames as discussed in Section II-B2. Furthermore, it can avoid premature prediction of the end-of-sequence label, which is not handled by the coverage term. Accordingly, it is possible to choose a hypothesis with a better alignment and exclude irrelevant hypotheses without relying on the coverage term, length penalty, or min/max lengths.

In the beam search process, the decoder needs to compute a score for each partial hypothesis using (40). However, it is nontrivial to combine the CTC and attention-based scores in the beam search, because the attention decoder performs it output-label-synchronously while CTC perfomrs it frame-synchronously. To incorporate the CTC probabilities in the hypothesis score, we propose two methods.

3) Rescoring: The first method is a two-pass approach, in which the first pass obtains a set of complete hypotheses using the beam search, where only the attention-based sequence probabilities are considered. The second pass rescores the complete hypotheses using the CTC and attention probabilities, where the CTC probabilities are obtained by the forward algorithm for CTC [15]. The rescoring pass obtains the final result according to

$$\hat{C} = \arg\max_{h\in\hat{\Omega}} \left\{\lambda\alpha_{\text{ctc}}(h, X) + (1-\lambda)\alpha_{\text{att}}(h, X)\right\}, \tag{46}$$

where

$$\begin{cases} \alpha_{\text{ctc}}(h, X) \triangleq \log p_{\text{ctc}}(h|X) \\ \alpha_{\text{att}}(h, X) \triangleq \log p_{\text{att}}(h|X) \end{cases} \tag{47}$$

3) One-Pass Decoding: The second method is one-pass decoding, in which we compute the probability of each partial hypothesis using CTC and an attention model. Here, we utilize the CTC prefix probability [32] defined as the cumulative probability of all label sequences that have h as their prefix:

$$p_{\text{ctc}}(h, \ldots|X) = \sum_{\nu\in(\mathcal{U}\cup\{\texttt{<eos>}\})^{+}} p_{\text{ctc}}(h\cdot\nu|X), \tag{48}$$

and we define the CTC score as

$$\alpha_{\text{ctc}}(h, X) \triangleq \log p_{\text{ctc}}(h, \ldots|X), \tag{49}$$

where ν represents all possible label sequences except the empty string. The CTC score cannot be obtained recursively as in (40), but it can be computed efficiently by keeping the forward probabilities over the input frames for each partial hypothesis. Then it is combined with $\alpha_{\text{att}}(h, X)$ using λ.

The beam search algorithm for one-pass decoding is shown in Algorithm 1. Ω_l and $\hat{\Omega}$ are initialized in lines 2 and 3 of the algorithm, which are implemented as queues that accept partial hypotheses of the length l and complete hypotheses, respectively. In lines 4–25, each partial hypothesis g in Ω_{l-1} is extended by each label c in the label set $\mathcal{U}$. Each extended hypothesis, h, is scored in line 11, where CTC and attention-based scores are obtained by $\alpha_{\text{ctc}}()$ and $\alpha_{\text{att}}()$. After that, if $c = \texttt{<eos>}$, the hypothesis h is assumed to be complete and stored in $\hat{\Omega}$ in line 13. If $c \neq \texttt{<eos>}$, h is stored in Ω_l in line 15, where the number of hypotheses in Ω_l is checked in line 16.

Algorithm 1: Joint CTC/attention One-pass Decoding.

```
 1: procedure OnePassBeamSearch ($X, L_{\max}$)
 2:   $\Omega_0 \leftarrow \{<\texttt{sos}>\}$
 3:   $\hat{\Omega} \leftarrow \emptyset$
 4:   for $l = 1 \ldots L_{\max}$ do
 5:     $\Omega_l \leftarrow \emptyset$
 6:     while $\Omega_{l-1} \neq \emptyset$ do
 7:       $g \leftarrow$ Head($\Omega_{l-1}$)
 8:       Dequeue($\Omega_{l-1}$)
 9:       for each $c \in \mathcal{U} \cup \{<\texttt{eos}>\}$ do
10:         $h \leftarrow g \cdot c$
11:         $\alpha(h) \leftarrow \lambda\alpha_{\text{ctc}}(h, X) + (1-\lambda)\alpha_{\text{att}}(h, X)$
12:         if $c = <\texttt{eos}>$ then
13:           Enqueue($\hat{\Omega}, h$)
14:         else
15:           Enqueue($\Omega_l, h$)
16:           if $|\Omega_l| > beamWidth$ then
17:             RemoveWorst($\Omega_l$)
18:           end if
19:         end if
20:       end for
21:     end while
22:     if EndDetect($\hat{\Omega}, l$) $=$ true then
23:       break                              ▷ exit for loop
24:     end if
25:   end for
26:   return $\arg\max_{C\in\hat{\Omega}} \alpha(C)$
27: end procedure
```

If the number exceeds the beam width, the hypothesis with the worst score in Ω_l, i.e.,

$$h_{\text{worst}} = \arg\min_{h\in\Omega_l} \alpha(h, X),$$

is removed from Ω_l by RemoveWorst() in line 17.

We can optionally apply an end detection technique to reduce the computation by stopping the beam search before l reaches $L_{\max}$. Function EndDetect($\hat{\Omega}, l$) in line 22 returns `true` if there is little chance of finding complete hypotheses with higher scores as l increases in the future. In our implementation, the function returns `true` if

$$\sum_{m=0}^{M-1} \left[\left\{\max_{h\in\hat{\Omega}:|h|=l-m} \alpha(h, X) - \max_{h'\in\hat{\Omega}} \alpha(h', X)\right\} < D_{\text{end}}\right] = M, \tag{50}$$

where D_{end} and M are predetermined thresholds.

This equation becomes `true` if scores of recently completed hypotheses are all small enough compared to the best score of all the completed hypotheses up to the present in the decoding process. In the summation of (50), the first maximum corresponds to the best score in the complete hypotheses recently generated, whose length $|h|$ is $l - m$, where $m = 0, \ldots, M-1$ (e.g., $M = 3$). The second maximum corresponds to the best score in all the complete hypotheses in $\hat{\Omega}$. The Iverson bracket $[\cdot]$ returns 1 if the difference between these maximum scores is smaller

Algorithm 2: CTC Label Sequence Score.

```
1: function $\alpha_{\text{ctc}}\ h, X$
2:     $g, c \leftarrow h$  ▷ split $h$ into the last label $c$ and the rest $g$
3:     if $c =$<eos> then
4:         return $\log\{\gamma_T^{(n)}(g) + \gamma_T^{(b)}(g)\}$
5:     else
6:         $\gamma_1^{(n)}(h) \leftarrow \begin{cases} p(z_1 = c|X) & \text{if } g = \text{<sos>} \\ 0 & \text{otherwise} \end{cases}$
7:         $\gamma_1^{(b)}(h) \leftarrow 0$
8:         $\Psi \leftarrow \gamma_1^{(n)}(h)$
9:         for $t = 2 \ldots T$ do
10:            $\Phi \leftarrow \gamma_{t-1}^{(b)}(g) + \begin{cases} 0 & \text{if last}(g) = c \\ \gamma_{t-1}^{(n)}(g) & \text{otherwise} \end{cases}$
11:            $\gamma_t^{(n)}(h) \leftarrow \big(\gamma_{t-1}^{(n)}(h) + \Phi\big) p(z_t = c|X)$
12:            $\gamma_t^{(b)}(h) \leftarrow \big(\gamma_{t-1}^{(b)}(h) + \gamma_{t-1}^{(n)}(h)\big) p(z_t = \text{<b>} |X)$
13:            $\Psi \leftarrow \Psi + \Phi \cdot p(z_t = c|X)$
14:        end for
15:        return $\log(\Psi)$
16:    end if
17: end function
```

than threshold D_{end} (e.g., $D_{\text{end}} = -10$), otherwise it returns 0. Hence, the summation results in M if all the differences are less than the threshold.

In line 11, the CTC and attention model scores are computed for each partial hypothesis. The attention score is easily obtained in the same manner as (40), whereas the CTC score requires a modified forward algorithm that computes it label-synchronously. The algorithm performs the function, $\alpha_{\text{ctc}}(h, X)$, as shown in Algorithm 2. Let $\gamma_t^{(n)}(h)$ and $\gamma_t^{(b)}(h)$ be the forward probabilities of the hypothesis, h, over time frames $1 \ldots t$, where the superscripts (n) and (b) denote different cases in which all CTC paths end with a nonblank or blank symbol, respecitively. Before starting the beam search, $\gamma_t^{(n)}()$ and $\gamma_t^{(b)}()$ are initialized for $t = 1, \ldots, T$ as

$$\gamma_t^{(n)}(\text{<sos>}) = 0, \tag{51}$$

$$\gamma_t^{(b)}(\text{<sos>}) = \prod_{\tau=1}^{t} \gamma_{\tau-1}^{(b)}(\text{<sos>}) \cdot p(z_\tau = \text{<b>} |X), \tag{52}$$

where we assume that $\gamma_0^{(b)}(\text{<sos>}) = 1$ and <b> is a blank symbol. Note that the time index t and input length T may differ from those of the input utterance X owing to the subsampling technique for the encoder [10], [24].

In Algorithm 2, hypothesis h is first split into the last label, c, and the rest, g, in line 2. If c is <eos>, it returns the logarithm of the forward probability assuming that h is a complete hypothesis in line 4. The forward probability of h is given by

$$p_{\text{ctc}}(h|X) = \gamma_T^{(n)}(g) + \gamma_T^{(b)}(g) \tag{53}$$

TABLE I
ASR TASKS

CHiME-4 [35]	# utterances	Length (h)
Training	8,738	18
Development	3,280	5.6
Evaluation	2,640	4.4
WSJ [33], [34]	**# utterances**	**Length (h)**
Training (WSJ0 si84)	7,138	15
Training (WSJ1 si284)	37,416	80
Development	503	1.1
Evaluation	333	0.7
CSJ [36]	**# utterances**	**Length (h)**
Training (100k)	100,000	147
Training (Academic)	157,022	236
Training (Full)	445,068	581
Evaluation (task 1)	1,288	1.9
Evaluation (task 2)	1,305	2.0
Evaluation (task 3)	1,389	1.3
HKUST [37]	**# utterances**	**Length (h)**
Training	193,387	167
Training (speed perturb.)	580,161	501
Development	4,000	4.8
Evaluation	5,413	4.9

according to the definition of $\gamma_t^{(n)}()$ and $\gamma_t^{(b)}()$. If c is not <eos>, it computes forward probabilities $\gamma_t^{(n)}(h)$ and $\gamma_t^{(b)}(h)$, and the prefix probability, $\Psi = p_{\text{ctc}}(h, \ldots |X)$, assuming that h is not a complete hypothesis. The initialization and recursion steps for those probabilities are described in lines 6–14. In this function, we assume that whenever we compute the probabilities, $\gamma_t^{(n)}(h)$, $\gamma_t^{(b)}(h)$ and Ψ, the forward probabilities $\gamma_t^{(n)}(g)$ and $\gamma_t^{(b)}(g)$ have already been obtained through the beam search process because g is a prefix of h such that $|g| < |h|$. Accordingly, the prefix and forward probabilities can be computed efficiently for each hypothesis, and partial hypotheses with irrelevant alignments can be excluded by the CTC score during the beam search. Thus, the one-pass search method hopefully reduces the number of search errors with less computation compared to the rescoring method.

IV. Experiments

We demonstrate our experiments using four different ASR tasks, as summarzied in Table I. The first part of the experiments used famous English clean speech corpora, WSJ1 and WSJ0 [33], [34], and a noisy speech corpus, CHiME-4 [35]. CHiME-4 was recorded using a tablet device in everyday environments: a cafe, a street junction, public transport, and a pedestrian area. The experiments with these corpora are designed to focus on the effectiveness of the multiobjective learning part (Section III-A) of our hybrid CTC/attention architecture with various learning configurations thanks to the relatively small sizes of these corpora.

The second part of the experiments scaled up the size of the corpora by using the Corpus of Spontaneous Japanese (CSJ) [36] and HKUST Mandarin Chinese conversational telephone speech recognition (HKUST) [37]. These experiments mainly show the

TABLE II
COMMON EXPERIMENTAL HYPERPARAMETERS

Parameter initialization	uniform distribution $[-0.1, 0.1]$
# of encoder BLSTM cells	320
# of encoder projection units	320
Encoder subsampling	2nd and 3rd bottom layers (skip every 2nd feature, yielding $4/T$)
# of decoder LSTM cells	320
Optimization	AdaDelta
Adadelta ρ	0.95
Adadelta ϵ	10^{-8}
Adadelta ϵ decaying factor	10^{-2}
Gradient norm clip threshold	5
Maximum epoch	15
Threshold to stop iteration	10^{-4}
Sharpening parameter γ	2
Location-aware # of conv. filters	10
Location-aware conv. filter widths	100
End detection length threshold D_{end}	$\log 1e^{-10}$
End detection score threshold M	3

TABLE III
CHARACTER ERROR RATES (CERS) FOR THE CLEAN CORPORA WSJ0 AND WSJ1, AND THE NOISY CORPUS CHIME-4

Model	CER (valid)	CER (eval)
WSJ1 SI284 (80h)	dev93	eval92
CTC	11.48	8.97
Attention (content-based)	13.68	11.08
Attention (+location-aware)	11.98	8.17
MOL ($\lambda = 0.2$)	**11.27**	**7.36**
MOL ($\lambda = 0.5$)	12.00	8.31
MOL ($\lambda = 0.8$)	11.71	8.45
WSJ0 SI84 (15h)	dev93	eval92
CTC	27.41	20.34
Attention (content-based)	28.02	20.06
Attention (+location-aware)	24.98	17.01
MOL ($\lambda = 0.2$)	**23.03**	**14.53**
MOL ($\lambda = 0.5$)	26.28	16.24
MOL ($\lambda = 0.8$)	32.21	21.30
CHiME-4 (18h)	dt05_real	et05_real
CTC	37.56	48.79
Attention (content-based)	43.45	54.25
Attention (+location-aware)	35.01	47.58
MOL ($\lambda = 0.2$)	**32.08**	**44.99**
MOL ($\lambda = 0.5$)	34.56	46.49
MOL ($\lambda = 0.8$)	35.41	48.34

effectiveness of our joint decoding, as discussed in Section III-B. The main reason for choosing these two languages is that these ideogram languages have relatively shorter lengths (i.e., L) for letter sequences than those in alphabet languages, which greatly reduces the computational complexities, and makes it easy to handle context information in a decoder network. Actually, our preliminary investigation shows that Japanese and Mandarin Chinese end-to-end ASR can be easily scaled up, and shows reasonable performance without using various tricks developed for large-scale English tasks.

Table II lists the common experimental hyperparameters among all experiments. The task-specific hyperparameters are described in each experimental section. This paper also strictly followed an end-to-end ASR concept, and did not use any pronunciation lexicon, language model, GMM/HMM, or DNN/HMM. Our hybrid CTC/attention architecture was implemented with Chainer [28].

A. WSJ and CHiME-4

As presented in Table I, the evaluation was performed for 1) "eval92" for WSJ0 and WSJ1 and 2) "et05_real_isolated_1ch_track" for CHiME-4, while hyperparameter selection was performed for 1) "dev93" for WSJ0 and WSJ1 and 2) "dt05_multi_isolated_1ch_track" for CHiME-4.

As input features, we used 40 mel-scale filterbank coefficients with their first- and second-order temporal derivatives to obtain a total of 120 feature values per frame. For the attention model, we used only 32 distinct labels: 26 characters, apostrophe, period, dash, space, noise, and sos/eos tokens. The CTC model used the blank instead of sos/eos, and our MOL model used both sos/eos and the blank. The encoder was a four-layer BLSTM with 320 cells in each layer and direction, and the linear projection layer with 320 units is followed by each BLSTM layer. The second and third bottom LSTM layers of the encoder read every second state feature in the network below, reducing the utterance length by a factor of four, i.e., $T/4$. The decoder was a one-layer unidirectional LSTM with 320 cells. The other experimental

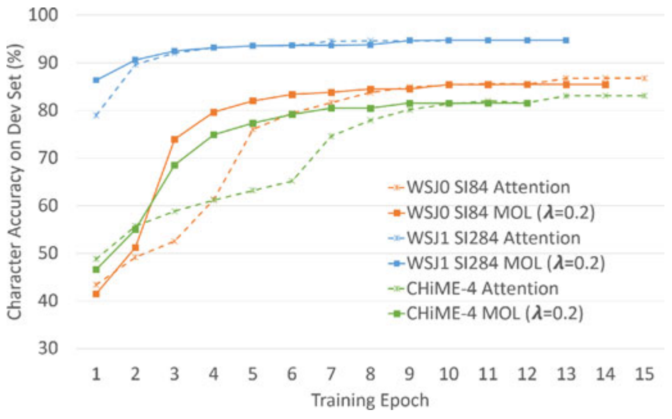

Fig. 2. Learning curves: location-aware attention model and MOL with $\lambda = 0.2$. Note that the approximated accuracies of the attention and our MOL were obtained given the ground truth history, as discussed in Section II-C5.

setup is summarized in Table II. For our MOL, we tested three different task weights λ: 0.2, 0.5, and 0.8.

For the decoding of the attention and MOL models, we used a conventional beam search algorithm similar to [38] with a beam size of 20 to reduce the computational cost. For CHiME-4, we manually set the minimum and maximum lengths of the output sequences to 0.1 and 0.18 times the input sequence lengths, respectively, and the length penalty γ in (42) was set to 0.3. For WSJ, the minimum and maximum lengths were set to 0.075 and 0.2 times the input sequence lengths, respectively, without a length penalty (i.e., $\gamma = 0$). For the decoding of the CTC model, we took the Viterbi sequence as a result.

The resutls in Table III show that our proposed model MOL significantly outperformed both CTC and the attention model with regards to the CER for both the noisy CHiME-4 and clean WSJ tasks. Our model showed relative improvements of

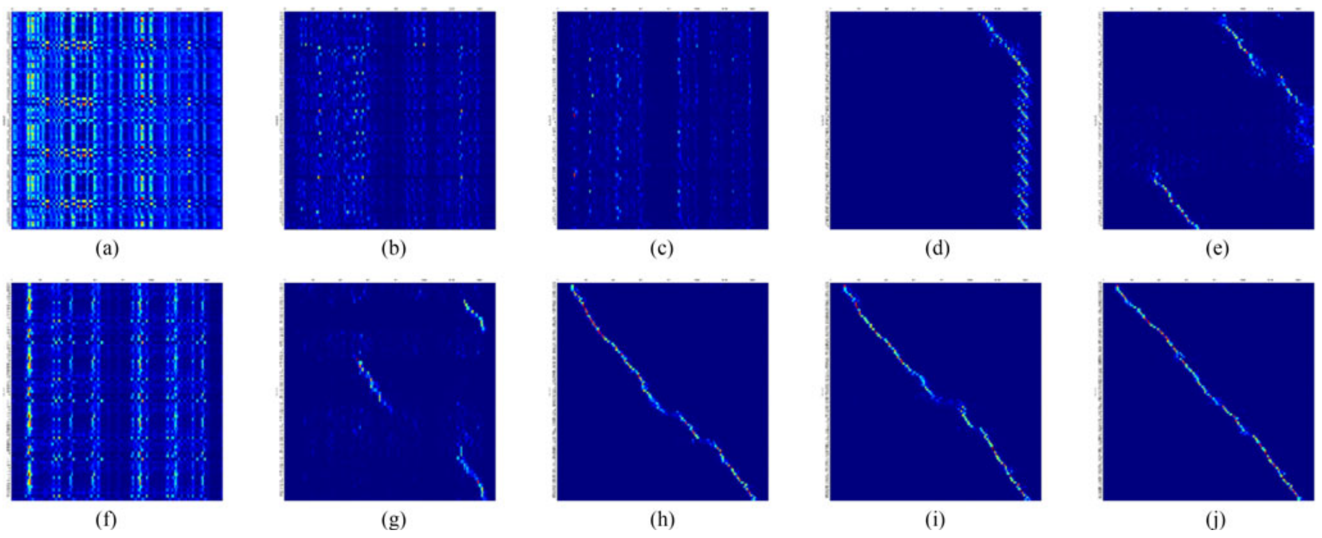

Fig. 3. Comparison of the speed in learning alignments between characters (y axis) and acoustic frames (x axis) between the location-based attention model (1st row) and our model MOL (2nd row) over the training epochs (1, 3, 5, 7, and 9). All alignments are for one manually chosen utterance (F05_442C020U_CAF_REAL) in the noisy CHiME-4 evaluation set. (a) Attention 1 epoch; (b) Attention 3 epoch; (c) Attention 5 epoch; (d) Attention 7 epoch; (e) Attention 9 epoch; (f) MOL 1 epoch; (g) MOL 3 epoch; (h) MOL 5 epoch; (i) MOL 7 epoch; (j) MOL 9 epoch.

6.0\hbox{–}8.4% and 5.4\hbox{–}14.6% for the validation and evaluation sets, respectively. We observed that our hybrid CTC/attention MOL achieved the best performance when we used $\lambda = 0.2$ for both the noisy CHiME-4 and clean WSJ tasks. As a reference, we also computed the word error rate (WER) of our model MOL ($\lambda = 0.2$), which scored 18.2% and was slightly better than the WER of the model in [39].

Apart from the CER improvements, MOL can also be very helpful in accelerating the learning of the desired alignment. Fig. 2 shows the learning curves of the character accuracy for the validation sets of CHiME-4, WSJ0 SI84, and WSJ1 SI284 over the training epochs. Note that the approximated accuracies of the attention and our MOL with $\lambda = 0.2$ were obtained given the ground truth history $c_1^*, \ldots, c_{l-1}^*$, as discussed in Section II-C5, and we cannot directly compare the absolute values of the validation character accuracy between MOL and the attention owing to the approximation. However, from the learning curve behaviors, we can argue that MOL training converged more quickly compared with the attention one.

Fig. 3 shows the attention alignments between characters and acoustic frames over the training epochs. We observed that our MOL learned the desired alignment in an early training stage, the 5th epoch, whereas the attention model could not learn the desired alignment even at the 9th epoch. This result indicates that the CTC loss guided the alignment to be monotonic in our MOL approach.

B. Corpus of Spontaneous Japanese (CSJ)

CSJ is a standard Japanese ASR task based on a collection of monologue speech data including academic lectures and simulated presentations. It has a total of 581 hours of training data and three types of evaluation data, where each evaluation task consists of 10 lectures (5 hours in total), as summarized in Table I. The experimental setup was similar to the previous English experiments, and we used 40 mel-scale filterbank coefficients with their first- and second-order temporal derivatives as an input feature vector. Further, we used a four-layer BLSTM and one-layer LSTM for the encoder and decoder networks, respectively. We used 3315 distinct labels including Kanji, two types of Japanese syllable characters (hiragana and katakana), alphabets, and Arabic numbers, with the "blank" symbol for CTC and the eos/sos symbol for the attention.

TABLE IV
CHARACTER ERROR RATES (CERs) FOR CONVENTIONAL ATTENTION AND THE PROPOSED HYBRID CTC/ATTENTION END-TO-END ASR FOR THE CORPUS OF SPONTANEOUS JAPANESE SPEECH RECOGNITION (CSJ) TASK

Model	Task1	Task2	Task3
Attention (147h)	20.1	14.0	32.7
MOL (147h)	16.9	12.7	28.9
Attention (236h)	16.3	12.2	24.7
MOL (236h)	13.4	10.1	21.5
Attention (581h)	11.4	7.9	9.0
MOL (581h)	10.5	7.6	8.3
MOL + joint decoding (rescoring, 581h)	10.1	7.1	7.8
MOL + joint decoding (one pass, 581h)	10.0	7.1	7.6
MOL-large + joint decoding (rescoring, 581h)	**8.4**	6.2	**6.9**
MOL-large + joint decoding (one pass, 581h)	**8.4**	**6.1**	**6.9**
GMM-discr. [40] (236h for AM, 581h for LM)	11.2	9.2	12.1
HMM/DNN [40] (236h for AM, 581h for LM)	9.0	7.2	9.6
CTC-syllable [27] (581 h)	9.4	7.3	7.5

Table IV first compares the CERs for conventional attention and MOL-based end-to-end ASR without joint decoding for various amounts of training data (147, 236, and 581 hours). λ in (38) was set to 0.1. When decoding, we manually set the minimum and maximum lengths of the output sequences to 0.1 and 0.5 times the input sequence lengths, respectively. The length penalty γ in (42) was set to 0.1. MOL significantly outperformed attention-based ASR in all evaluation tasks for all amounts of training data, which confirms the effectiveness of

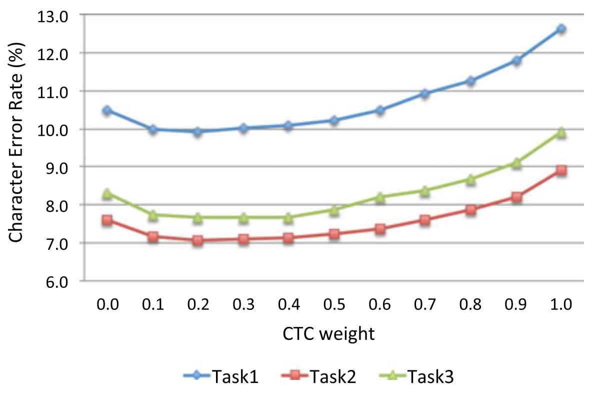

Fig. 4. Effect of CTC weight λ in (45) on the CSJ evaluation tasks.

TABLE V
CHARACTER ERROR RATES (CERS) FOR CONVENTIONAL ATTENTION AND THE PROPOSED HYBRID CTC/ATTENTION END-TO-END ASR FOR THE HKUST MANDARIN CHINESE CONVERSATIONAL TELEPHONE SPEECH RECOGNITION TASK

Model	Dev	Eval
Attention	40.3	37.8
MOL	38.7	36.6
Attention + coverage	39.4	37.6
MOL + coverage	36.9	35.3
MOL + joint decoding (rescoring)	35.9	34.2
MOL + joint decoding (one pass)	35.5	33.9
MOL-large (speed perturb.) + joint decoding (rescoring)	31.1	30.1
MOL-large (speed perturb.) + joint decoding (one pass)	**31.0**	**29.9**
MOL + CNN + LSTML (speed perturb.) + joint decoding (one pass) [41]	**29.1**	**28.0**
HMM/DNN	–	35.9
HMM/LSTM (speed perturb.)	–	33.5
CTC with language model [42]	–	34.8
HMM/TDNN, lattice-free MMI (speed perturb.) [24]	–	28.2

MOL in our hybrid CTC/attention architecture. The results in Table IV also show that the proposed joint decoding, described in Section III-B, further improved the performance without setting any search parameters (maximum and minimum lengths, length penalty), but only setting a weight parameter $\lambda = 0.1$ in (45), similar to the MOL case. Fig. 4 also compares the dependency of λ on the CER for the CSJ evaluation tasks, and shows that λ was not too sensitive to the performance if we set λ around the value we used for MOL (i.e., 0.1).

We also compare the performance of the proposed method of a larger network (a five-layer encoder network, MOL-large) with the conventional state-of-the-art techniques obtained by using linguistic resources including a morphological analyzer, pronunciation dictionary, and language model. The state-of-the-art CERs of the GMM discriminative training and HMM/DNN-sMBR (sMBR: state-level minimum Bayes risk) systems are obtained from the Kaldi recipe [40] and a system based on syllable-based CTC with MAP decoding [27]. The Kaldi recipe systems used academic lectures (236 h) for AM training and all training-data transcriptions for LM training. Note that since the amount of training data and experimental configurations of the proposed and reference methods were slightly different, it is difficult to compare the performance listed in the table directly. However, since the CERs of the proposed method were comparable to or better than those of the best reference results, we can state that the proposed method achieves state-of-the-art performance.

C. HKUST Mandarin Telephone Speech

HKUST Mandarin Chinese conversational telephone speech recognition [37] has 5 hours of recording for evaluation, and we extracted an additional 5 hours from the training data as a development set, and used the rest (167 hours) as a training set, as summarized in Table I. We used $\lambda = 0.5$ for training and decoding instead of 0.1 on the basis of our preliminary investigation, 80 mel-scale filterbank coefficients with pitch features as suggested in [42], and a five-layer BLSTM and two-layer LSTM for the encoder and decoder networks, respectively. The rest of the experimenal conditions were the same as those in Section IV-B and Table II. We used 3653 distinct labels with "blank" for CTC and eos/sos for the attention. For decoding, we also added the result from coverage-term-based decoding [14], as discussed in Section III-B ($\eta = 1.5, \tau = 0.5$, and $\gamma = -0.6$ for the attention model and $\eta = 1.0, \tau = 0.5$, and $\gamma = -0.1$ for MOL), since it was difficult to eliminate the irregular alignments during decoding by only tuning the maximum and minimum lengths and the length penalty (we set the minimum and maximum lengths of the output sequences to 0.0 and 0.1 times the input sequence lengths, respectively, and set $\gamma = 0.6$ in Table V).

The results in Table V show the effectiveness of MOL and joint decoding over the attention-based approach, especially showing a significant improvement for joint CTC/attention decoding. The results also show that our joint decoding "MOL+joint decoding (one pass)" works better than the coverage term "MOL+coverage," where the CER was reduced from 35.3% to 33.9% [2]. Similar to the CSJ experiments in Section IV-B, we did not use the length-penalty term or coverage term in joint decoding. This is an advantage of joint decoding over conventional approaches that require many tuning parameters. Moreover, Fig. 5 again shows that λ was not too sensitive to the performance if we set λ around the value we used for MOL (i.e., 0.5).

Finally, we generated more training data by linearly scaling the audio lengths by factors of 0.9 and 1.1 (speed perturb.). The final model achieved **29.9**% without using linguistic resources, which defeats moderate state-of-the-art systems including CTC-based methods[3].

[2]We further conducted an experiment of joint decoding with both CTC and the coverage term. Although we tuned decoding parameters including the length penalty, its CER was 34.2%, which was slightly worse than that of joint decoding, i.e., 33.9%

[3]Although the proposed method did not reach the performance obtained by a time delayed neural network (TDNN) with lattice-free sequence discriminative training, this method fully utilizes linguistic resources, including phonetic representations and phoneme-based language models, in the discriminative training [24]. Moreover, our recent work scored **28.0**%, and outperformed the lattice-free MMI result with advanced network architectures [41].

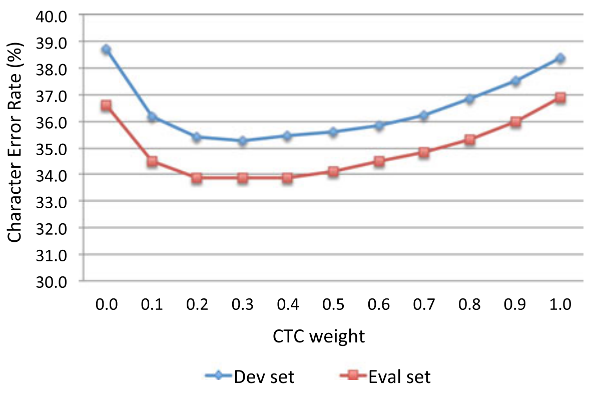

Fig. 5. Effect of CTC weight λ in (45) on the HKUST evaluation tasks.

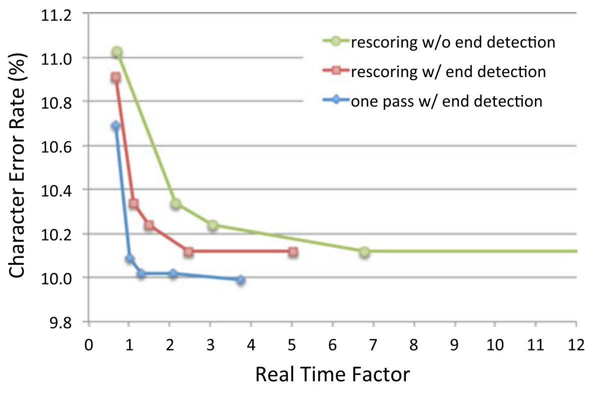

Fig. 6. RTF versus CER for the one-pass and rescoring methods for CSJ Task1.

D. Decoding Speed

We evaluated the speed of the joint decoding methods described in Section III-B3 for our hybrid CTC/attention architecture, where ASR decoding was performed with different beam widths of 1, 3, 5, 10, and 20, and the processing time and CER were measured using a computer with Intel(R) Xeon(R) processors, E5-2690 v3, 2.6 GHz. Although the processors were multicore CPUs and the computer also had GPUs, we ran the decoding program as a single-threaded process on a CPU to investigate its basic computational cost.

Figs. 6 and 7 show the relationships between the real-time factor (RTF) and the CER for the CSJ and HKUST tasks, respectively. We evaluated the rescoring method with and without end detection, and the one-pass method with end detection. For the both tasks, we can see that end detection successfully reduces the RTF without any accuracy degradation. Furthermore, the one-pass method achieves faster decoding with a lower CER than the rescoring method. With one-pass decoding, we achieved 1xRT with a small accuracy degradation, even if it was a single-threaded process on a CPU. However, the decoding process has not yet achieved real-time ASR since CTC and the attention mechanism need to access all of the frames of the input utterance even when predicting the first label. This is an essential problem of most end-to-end ASR approaches and will be solved in future work.

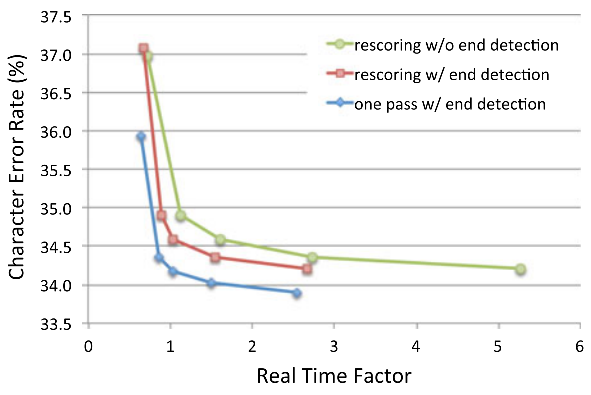

Fig. 7. RTF versus CER for the one-pass and rescoring methods for the HKUST test Set.

V. Summary and Discussion

This paper proposes end-to-end ASR by using hybrid CTC/attention architectures, which outperformed ordinary attention-based end-to-end ASR by solving the misalignment issues. This method does not require linguistic resources, such as a morphological analyzer, pronunciation dictionary, and language model, which are essential components of conventional Japanese and Mandarin Chinese ASR systems. Nevertheless, the method achieved comparable performance to state-of-the-art conventional systems for the CSJ and HKUST tasks. In addition, the proposed method does not require GMM/HMM construction for the initial alignments, DNN pre-training, lattice generation for sequence discriminative training, complex search during decoding (e.g., an FST decoder or a lexical-tree-search-based decoder). Thus, the method greatly simplifies the ASR building process, reducing code size and complexity. Currently, training takes 7–9 days using a single GPU to train the network with full training data (581 hours) for the CSJ task, which is comparable to the entire training time of the conventional state-of-the-art system owing to simplification of the building process.

Future work will apply this technique to the other languages including English, where we have to solve the issue of long sequence lengths, which requires a large computational cost and makes it difficult to train a decoder network. Actually, recent sequence-to-sequence studies have handled this issue by using a subword unit (concatenating several letters to form a new subword unit) [13], [43], which would be a promising direction for our end-to-end ASR. Another future work is to make use of existing conventional HMM/DNN when it is available apart from an end-to-end concept. It would be interesting to combine conventional HMM/DNN instead of or in addition to CTC in our framework (e.g., as another training objective) since they are complementary. Further investigation of CTC usage in training and decoding is also an interesting direction for future work. We could compare different cases of CTC usage, for example, the case when CTC is used only for pre-training the encoder of the attention model and the case when CTC is used only for decoding but not for training.

ACKNOWLEDGMENT

The author would like to thank J. Nishitoba at Retrieva, Inc. and S. Hido at Preferred Networks, Inc. for their valuable discussions and comments on the applications of Chainer to (end-to-end) speech recognition and also N. Kanda at Hitachi, Ltd. for providing the CER results of baseline Japanese systems.

REFERENCES

[1] F. Jelinek, "Continuous speech recognition by statistical methods," *Proc. IEEE*, vol. 64, no. 4, pp. 532–556, Apr. 1976.

[2] G. Hinton *et al.*, "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *IEEE Signal Process. Mag.*, vol. 29, no. 6, pp. 82–97, Nov. 2012.

[3] T. Kudo, K. Yamamoto, and Y. Matsumoto, "Applying conditional random fields to Japanese morphological analysis." in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2004, vol. 4, pp. 230–237.

[4] S. Bird, "NLTK: The natural language toolkit," in *Proc. Joint Conf. Int. Committee Comput. Linguist. Assoc. Comput. Linguist. Interact. Presentat. Sessions*, 2006, pp. 69–72.

[5] M. Mohri, "Finite-state transducers in language and speech processing," *Comput. Linguist.*, vol. 23, no. 2, pp. 269–311, 1997.

[6] T. Hori and A. Nakamura, *Speech Recognition Algorithms Using Weighted Finite-state Transducers*. Williston, VT, USA: Morgan & Claypool, 2013.

[7] J. Chorowski, D. Bahdanau, K. Cho, and Y. Bengio, "End-to-end continuous speech recognition using attention-based recurrent nn: First results," in *NIPS 2014 Workshop Deep Learning*, Dec. 2014.

[8] A. Graves and N. Jaitly, "Towards end-to-end speech recognition with recurrent neural networks," in *Proc. Int. Conf. Mach. Learn.*, 2014, pp. 1764–1772.

[9] J. K. Chorowski, D. Bahdanau, D. Serdyuk, K. Cho, and Y. Bengio, "Attention-based models for speech recognition," in *Proc. Adv. Neural Inf. Process. Syst.*, 2015, pp. 577–585.

[10] W. Chan, N. Jaitly, Q. V. Le, and O. Vinyals, "Listen, attend and spell," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2016, pp. 4960–4964.

[11] L. Lu, X. Zhang, and S. Renals, "On training the recurrent neural network encoder-decoder for large vocabulary end-to-end speech recognition," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2016, pp. 5060–5064.

[12] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," in *Proc. Int. Conf. Learn. Representations*, 2015.

[13] Y. Wu *et al.*, "Google's neural machine translation system: Bridging the gap between human and machine translation," arXiv:1609.08144, 2016.

[14] J. Chorowski and N. Jaitly, "Towards better decoding and language model integration in sequence to sequence models," in *Proc. Interspeech*, 2017, pp. 532–527.

[15] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, "Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks," in *Proc. Int. Conf. Mach. Learn.*, 2006, pp. 369–376.

[16] Y. Miao, M. Gowayyed, and F. Metze, "EESEN: End-to-end speech recognition using deep RNN models and WFST-based decoding," in *Proc. IEEE Workshop Autom. Speech Recognit. Understanding*, 2015, pp. 167–174.

[17] D. Amodei *et al.*, "Deep speech 2: End-to-end speech recognition in English and Mandarin," *Int. Conf. Mach. Learn.*, 2016, pp. 173–182.

[18] H. Soltau, H. Liao, and H. Sak, "Neural speech recognizer: Acoustic-to-word LSTM model for large vocabulary speech recognition," in *Proc. Interspeech*, 2017, pp. 3707–3711.

[19] S. Kim, T. Hori, and S. Watanabe, "Joint CTC-attention based end-to-end speech recognition using multi-task learning," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2017, pp. 4835–4839.

[20] T. Hori, S. Watanabe, and J. R. Hershey, "Joint CTC/attention decoding for end-to-end speech recognition," in *Proc. Assoc. Comput. Linguist.*, 2017, pp. 518–529.

[21] N. Xue *et al.*, "Chinese word segmentation as character tagging," *Comput. Linguist. Chin. Lang. Process.*, vol. 8, no. 1, pp. 29–48, 2003.

[22] H. Bourlard and N. Morgan, *Connectionist Speech Recognition: A Hybrid Approach*. Norwell, MA, USA: Kluwer, 1994.

[23] T. Mikolov, M. Karafiát, L. Burget, J. Cernocký, and S. Khudanpur, "Recurrent neural network based language model," in *Proc. Interspeech*, 2010, pp. 1045–1048.

[24] D. Povey *et al.*, "Purely sequence-trained neural networks for ASR based on lattice-free MMI," in *Proc. Interspeech*, 2016, pp. 2751–2755.

[25] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.

[26] A. Graves, N. Jaitly, and A.-R. Mohamed, "Hybrid speech recognition with deep bidirectional LSTM," in *Proc. IEEE Workshop Autom. Speech Recognit. Understanding*, 2013, pp. 273–278.

[27] N. Kanda, X. Lu, and H. Kawai, "Maximum a posteriori based decoding for CTC acoustic models," in *Proc. Interspeech*, 2016, pp. 1868–1872.

[28] S. Tokui, K. Oono, S. Hido, and J. Clayton, "Chainer: A next-generation open source framework for deep learning," in *Proc. Workshop Mach. Learn. Syst., Annu. Conf. Neural Inf. Process. Syst.*, 2015.

[29] D. Povey *et al.*, "The Kaldi speech recognition toolkit," in *Proc. IEEE Workshop Autom. Speech Recognit Understanding*, 2011.

[30] L. Lu, L. Kong, C. Dyer, and N. A. Smith, "Multi-task learning with CTC and segmental CRF for speech recognition," in *Proc. Interspeech*, 2017, pp. 954–958.

[31] D. Bahdanau, J. Chorowski, D. Serdyuk, P. Brakel, and Y. Bengio, "End-to-end attention-based large vocabulary speech recognition," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, 2016, pp. 4945–4949.

[32] A. Graves, "Supervised sequence labelling with recurrent neural networks," Ph.D. dissertation, Dept. Informat., Technische Univ. München, Munich, Germany, 2008.

[33] L. D. Consortium, *CSR-II (WSJ1) Complete*, vol. LDC94S13A. Philadelphia, PA, USA: Linguistic Data Consortium, 1994.

[34] J. Garofalo, D. Graff, D. Paul, and D. Pallett, *CSR-I, (WSJ0) Complete*, vol. LDC93S6A. Philadelphia, PA, USA: Linguistic Data Consortium, 2007.

[35] E. Vincent, S. Watanabe, A. A. Nugraha, J. Barker, and R. Marxer, "An analysis of environment, microphone and data simulation mismatches in robust speech recognition," in *Proc. Comput. Speech Lang.*, 2017, pp. 535–557.

[36] K. Maekawa, H. Koiso, S. Furui, and H. Isahara, "Spontaneous speech corpus of Japanese," in *Proc. Int. Conf. Lang. Resour. Eval.*, 2000, vol. 2, pp. 947–952.

[37] Y. Liu, P. Fung, Y. Yang, C. Cieri, S. Huang, and D. Graff, "HKUST/MTS: A very large scale Mandarin telephone speech corpus," in *Proc. Chin. Spoken Lang. Process.*, 2006, pp. 724–735.

[38] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 3104–3112.

[39] D. Bahdanau, J. Chorowski, D. Serdyuk, P. Brakel, and Y. Bengio, "End-to-end attention-based large vocabulary speech recognition," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2016, pp. 4945–4949.

[40] T. Moriya, T. Shinozaki, and S. Watanabe, "Kaldi recipe for Japanese spontaneous speech recognition and its evaluation," in *Proc. Autumn Meet. Acoust. Soc. Japan*, 2015, pp. 155–156.

[41] T. Hori, S. Watanabe, Y. Zhang, and W. Chan, "Advances in joint CTC-attention based end-to-end speech recognition with a deep CNN encoder and RNN-LM," in *Proc. Interspeech*, 2017, pp. 949–953.

[42] Y. Miao, M. Gowayyed, X. Na, T. Ko, F. Metze, and A. Waibel, "An empirical exploration of CTC acoustic models," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, 2016, pp. 2623–2627.

[43] W. Chan, Y. Zhang, Q. Le, and N. Jaitly, "Latent sequence decompositions," in *Proc. Int. Conf. Learn. Representations*, 2017.

Shinji Watanabe received the Ph.D. degree from Waseda University, Tokyo, Japan, in 2006. He is currently an Associate Research Professor at Johns Hopkins University, Baltimore, MD, USA. From 2001 to 2011, he was a Research Scientist in NTT Communication Science Laboratories, Kyoto, Japan. From January to March in 2009, he was a Visiting Scholar in Georgia Institute of Technology, Atlanta, GA, USA. From 2012 to 2017, he is a Senior Principal Research Scientist at Mitsubishi Electric Research Laboratories (MERL), Cambridge, MA, USA. He has published more than 100 papers in journals and conferences, and received several awards including the best paper award from the IEICE in 2003. His research interests include Bayesian machine learning and speech and spoken language processing.

He worked an Associate Editor of the IEEE TRANSACTIONS ON AUDIO, SPEECH, AND LANGUAGE PROCESSING, and is a member of several committees including the IEEE Signal Processing Society Speech and Language Technical Committee.

Takaaki Hori received the B.E. and M.E. degrees in electrical and information engineering from Yamagata University, Yonezawa, Japan, in 1994 and 1996, respectively, and the Ph.D. degree in system and information engineering from Yamagata University, Yamagata, Japan, in 1999. From 1999 to 2015, he had been engaged in researches on speech recognition and spoken language understanding in the Cyber Space Laboratories and Communication Science Laboratories, Nippon Telegraph and Telephone (NTT) Corporation, Japan. He was a Visiting Scientist in Massachusetts Institute of Technology from 2006 to 2007. Since 2015, he has been a Senior Principal Research Scientist in Mitsubishi Electric Research Laboratories, Cambridge, MA, USA. He has authored more than 90 peer-reviewed papers in speech and language research fields. He received the 24th TELECOM System Technology Award from the Telecommunications Advancement Foundation in 2009, the IPSJ Kiyasu Special Industrial Achievement Award from the Information Processing Society of Japan in 2012, and the 58th Maejima Hisoka Award from Tsushinbunka Association in 2013.

Suyoun Kim (S'15) received the M.S. degree in language technologies from Carnegie Mellon University, Pittsburgh, PA, USA, in 2014. She is currently working toward the Ph.D. degree at Carnegie Mellon University, Pittsburgh, PA, USA, since 2014. She was a research intern at Speech & Audio Lab., Mitsubishi Electric Research Laboratories, Cambridge, MA, USA, during the summer 2016. Her research interests include machine learning, deep learning, and spoken language processing.

John R. Hershey has been a Senior Principle Research Scientist and leader of the speech and audio team at Mitsubishi Electric Research Laboratories (MERL), Cambridge, MA, USA, since 2010. Prior to joining MERL, he spent five years at IBM's T.J. Watson Research Center Yorktown Heights, NY, USA, and led the Noise Robust Speech Recognition team. He also spent a year as a Visiting Researcher in the Speech Group, Microsoft Research, after receiving his doctorate from University of California San Diego, La Jolla, CA, USA. Over the years, he has contributed to more than 100 publications and more than 30 patents in the areas of machine perception, speech processing, speech recognition, and natural language understanding.

Tomoki Hayashi (S'16) received the B.E. degree in engineering and M.E. degree in information science from Nagoya University, Nagoya, Japan, in 2013 and 2015, respectively. He is currently working toward the Ph.D. degree in Nagoya University. His research interest include statistical speech and audio signal processing. He received the Acoustical Society of Japan 2014 Student Presentation Award. He is a student member of the Acoustical Society of Japan.