End-to-End Speech Recognition: A Survey

Rohit Prabhavalkar, *Member, IEEE*, Takaaki Hori, *Senior Member, IEEE*, Tara N. Sainath, *Fellow, IEEE*, Ralf Schlüter, *Senior Member, IEEE*, and Shinji Watanabe, *Fellow, IEEE*

***Abstract*—In the last decade of automatic speech recognition (ASR) research, the introduction of deep learning has brought considerable reductions in word error rate of more than 50% relative, compared to modeling without deep learning. In the wake of this transition, a number of all-neural ASR architectures have been introduced. These so-called *end-to-end* (E2E) models provide highly integrated, completely neural ASR models, which rely strongly on general machine learning knowledge, learn more consistently from data, with lower dependence on ASR domain-specific experience. The success and enthusiastic adoption of deep learning, accompanied by more generic model architectures has led to E2E models now becoming the prominent ASR approach. The goal of this survey is to provide a taxonomy of E2E ASR models and corresponding improvements, and to discuss their properties and their relationship to classical hidden Markov model (HMM) based ASR architectures. All relevant aspects of E2E ASR are covered in this work: modeling, training, decoding, and external language model integration, discussions of performance and deployment opportunities, as well as an outlook into potential future developments.**

***Index Terms*—End-to-end, automatic speech recognition.**

I. Introduction

The classical[1] statistical architecture decomposes an automatic speech recognition (ASR) system into four main components: acoustic feature extraction from speech audio signals, acoustic modeling, language modeling and search based on Bayes' decision rule [1], [2], [3]. Classical acoustic modeling is based on hidden Markov models (HMMs) to account for speaking rate variation. Within the classical approach, deep learning has been introduced into acoustic and language modeling. In acoustic modeling, deep learning has replaced Gaussian mixture distributions (hybrid HMM [4], [5]) or augmented the acoustic feature set (e.g., non-linear discriminant/tandem approach [6], [7]). In language modeling, deep learning has replaced count-based approaches [8], [9], [10]. However, in these early attempts at introducing deep learning, the classical ASR architecture was unmodified. Classical state-of-the-art ASR systems today are composed of many separate components and knowledge sources: especially speech signal preprocessing; methods for robustness with respect to recording conditions; phoneme inventories and pronunciation lexica; phonetic clustering; handling of out-of-vocabulary words; various methods for adaptation/normalization; elaborate training schedules with different objectives including sequence discriminative training, etc. The potential of deep learning, on the other hand, initiated successful approaches to integrate formerly separate modeling steps, e.g., by integrating speech signal pre-processing and feature extraction into acoustic modeling [11], [12].

More consequently, the introduction of deep learning to ASR also initiated research to replace classical ASR architectures based on hidden Markov models (HMM) with more integrated joint neural network model structures [13], [14], [15], [16]. These ventures might be seen as trading specific speech processing models for more generic machine learning approaches to sequence-to-sequence processing – akin to how statistical approaches to natural language processing have come to replace more linguistically oriented models. For these all-neural approaches recently the term *end-to-end* (E2E) [14], [17], [18], [19] has been established. Therefore, first of all an attempt to define the term *end-to-end* in the context of ASR is due in this survey. According to the Cambridge Dictionary, the adjective "end-to-end" is defined as: "including all the stages of a process" [20]. We therefore propose the following definition of end-to-end ASR: *an integrated ASR model that enables joint training from scratch; avoids separately obtained knowledge sources; and, provides single-pass recognition consistent with the objective to optimize the task-specific evaluation measure, i.e., usually label (word, character, subword, etc.) error rate*. While this definition suffices for the present discussion, we note that such an idealized definition hides many nuances involved in the term E2E and lacks distinctiveness; we elaborate on some of these nuances in Section II to discuss the various connotations of the term E2E in the context of ASR.

What are potential benefits of E2E approaches to ASR? The primary objective when developing an ASR systems is to minimize the expected word error rate; secondary objectives are to reduce time and memory complexity of the resulting decoder, and – assuming a constrained development budget – genericity, and ease of modeling. First of all, an integrated ASR system, defined

Manuscript received 21 February 2023; revised 2 September 2023; accepted 5 October 2023. Date of publication 30 October 2023; date of current version 16 November 2023. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Kai Yu. *(Corresponding author: Rohit Prabhavalkar.)*

Rohit Prabhavalkar is with Google LLC., Mountain View, CA 94043 USA (e-mail: prabhavalkar@google.com).

Takaaki Hori is with Apple Inc., Cambridge, MA 02142 USA (e-mail: thori@ieee.org).

Tara N. Sainath is with Google LLC., New York, NY 10011 USA (e-mail: tsainath@google.com).

Ralf Schlüter is with Lehrstuhl Informatik 6 - Computer Science Department, RWTH Aachen University, 52074 Aachen, Germany (e-mail: schlueter@cs.rwth-aachen.de).

Shinji Watanabe is with Carnegie Mellon University, Pittsburgh, PA 15213 USA (e-mail: shinjiw@ieee.org).

Digital Object Identifier 10.1109/TASLP.2023.3328283

[1]The term "classical" here refers to the former, long-term, state-of-the-art ASR architecture based on the decomposition into acoustic and language model, and with acoustic modeling based on hidden Markov models.

in terms of a single neural network structure supports genericity of modeling and may allow for faster development cycles when building ASR systems for new languages or domains. Similarly, ASR models defined by a single neural network structure may become more 'lean' compared to classical modeling, with a simpler decoding process, obviating the need to integrate separate models. The resulting reduction in memory footprint and power consumption supports embedded ASR applications [21], [22]. Furthermore, end-to-end joint training may help to avoid spurious optima from intermediate training stages. Avoiding secondary knowledge sources like pronunciation lexica may be helpful for languages/domains where such resources are not easily available. Also, secondary knowledge sources may themselves be erroneous; avoiding these may improve models trained directly from data, provided that sufficient amounts of task-specific training data are available.

With the current surge of interest in E2E ASR models and an increasing diversity of corresponding work, the authors of this review think it is time to provide an overview of this rapidly evolving domain of research. The goal of this survey is to provide an in-depth overview of the current state of research on E2E ASR systems, covering all relevant aspects of E2E ASR, with a contrastive discussion of the different E2E and classical ASR architectures.

This survey of E2E speech recognition is structured as follows. Section II discusses the nuances in the term E2E as it applies to ASR. Section III describes the historical evolution of E2E speech recognition, with specific focus on the input-output alignment and an overview of prominent E2E ASR models. Section IV discusses improvements of the basic E2E models, including E2E model combination, training loss functions, context, encoder/decoder structures and endpointing. Section V provides an overview of E2E ASR model training. Decoding algorithms for the different E2E approaches are discussed in Section VI. Section VII discusses the role and integration of (separate) language models in E2E ASR. Section VIII reviews experimental comparisons of the different E2E as well as classical ASR approaches. Section IX provides an overview of applications of E2E ASR. Section X investigates future directions of E2E research in ASR, before concluding in Section XI. Finally, we note that this survey paper also includes comparative discussions between novel E2E models and classical HMM-based ASR approaches in terms of various aspects; most sections end with a summarization of the relationship between E2E models and HMM-based ASR approaches in relation to the topics covered within the respective sections.

II. Distinctiveness of the Term E2E

As noted in Section I the term E2E provides an idealized definition of ASR systems, and can benefit from a more detailed discussion based on the following perspectives.

a) Joint Modeling: In terms of ASR, the E2E property can be interpreted as considering all components of an ASR system jointly as a single computational graph. Even more so, the common understanding of E2E in ASR is that of a single joint modeling approach that does not necessarily distinguish separate components, which may also mean dropping the classical separation of ASR into an acoustic model and a language model. However, in practice E2E ASR systems are often combined with external language models trained on text-only data, which weakens the end-to-end nature of the system to some extent.

b) Joint Training: In terms of model training, E2E can be interpreted as estimating all parameters, of all components of a model jointly using a single objective function that is consistent with the task at hand, which in case of ASR means minimizing the expected word error rate.[2] However, the term lacks distinctiveness here, as classical and/or modular ASR model architectures also support joint training with a single objective.

c) Training from Scratch: The E2E property can also be interpreted with respect to the training process itself, by requiring training *from scratch*, avoiding external knowledge like prior alignments or initial models pre-trained using different criteria or knowledge sources. However, note that pre-training and fine-tuning strategies are also relevant, if the model has explicit modularity, including self-supervised learning [25] or joint training of front-end and speech recognition models [26]. Especially in case of limited amounts of target task training data, utilizing large pretrained models is important to obtain performant E2E ASR systems.

d) Avoiding Secondary Knowledge Sources: For ASR, standard secondary knowledge sources are pronunciation lexica and phoneme sets, as well as phonetic clustering, which in classical state-of-the-art ASR systems usually is based on classification and regression trees (CART) [27]. Secondary knowledge sources and separately trained components may introduce errors, might be inconsistent with the overall training objective and/or may generate additional cost. Therefore, in an E2E approach, these would be avoided. Standard joint training of an E2E model requires using a single kind of training data, which in case of ASR would be transcribed speech audio data. However, in ASR often even larger amounts of text-only data, as well as optional untranscribed speech audio are available. One of the challenges of E2E modeling therefore is how to take advantage of text-only and audio-only data jointly without introducing secondary (pretrained) models and/or training objectives [28], [29].

e) Direct Vocabulary Modeling: Avoiding pronunciation lexica and corresponding subword units leave E2E recognition vocabularies to be derived from whole word or character representations. Whole word models [30], according to Zipf's law [31], would require unrealistically high amounts of transcribed training data for large vocabularies, which might not be attainable for many tasks. On the other hand, methods to generate subword vocabularies based on characters, like the currently popular byte pair encoding (BPE) approach [32], might be seen as secondary approaches outside the E2E objective, even more so if acoustic data is considered for subword derivation [33], [34], [35], [36].

[2] Note that this does not necessarily require Bayes Risk training, as standard training criteria like cross entropy, maximum mutual information and maximum likelihood in case of classical ASR models asymptotically guarantee optimal performance in the sense of Bayes decision rule, also [23], [24].

f) Generic Modeling: Finally, E2E modeling also requires genericity of the underlying modeling: task-specific constraints are learned completely from data, in contrast to task-specific knowledge which influences the modeling of the system architecture in the first place. For example, the monotonicity constraint in ASR may be learned completely from data in an end-to-end fashion (e.g., in attention-based approaches [16]), or it may directly be implemented, as in classical HMM structures. However, model constraints may be considered by way of regularization in E2E ASR model training, and can thus provide an alternative way to introduce task-specific knowledge.

g) Single-Pass Search: In terms of the recognition/search problem, the E2E property can be interpreted as integrating all components (models, knowledge sources) of an ASR system before coming to a decision. This is in line with Bayes' decision rule, which exactly requires a single global decision integrating all available knowledge sources, which is supported by both classical ASR models as well as E2E models. On the other hand, multipass search is not only exploited by classical ASR models, but also by E2E ASR models, the most prominent case here being (external) language model rescoring.

All in all, we need to conclude that a) "E2E" does not provide a clear distinction between classical and novel, so-called E2E models, and b) the E2E property often is weakened in practice, leaving the term as a more general, idealized perspective on ASR modeling.

III. A Taxonomy of E2E Models in ASR

Before we derive a taxonomy of E2E ASR modeling approaches, we first introduce our notation. We denote the input speech utterance as X, which we assume has been parameterized into D-dimensional acoustic frames (e.g., log-mel features) of length T': $X = (\mathbf{x}_1, \ldots, \mathbf{x}_{T'})$, where $\mathbf{x}_t \in \mathbb{R}^D$. We denote the corresponding word sequences as C, which can be decomposed into a suitable sequence of labels of length L: $C = (c_1, \ldots, c_L)$, where each label $c_j \in \mathcal{C}$. Our description is agnostic to the specific representation used for decomposing the word sequence into labels; popular choices include characters, words, or sub-word sequences (e.g., BPE [32], word-pieces [37]).

ASR may be viewed as a sequence classification problem which maps a variable length input, X, into an output, C, of unknown length. Following Bayes' decision rule, any statistical approach to ASR must determine how to model the word sequence posterior probability, $P(C|X)$. Thus, a natural taxonomy of E2E ASR modeling can be based on the various strategies for modeling this word sequence posterior: i.e., how the alignment problem between input and output sequence is handled; and, how sequence modeling is decomposed to the level of individual input vectors $x_{t'}$ and/or output labels c_l. We find that it is useful to distinguish *implicit* and *explicit* modeling approaches, based on the modeling of the sequence-to-sequence alignment:

a) Explicit Alignment Modeling: does not necessarily refer to the determination of a single unique alignment, but instead introduces an explicit alignment modeled as a latent variable, A:

$$P(C|X) = \sum_A P(C, A|X)$$

b) Implicit Alignment Modeling: does not introduce a latent alignment variable, but models the label sequence posterior $P(C|X)$ directly.

Explicit alignment modeling approaches can mainly be distinguished by their choice of latent variable; these can be encoded in terms of valid emission paths in corresponding finite state automata (FSA) [38] which relate the input and output sequences – the approach taken in our article. Typically, latent variables in explicit alignment modeling in transducer E2E models introduce extensions to the output label set with different forms of continuation labels (including, but not limited to so-called blank labels).[3]

A. Encoder and Decoder Modules

Irrespective of the alignment modeling approach, following the notation introduced in [41], it is useful to view all E2E ASR models as being composed of an *encoder* module and a *decoder* module. The encoder module, denoted $H(X)$, maps an input acoustic frame sequence, X, of length T' into a higher-level representation, $H(X) = (\mathbf{h}_1, \ldots, \mathbf{h}_T)$ of length T (typically $T \leq T'$). Note that the encoder output is independent of the hypothesized label sequence. The decoder module models the label sequence posterior on top of the encoder output:

$$P(C|X) = P\left(C\middle|H(X)\right)$$

Thus, we may distinguish different approaches based upon how the output label sequence distribution (including potential latent variables resulting from the alignment modeling) are decomposed into individual label (and alignment) contributions; these may occur *per output label* position, per encoder *frame* position, or combinations thereof:

$$\begin{aligned} &P\left(C[, A]\middle|H(X)\right) \\ &\quad= \prod_{i=1}^{L} P\left(c_i[, a_i]\middle|c_1^{i-1}[, a_1^{i-1}], v_i(c_1^{i-1}[, a_1^{i-1}], H(X))\right) \end{aligned}$$

where the notation m_1^{i-1} corresponds to the sequence of $i-1$ previous instances of the variables m; and, $v_i(c_1^{i-1}[, a_1^{i-1}], H(X))$ denotes a context-vector that provides the connection between encoder output, $H(X)$, and the label output position, i. In general the context vector may depend on the label context (and possibly the latent variable context, for explicit alignment modeling approaches). Apart from the underlying alignment model and corresponding output label decomposition, decoder modules differ in terms of the assumptions on their label context c_1^{i-1} (and their latent variable context a_1^{i-1}), which correspond to different conditional independence assumptions, and by their access to the encoder output. For example, the local posterior may only depend on a single encoder frame output (i.e., with the context vector being reduced to a single encoder frame's output): $v_i(c_1^{i-1}, H(X)) = h_{t_i}(X)$. As we shall see in detail in the following sections, the simplest case of an encoder frame-level decomposition (with $L = T$, and

[3]For example, these extensions may also include explicit duration variables, leading to segmental models [39]. Such models can be rewritten into equivalent transducer models [40], and vice-versa.

$t_i = i$) corresponds to CTC [13]; AED models [16] and their variants maintain the full dependency of the context vector.

Finally, different E2E models can also be distinguished by the specific modeling choices that are involved in the design of the neural network used to implement the encoder and the decoder. These might involve feed-forward neural networks, convolutional neural networks, recurrent neural networks (either uni-directional or bi-directional) [42], attention [43], and various combinations thereof (e.g., transformers [44] or conformers [45]). These modeling choices and corresponding training methods can be applied across E2E ASR models and therefore do not enter the taxonomy of E2E ASR models discussed here. However, specific choices will be discussed as part of the exemplary E2E ASR models presented in Section VIII and Section IX.

B. Explicit Alignment Modeling Approaches

Early E2E modeling approaches *modeled alignments explicitly* through a latent variable, which is marginalized out (possibly, approximately) during training and inference. Examples of this family of approaches include connectionist temporal classification (CTC) [13], the recurrent neural network transducer (RNN-T) [14], the recurrent neural aligner (RNA) [46], and the hybrid auto-regressive transducer [47] (HAT). As will be discussed in subsequent sections, the latter modeling approaches in this family represent increasingly sophisticated modeling of alignments, with fewer independence assumptions and are thus increasingly powerful. A common feature of all explicit alignment models discussed in this section is that they introduce an additional *blank* symbol, denoted $\langle \mathrm{b} \rangle$, and define an output probability distribution over symbols in the set $\mathcal{C}_b = \mathcal{C} \cup \{\langle \mathrm{b} \rangle\}$. The interpretation of the $\langle \mathrm{b} \rangle$ symbol varies slightly between each of these models, as we discuss in greater details below. For now, it suffices to say that given a specific training example, (X, C), each of these models defines a set of *valid alignments*, denoted by $\mathcal{A}_{(T,C)}$, and define the conditional distribution $P(C|X)$ by marginalizing over all valid alignment sequences:

$$P(C|X) = \sum_{A} P(C|A, H(X)) P(A|H(X))$$
$$= \sum_{A \in \mathcal{A}_{(T=|H(X)|,C)}} P(A|H(X)) \quad (1)$$

where, by definition $P(C|A, H(X)) = 1$ if and only if $A \in \mathcal{A}_{(T,C)}$ and 0 otherwise.[4] We discuss the specific formulations of each of these models in the subsequent sections.

1) Connectionist Temporal Classification (CTC): Connectionist Temporal Classification (CTC) was proposed by Graves et al. [13] as a technique for mapping a sequence of input tokens to a corresponding sequence of output tokens. CTC explicitly models alignments between the encoder output, $H(X)$, and the label sequence, C, by introducing a special "blank" label, denoted by $\langle \mathrm{b} \rangle$: $\mathcal{C}_b = \mathcal{C} \cup \{\langle \mathrm{b} \rangle\}$. An alignment, $A \in \mathcal{C}_b^*$, is thus a sequence of labels in $\mathcal{C}$ or $\langle \mathrm{b} \rangle$.[5] Given a specific training example, (X, C), we denote the set of all valid alignments, $\mathcal{A}^{\mathrm{CTC}}_{(X,C)} = \{A = (a_1, a_2, \ldots, a_T)\}$, such that each $a_t \in \mathcal{C}_b$ with the additional constraint that A is identical to C after first collapsing consecutive identical labels, and then removing all blank symbols. For example, if $T = 10$, and $C = (\mathtt{s}, \mathtt{e}, \mathtt{e})$, then $A = (\mathtt{s}, \langle \mathrm{b} \rangle, \langle \mathrm{b} \rangle, \mathtt{e}, \mathtt{e}, \langle \mathrm{b} \rangle, \mathtt{e}, \mathtt{e}, \langle \mathrm{b} \rangle, \langle \mathrm{b} \rangle) \in \mathcal{A}^{\mathrm{CTC}}_{(X,C)}$, as illustrated in Fig. 1. As can be seen in this example, repeated labels in the output can be represented by intervening blanks. Following (1), CTC defines the posterior probability of the label sequence C conditioned on the input, X, by marginalizing over all possible CTC alignments as:

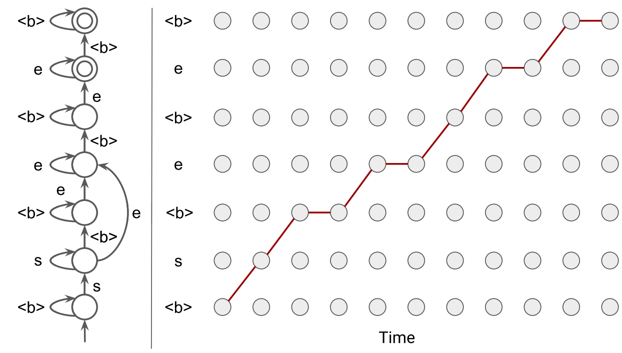

Fig. 1. Example alignment sequence for a CTC model with the target sequence $C = (\mathtt{s}, \mathtt{e}, \mathtt{e})$ (right), alongside a (non-deterministic) finite state automaton (FSA) [38] (left) representing the set of all valid alignment paths.

$$P_{\mathrm{CTC}}(C|X) = \sum_{A \in \mathcal{A}^{\mathrm{CTC}}_{(X,C)}} P(A|H(X))$$
$$= \sum_{A \in \mathcal{A}^{\mathrm{CTC}}_{(X,C)}} \prod_{t=1}^{T} P(a_t|a_{t-1}, \ldots, a_1, H(X))$$
$$= \sum_{A \in \mathcal{A}^{\mathrm{CTC}}_{(X,C)}} \prod_{t=1}^{T} P(a_t|\mathbf{h}_t) \quad (2)$$

Critically, as can be seen in (2), CTC makes a strong independence assumption that the model's output at time t is conditionally independent of the outputs at other timesteps, given the local encoder output at time t.

Thus, a CTC model consists of a neural network that models the distribution $P(a_t|X)$, at each step as shown in Fig. 2. The encoder is connected to a softmax layer with $|\mathcal{C}_b|$ targets representing the individual probabilities in (2): $P(a_t = c|X) = P(a_t = c|H(X))$, which comprises the decoder module for CTC. Thus, at each step, t, the model consumes a single encoded frame h_t and outputs a distribution over the labels; in other words, the model "outputs" a single label either blank, $\langle \mathrm{b} \rangle$, or one of the targets in $\mathcal{C}$.

2) Recurrent Neural Network Transducer (RNN-T): The Recurrent Neural Network Transducer (RNN-T) [14], [48] was proposed by Graves as an improvement over the basic CTC model [13], by removing some of the conditional independence assumptions that we discussed previously. The RNN-T model, which is depicted in Fig. 3, is best understood by contrasting it against the CTC model. As with CTC, the RNN-T model

[4] This is equivalent to the assumption that the mapping from an alignment A to a label sequence C is unique, by definition.

[5] $\mathcal{S}^*$ denotes a Kleene closure: the set of all possible sequences composed of tokens in the set $\mathcal{S}$.

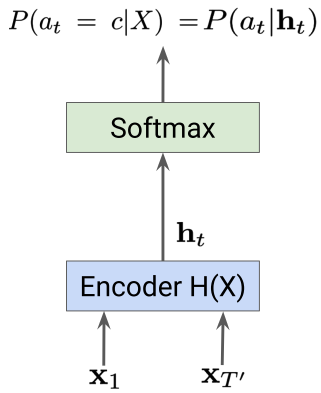

Fig. 2. Representation of the CTC model consisting of an encoder which maps the input speech into a higher-level representation, and a softmax layer which predicts frame-level probabilities over the set of output labels and blank.

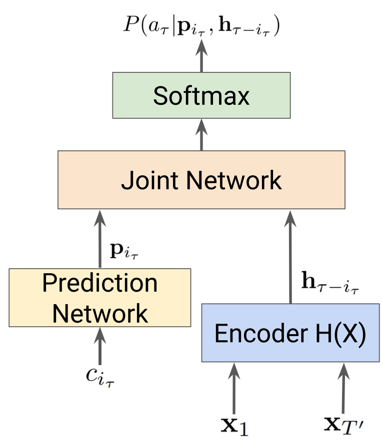

Fig. 3. RNN-T Model [14], [48] consists of an encoder which transforms the input speech frames into a high-level representation, and a prediction-network which models the sequence of non-blank labels that have been output previously. The prediction network output, p_{i_t}, represents the output after producing the previous non-blank label sequence $c_1, \ldots, c_{i_t}$. The joint network produces a probability distribution over the output symbols (augmented with blank) given the prediction network state and a specific encoded frame.

augments the output symbols with the blank symbol, and thus defines a distribution over label sequences in $\mathcal{C}_b$. Similarly, as with CTC, the model consists of an encoder which processes the input acoustic frames X to generate the encoded representation $H(X) = (\mathbf{h}_1, \ldots, \mathbf{h}_T)$.

Unlike CTC, however, the blank symbol in RNN-T has a slightly different interpretation; for each input encoder frame, $\mathbf{h}_t$, the RNN-T model outputs a sequence of zero or more symbols in $\mathcal{C}$ which are terminated by a single blank symbol. Thus, we may define the set of all valid alignment sequences in RNN-T as: $\mathcal{A}^{\text{RNNT}}_{(X,C)} = \{A = (a_1, a_2, \ldots, a_{T+L})\}$, the set of all sequences of $T + L$ symbols in $\mathcal{C}_b^*$, which are identical to C after removing all blanks. Finally, for a given output position τ, let i_τ denote the number of non-blank labels in the partial sequence $(a_1, \ldots, a_{\tau-1})$. Thus, the number of blanks in the partial sequence $(a_1, \ldots, a_{\tau-1})$ is $\tau - i_\tau - 1$. For example, if $T = 7$, and $C = (\mathtt{s}, \mathtt{e}, \mathtt{e})$, then $A = (\langle \mathrm{b} \rangle, \mathtt{s}, \langle \mathrm{b} \rangle, \langle \mathrm{b} \rangle, \langle \mathrm{b} \rangle, \mathtt{e}, \mathtt{e}, \langle \mathrm{b} \rangle, \langle \mathrm{b} \rangle, \langle \mathrm{b} \rangle) \in \mathcal{A}^{\text{RNNT}}_{(X,C)}$. Note that, unlike the CTC model, repeated labels in the output require no special treatment as illustrated in Fig. 4, where, $i_1 = i_2 = 0; i_3 = i_4 = 1; i_{10} = 3$; etc.

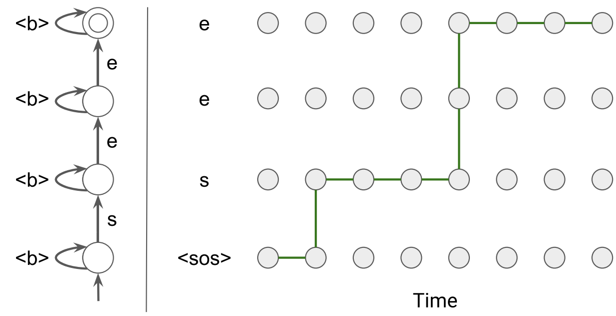

Fig. 4. Example alignment sequence (right) for an RNN-T model with the target sequence $C = (\mathtt{s}, \mathtt{e}, \mathtt{e})$. Horizontal transitions in the image correspond to blank outputs. The FSA (left) represents the set of all valid RNN-T alignment paths.

We may then define the posterior probability $P(C|X)$ as before:

$$
\begin{aligned}
P_{\text{RNNT}}(C|X) &= \sum_{A \in \mathcal{A}^{\text{RNNT}}_{(X,C)}} P(A|H(X)) \\
&= \sum_{A \in \mathcal{A}^{\text{RNNT}}_{(X,C)}} \prod_{\tau=1}^{T+L} P(a_\tau | a_{\tau-1}, \ldots, a_1, H(X)) \\
&= \sum_{A \in \mathcal{A}^{\text{RNNT}}_{(X,C)}} \prod_{\tau=1}^{T+L} P(a_\tau | c_{i_\tau}, c_{i_\tau - 1}, \ldots, c_0, \mathbf{h}_{\tau - i_\tau}) \\
&= \sum_{A \in \mathcal{A}^{\text{RNNT}}_{(X,C)}} \prod_{\tau=1}^{T+L} P(a_\tau | \mathbf{p}_{i_\tau}, \mathbf{h}_{\tau - i_\tau}) \qquad (3)
\end{aligned}
$$

where, $P = (\mathbf{p}_1, \ldots, \mathbf{p}_L)$ represents the output of the *prediction network* depicted in Fig. 3 which summarizes the sequence of previously predicted non-blank labels, implemented as another neural network: $\mathbf{p}_j = NN(\cdot|c_0, \ldots, c_{j-1})$, where c_0 is a special start-of-sentence label, $\langle \mathrm{sos} \rangle$. Thus, as can be seen in (2), RNN-T reduces some of the independence assumptions in CTC since the output at time t is conditionally dependent on the sequence of previous non-blank predictions, but is independent of the specific choice of alignment (i.e., the choice of the frames at which the non-blank tokens were emitted).

Our presentation of RNN-T alignments considers the "canonical" case. In principle, however, the model can encode the same set of conditional independence assumptions in RNN-T (i.e., the model structure), while considering alternative alignment structures as in the work of [49]. In their work, Moritz et al., represent valid frame-level alignments as an arbitrary graph. This formulation, for example, allows for the use of "CTC-like" alignments in the RNN-T model (i.e., outputting a single label – blank, or non-blank – at each frame) while conditioning on the set of previous non-blank symbols as in the RNN-T model.

3) Recurrent Neural Aligner (RNA): The recurrent neural aligner (RNA) was proposed by Sak et al. [46]. The RNA model generalizes the RNN-T model by removing one of its conditional independence assumptions. The model, depicted in

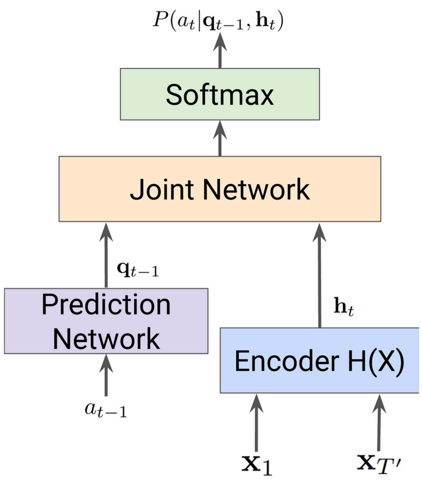

Fig. 5. RNA Model [46] resembles the RNN-T model [14], [48] in terms of the model structure. However, this model is only permitted to output a single label – either blank, or non-blank – in a single frame. Unlike RNN-T, the prediction network state in the RNA model, $\mathbf{q}_{t-1}$, depends on the entire alignment sequence $a_{t-1}, \ldots, a_1$. The joint network produces a probability distribution over the output symbols (augmented with blank) given the prediction network state and a specific encoded frame.

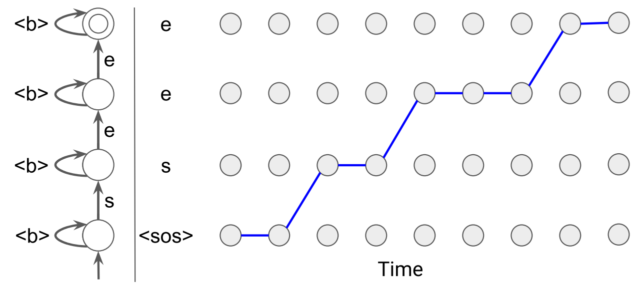

Fig. 6. Example alignment sequence (right) for an RNA model with the target sequence $C = (\texttt{s}, \texttt{e}, \texttt{e})$. Horizontal transitions in the image correspond to blank outputs; diagonal transitions correspond to outputting a non-blank symbol. The FSA (left) represents the set of valid alignments for the RNA model. Although the FSA is identical to the corresponding FSA for RNN-T in Fig. 4, the semantics of the $\langle \mathrm{b} \rangle$ label are different in the two cases.

Fig. 5, is best understood by considering how it differs from the RNN-T model. As with CTC and RNN-T, the RNA model defines a probability distribution over blank augmented labels in the set $\mathcal{C}_b$, where $\langle \mathrm{b} \rangle$ has the same semantics as in the CTC model: at each frame the model can only output a single label – either blank, or non-blank – before advancing to the next frame; unlike CTC (but as in RNN-T) the model only outputs a single instance of each non-blank label. More specifically, the set of valid alignments, $\mathcal{A}^{\text{RNA}}_{(X,C)} = (a_1, \ldots, a_T)$, in the RNA model consist of length T sequences in $\mathcal{C}_b^*$ with exactly $T - L$ blank symbols, and which are identical to C after removing all blanks. Thus, the blank symbol has a different interpretation in RNA and the RNN-T models: in RNN-T, outputting a blank symbol advances the model to the next frame; in RNA, however, the model advances to the next frame after outputting a single blank or non-blank label. Restricting the model to output a single non-blank label at each frame improves computational efficiency and simplifies the decoding process, by limiting the number of model expansions at each frame (in constrast to RNN-T decoding). For example, if $T = 8$, and $C = (\texttt{s}, \texttt{e}, \texttt{e})$, then $A = (\langle \mathrm{b} \rangle, \texttt{s}, \langle \mathrm{b} \rangle, \texttt{e}, \langle \mathrm{b} \rangle, \langle \mathrm{b} \rangle, \texttt{e}, \langle \mathrm{b} \rangle) \in \mathcal{A}^{\text{RNA}}_{(X,C)}$ as illustrated in Fig. 6.

The RNA posterior probability, $P(C|X)$, is defined as:

$$\begin{aligned} P_{\text{RNA}}(C|X) &= \sum_{A \in \mathcal{A}^{\text{RNA}}_{(X,C)}} P(A|H(X)) \\ &= \sum_{A \in \mathcal{A}^{\text{RNA}}_{(X,C)}} \prod_{t=1}^{T} P(a_t|a_{t-1}, \ldots, a_1, H(X)) \\ &= \sum_{A \in \mathcal{A}^{\text{RNA}}_{(X,C)}} \prod_{t=1}^{T} P(a_t|\mathbf{q}_{t-1}, \mathbf{h}_t) \end{aligned} \tag{4}$$

where, as before i_t denotes the number of non-blank symbols in the partial alignment sequence $(a_1, \ldots, a_{t-1})$, and $\mathbf{q}_{t-1} = \text{NN}(\cdot|a_{t-1}, \ldots, a_1)$ represents the output of a neural network which summarizes the entire partial alignment sequence, where $\text{NN}(\cdot)$ represents a suitable neural network (an LSTM in [46]). Thus, RNA removes the one remaining conditional independence assumption of the RNN-T model, by conditioning on the sequence of previous non-blank labels *as well as the alignment that generated them*. However, this comes at a cost: the exact computation of the log-likelihood in (3) (and corresponding gradients) is intractable. Instead, RNA makes two simplifying assumption to ensure tractable training: by assuming that the model can only output a single label at each frame; and utilizing a straight-through estimator for the alignment [50]. The latter constraint – allowing only a single label (blank or non-blank) at each frame – has also been explored in the context of the monotonic RNN-T model [51]. Finally, we note that the work in [52] further generalizes the RNA model by employing two RNNs when defining the state: a slow RNN (which corresponds to the sequence of previously predicted non-blank labels), and a fast RNN (which also conditions on the frames at which the non-blank labels were output).

C. Implicit Alignment Modeling Approaches

One of the main benefits of the explicit alignment approaches such as CTC, RNN-T, or RNA is that they result in ASR models that are easily amenable to *frame-synchronous* decoding.[6] In this section, we discuss the attention-based encoder-decoder (AED) models (also known as, listen-attend-and-spell (LAS)) [15], [16], [53], which employs the *attention mechanism* [43] to implicitly identify and model the portions of the input acoustics which are relevant to each output unit. These models were first popularized in the context of machine translation [54]. Unlike explicit alignment modeling approaches, attention-based encoder-decoder models use an attention mechanism [43] to learn a correspondence between the entire acoustic sequence and the individual labels. Such models support label-synchronous

[6]By frame-synchronous decoding, we refer to the ability of the model to produce output label for each input frame of speech. Models such as CTC, RNN-T, or RNA, support frame-synchronous decoding.

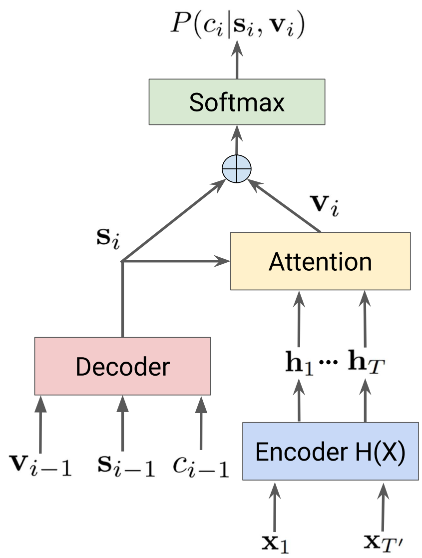

Fig. 7. Attention-based encoder decoder (AED) model [15], [16], [53]. The output distribution is conditioned on the decoder state, $\mathbf{s}_i$ (which summarizes the previously decoded symbols), and the context vector, $\mathbf{v}_i$ (which summarizes the encoder output based on the decoder state). In the seminal work of Chan et al., [16], for example, this is accomplished by concatenating the two vectors, as denoted by the $\bigoplus$ symbol in the figure.

decoding, meaning that during inference, each hypothesis in the beam is expanded by 1 label.

In the explicit alignment approaches presented in Section III-B, during inference, the model continues to output symbols until it has processed the final frame at which point the decoding process is complete; similarly, during training, the forward-backward algorithm aligns over all possible alignment sequences. Since an AED model processes the entire acoustic sequence at once, the model needs a mechanism by which it can indicate that it is done emitting all output symbols. This is achieved by augmenting the set of outputs with an end-of-sentence symbol, $\langle \text{eos} \rangle$, so that the output vocabulary consists of the set $\mathcal{C}_{\text{eos}} = \mathcal{C} \cup \{\langle \text{eos} \rangle\}$. Thus, the AED model, depicted in Fig. 7, consists of an encoder network – which encodes the input acoustic frame sequence, $X = (\mathbf{x}_1, \ldots, \mathbf{x}_{T'})$, into a higher-level representation $H(X) = (\mathbf{h}_1, \ldots, \mathbf{h}_T)$ – and an attention-based decoder which defines the probability distribution over the set of output symbols, $\mathcal{C}_{\text{eos}}$. Thus, given a paired training example, (X, C), we denote by $C_e = (c_1, \ldots, c_L, \langle \text{eos} \rangle)$, the ground-truth symbol sequence of length $(L+1)$ augmented with the $\langle \text{eos} \rangle$ symbol. AED models compute the conditional probability of the output sequence augmented with the $\langle \text{eos} \rangle$ symbol as:

$$\begin{aligned} P(C_e|X) &= P(C_e|H(X)) \\ &= \prod_{i=1}^{L+1} P(c_i|c_{i-1}, \ldots, c_0 = \langle \text{sos} \rangle, H(X)) \\ &= \prod_{i=1}^{L+1} P(c_i|c_{i-1}, \ldots, c_0 = \langle \text{sos} \rangle, \mathbf{v}_i) \\ &= \prod_{i=1}^{L+1} P(c_i|\mathbf{s}_i, \mathbf{v}_i) \end{aligned} \tag{5}$$

where, $\mathbf{v}_i$ corresponds to a *context vector*, which summarizes the relevant portions of the encoder output, $H(X)$, given the sequence of previous predictions $c_{i-1}, \ldots, c_0$; and, $\mathbf{s}_i$ corresponds to the corresponding decoder state after outputting the sequence of previous symbols, which is produced by updating the decoder state based on the previous context vector and output label:

$$\mathbf{s}_i = \text{Decoder}(\mathbf{v}_{i-1}, \mathbf{s}_{i-1}, c_{i-1})$$

The symbol $c_0 = \langle \text{sos} \rangle$ is a special start-of-sentence symbol which serves as the first input to the attention-based decoder before it has produced any outputs. As can be seen in (5), an important benefit of AED models over models such as CTC or RNN-T is that they do not make any independence assumptions between model outputs and the input acoustics, and are thus more general than the implicit alignment models, while being considerably easier to train and implement since we do not have to explicitly marginalize over all possible alignment sequences. However, this comes at a cost: previously generated context vectors (which are analogous to the decoded partial alignment in explicit alignment models) are not revised as the decoding proceeds. Stated another way, while the encoder processing $H(X)$ might be bi-directional, the decoding process in AED models reveals a left-right asymmetry [55].

1) Computing the Context Vector in AED Models: As we mentioned before, the context vector, $\mathbf{v}_i$, is computed by employing the attention mechanism [43]. The central idea behind these approaches is to define a state vector $\mathbf{s}_i$ which corresponds to the state of the model after outputting $c_1, \ldots, c_{i-1}$. The attention function, $\text{atten}(\mathbf{h}_t, \mathbf{s}_i) \in \mathbb{R}$, then defines a score between the model state after outputting $i-1$ previous symbols, and each of the encoded frames in $H(X)$. These scores can then be normalized using the softmax function to define a set of weights corresponding to each $\mathbf{h}_t$ as:

$$\alpha_{t,i} = \frac{\exp\{\text{atten}(\mathbf{h}_t, \mathbf{s}_i)\}}{\sum_{t'=1}^{T} \exp\{\text{atten}(\mathbf{h}_{t'}, \mathbf{s}_i)\}}$$

Intuitively, the weight $\alpha_{t,i}$ represents the relevance of a particular encoded frame $\mathbf{h}_t$ when outputting the next symbol c_i, after the model has already output the symbols $c_1, \ldots, c_{i-1}$, as illustrated in Fig. 8. The context vector summarizes the encoder output based on the computed attention weights:

$$\mathbf{v}_i = \sum_t \alpha_{t,i} \mathbf{h}_t$$

A number of possible attention mechanisms have been explored in the literature: the most common forms are called 'content-based attention', which include dot-product attention [16] and additive attention [43]. The content-based attention computes the attention score $\text{atten}(\mathbf{h}_t, \mathbf{s}_i)$ based on the relevance between $\mathbf{h}_t$ and $\mathbf{s}_i$. However, the score does not consider location information, i.e., it is determined by only the content, independent of the position. This can lead to incorrect attention weights with a large discrepancy against the previous steps. Thus, location-based attention $\text{atten}(\mathbf{s}_i, \mathbf{f}_{i,j})$ has been proposed [15], where $\mathbf{f}_{i,j}$ is a convolutional feature vector extracted

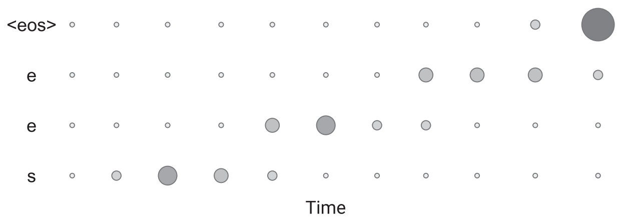

Fig. 8. Unlike models such as RNN-T or CTC, AED models do not have explicit alignment. However, it is possible to interpret the attention weights $\alpha_{t,i}$ for a particular output symbol c_i as an alignment weight which is represented above for the target sequence $C = (\mathtt{s}, \mathtt{e}, \mathtt{e}, \langle\text{eos}\rangle)$. In this representation, the size of the circle and the darkness level are proportional to the corresponding attention weights; thus the total probability mass is the same for each row. As illustrated above, the first few frames correspond to the first symbol $c_1 = \mathtt{s}$, while the latter frames correspond to the second 'e': $c_3 = \mathtt{e}$.

from $\boldsymbol{\alpha}_{i-1}$, the attention weights in the previous step. The hybrid attention, i.e., a combination of the content- and location-based attentions, has also been investigated in [15], showing a higher accuracy than the separate ones. Besides, other location-based methods use a Gaussian (mixture) model estimated with $\mathbf{s}_i$ to obtain attention weights [56], [57]. Transformer model [44] uses only content-based dot-product attention, but also takes location information into account through positional encoding. Apart from the specific choice of the attention mechanism, a common technique to improve performance involves the use of multiple independent attention heads – $\mathbf{v}_i^1, \ldots, \mathbf{v}_i^K$ – which are then concatenated together to obtain the final context vector $\mathbf{v}_i = [\mathbf{v}_i^1; \ldots; \mathbf{v}_i^K]$, in the so-called multi-head attention approach [44], or indeed by stacking together multiple attention-based layers in the transformer decoder presented by Vaswani et al. [44].

D. From Implicit to Explicit Alignment Modeling

AED models, which make no conditional independence assumptions, are extremely powerful, often outperforming explicit alignment E2E approaches such as CTC, or RNN-T [41]. However, these models also have some significant disadvantages, most notably that the models are typically non-streaming: i.e., the models must process all acoustic frames before they can generate any output hypotheses. A somewhat related issue is that the models are extremely sensitive to the length of the acoustic sequences, which requires special processing to be able to decode long-form audio [58]. There is a body of work that lies in between these two extremes: models such as the neural transducer [59], or those based on monotonic alignments [60] and its variants (e.g., monotonic chunkwise alignments (MoChA) [61], monotonic infinite lookback (MILK) [62] etc.) use an explicit alignment model, while also utilizing an attention mechanism that allows the model to *examine local acoustics* in order to refine predictions. In other words, this corresponds to a class of streaming AED models. Generally speaking, these models are motivated by the observation that speech (unlike tasks such as machine translation) exhibits a 'local' relationship between the encoded frames (assuming that the encoder is uni-directional) and the output units; thus, unlike the general AED model which computes the context vector, $\mathbf{v}_i$, as a sum over all input frames $\mathbf{h}_t$, the various proposed models constrain this sum to be computed over a subset of frames to allow for streaming decoding. In the context of our presentation, it is easiest to think of these models as consisting of an underlying alignment (whether known or unknown) which can be used to perform streaming inference.

The Neural Transducer (NT) [59] explicitly partitions the input encoder frames into T^W non-overlapping chunks of length W: $H_1^W = [\mathbf{h}_1, \ldots, \mathbf{h}_W]; \cdots; H_{T^W}^W = [\mathbf{h}_{T^W+1}, \ldots, \mathbf{h}_{T^W W}]$, where $T^W = \lceil \frac{T}{W} \rceil$, and $\mathbf{h}_t = \mathbf{0}$ if $t > T$. Unlike the AED model which examines all encoded frames when computing the context vector, the NT model is restricted to process a single chunk at a time; the model only advances to the next chunk when it outputs a special end-of-chunk symbol (analogous to $\langle\text{eos}\rangle$ in the AED model); inference in the model terminates when the model has output the end-of-chunk symbol in the final chunk $H_{T^W}^W$. If the alignments of the ground-truth output sequence, C, with respect to the W-length chunks are unknown, then it is possible to train the system by using a rough initial alignment where symbols are distributed equally among the T^W chunks, followed by iterative refinement by computing the most likely output alignments given the current model parameters [59] similar to forced-alignments in HMM-based systems. An alternate approach [63] consists of using a separate system (e.g., a classical hybrid system) to get initial alignments (e.g., word-level alignments), which can be used to assign sub-word units to the individual chunks.

An alternative approach, proposed by Raffel et al. [60], modifies the vanilla AED model by explicitly introducing an alignment module which scans the encoder frames, $H(X)$, from left-to-right to identify whether the current frame should be used to emit any outputs (modeled as a Bernoulli random variable). If a frame, τ, is selected, then the model produces an output based on the local encoder frame, $\mathbf{h}_\tau$. The process is then repeated starting from the currently selected frame, thus allowing multiple outputs to be generated at the same frame. This results in a model with hard monotonic alignments between the input speech and the output labels since the models are constrained to generate outputs in a streaming fashion. A Monotonic Chunkwise Attention (MoChA) model [61] improves upon the work of Raffel et al., by allowing the model to generate the next output using a context vector computed using attention over a local window of frames to the left of the selected frame τ: $\mathbf{h}_{\tau-W+1}, \ldots, \mathbf{h}_\tau$. Thus, the MoChA model consists of a two-level process – identifying frames where output should be produced following [60], followed by an AED model over frames to the left of the selected frame. A refinement to the MoChA model, proposed by Arivazhagan et al. [62] – the monotonic infinite lookback (MILK) attention model – computes the context vector over all frames to the left of the selected frame τ (i.e., $\mathbf{h}_1, \ldots, \mathbf{h}_\tau$) at each step. Another two-fold approach to enable streaming operation is presented in [64] under the term of triggered attention, where a CTC-network is used to trigger, i.e. control the activation of an AED model with a limited decoder delay. We also direct interested readers to studies of various attention variants: Merboldt et al. [65] compare a number of local monotonic attention variants; Zeyer et al. [66] discuss segmental attention variants; Zeyer et al. [67] study the related decoding and the relevance of segment length modeling, leading to improved generalization towards long sequences. Segmental attention models are related to transducer models [68]. However, segmental E2E ASR models are not limited to be realized based on the attention mechanism and may not only be related to a direct HMM [39], but have also been shown to be equivalent to neural transducer modeling [40], thus even providing a clear relation between duration modeling and blank probabilities.

Relationship to Classical ASR

In classical ASR models, these frame-level alignments can be modeled with HMMs while using generative GMMs or neural networks to model the output distribution of acoustic frames; frame-level alignments to train neural network acoustic models may be obtained by force-alignment from a base GMM-HMM systems, but direct sequence training not requiring initial alignments is also possible [69].

E2E models introduce alternative alignment modeling approaches to ASR. While the attention mechanism provides a qualitatively novel approach to map acoustic observation sequences to label sequences, transducer approaches [13], [14], [46], [70] handle the alignment problem in a way that can be interpreted to be similar to HMMs with a specific model topology, including marginalization over alignments [71], [72], [73]. CTC models can also be employed in an HMM-like fashion during decoding [74]. Moreover, transducer approaches are equivalent to segmental models/direct HMM [40].

Another prominent feature of E2E systems besides the alignment property is their direct character-level modeling avoiding phoneme-based modeling and pronunciation lexica [16], [19], [74], [75], [76], [77], [78], [79], [80], [81], [82], with some even heading for whole-word modeling [30], [76]. However, character-level modeling also is viable with classical hybrid HMM architectures [83] and has been shown to work even with standard HMM models w/o neural networks [84].

IV. Architecture Improvements to Basic E2E Models

In this section, we describe various algorithmic changes to vanilla E2E models which are critical in order to obtain improved performance over classical ASR systems. First, we describe various ways of combining different complementary E2E models to improve performance. Next, we introduce ways to incorporate context into these models to improve performance on rare proper noun entities. We then describe improved encoder and decoder architectures that take better advantage of the many cores on specialized architectures such as tensor processing units (TPUs) [85]. Finally, we discuss how to improve the latency of the model through an integrated E2E endpointer.

A. Combinations of Models

Different end-to-end models are complementary, and there have been numerous attempts at combining these methods. For example, Watanabe et al. [86] find that attention-based models perform poorly on long or noisy utterances, mainly because the model has too much flexibility in predicting alignments when presented with the entire input utterance. In contrast, models such as CTC – which have left-to-right constraints during decoding – perform much better in these cases. They propose to employ a multi-task learning strategy with both CTC and attention-based losses, which provides a 5–14% relative improvement in word error rate over attention-based models on Wall Street Journal (WSJ) and Chime tasks. Pang et al. [87] explore combining the benefits of RNN-T and AED. Specifically, RNN-T decodes utterances in a left-to-right fashion, which works well for long utterances. On the other hand, since AED sees the entire utterance, it often shows improvements for utterances where surrounding context is needed to predict the current word, e.g., `"one dollar and fifty cents"` $\rightarrow$ \$1.50. To combine RNN-T and AED, the authors propose to produce a first-pass result with RNN-T, that is then rescored with AED in the second-pass. To reduce computation, the authors share the encoder between RNN-T and AED. The authors find that RNN-T + AED provides a 17–22% relative improvement in word error rate over RNN-T alone on a voice search task. Other flavors of streaming 1st-pass following by attention-based 2nd-pass rescoring, such as deliberation [88], have also been explored. One of the issues with such rescoring approaches is that any potential improvements are limited to the lattice produced by the 1st-pass system. To address this, methods which run a 2nd-pass beam search have also been explored, particularly in the context of streaming ASR – e.g. cascaded encoder [89], Y-architecture [90] and Universal ASR [91].

B. Incorporating Context

Contextual biasing to a specific domain, including a user's song names, app names and contact names, is an important component of any production-level automatic speech recognition (ASR) system. Contextual biasing is particularly challenging in E2E models because these models typically retain only a small list of candidates during beam search, and tend to perform poorly when recognizing words that are seen infrequently during training (typically named entities), which is the main source of biasing phrases. There have been a few approaches in the literature to incorporate context.

One approach, known as shallow-fusion contextual biasing [92], constructs a stand-alone weighted finite state transducer (FST) representing the biasing phrases. The scores from the biasing FST are interpolated with the scores of the E2E model during beam search, with special care taken to ensure we do not over- or under-bias phrases. An alternate approach proposes to inject biasing phrases into the model in an all-neural fashion. For example, Pundak et al. [93] represent a set of biasing phrases by embedding vectors. These vectors are fed as additional input to an attention-based model, which can then choose to attend to the phrases and hence boost the chances of predicting the phrases. Kim and Metze [94] propose to bias towards dialog context. In addition, Bruguier et al. [95] extend [93], by leveraging phonemic pronunciations for the biasing phrases when constructing phrase embeddings. Finally, Delcroix et al. [96] use an utterance-wise context vector like an i-vector computed by a pooling across frame-by-frame hidden state vectors obtained from a sub network (this sub-network is called a sequence-summary network).

C. Encoder and Decoder Structure

There have been improvements to encoder architectures of E2E models over time. The first end-to-end models used long short-term memory recurrent neural networks (LSTMs), for both the encoder and decoder. The main drawback of these sequential models is that each frame depends on the computation from the previous frame, and therefore multiple frames cannot be batched in parallel.

With the improvement of hardware, specifically on-device Edge Tensor Processing Units (TPUs), with thousands of cores, architectures that can better take advantage of the hardware, have been explored. Such architectures include convolution-based architectures, such as ContextNet [97]. The use of self-attention to replace the sequential recurrence in LSTMs was explored in Transformers for ASR [98], [99], [100]. Finally, combining self-attention with convolution, known as Conformer [45], or multi-layer perceptron [101], was also explored. Both Transformer and Conformer have shown competitive performance to LSTMs on many tasks [102], [103].

On the decoder side, research for transducer models has shown that a large LSTM decoder can be replaced with a simple embedding lookup table, that attends to only a few previous tokens from the model [47], [104], [105], [106], [107]. This demonstrates that most of the power of the E2E model is in the encoder, which has been a consistent theme of both E2E as well as classical hybrid HMM models. However, improved decoder modeling may also be effective depending on the specific downstream task. Research has shown that extended decoder architectures enable pre-training and adaptation of the decoder using extensive text-only data, leading to accuracy gains [108], [109]. For example, one approach separates RNN-T's prediction network into separate blank and vocabulary prediction (LM) components, where the LM component can be trained with text data [108]. In addition, in line with the growing interest in large language models in recent years, research has also begun on solving multiple tasks, including speech recognition, using only an auto-regressive, GPT-style decoder [110], [111].

D. Integrated Endpointing

An important characteristic of streaming speech recognition systems is that they must endpoint quickly, so that the ASR result can be finalized and sent to the server for the appropriate action to be performed. Endpointing is typically done with an external voice-activity detector. Since endpointing is both an acoustic and language model decision, recent works in streaming RNN-T models [112], [113] have investigated predicting a microphone closing token $\langle \text{eos} \rangle$ at the end of the utterance – e.g., "What's the weather $\langle \text{eos} \rangle$". Following the notation from Section III, this is done by including an $\langle \text{eos} \rangle$ token as part of the set of class labels C and encouraging the model to predict this token to terminate decoding. These models have shown improved latency and WER trade-off by having the endpointing decision predicted as part of the model. Furthermore, [114], [115] explored using the CTC blank symbol for endpoint detection.

V. Training E2E Models

In general, training of E2E models follows deep learning schemes [116], [117], with specific consideration of the sequential structure and the latent alignment problem to be handled in ASR. E2E ASR models may be trained end-to-end, notwithstanding potential elaborate training schedules and extensive data augmentation. Part of the appeal of end-to-end models is that they do not assume conditional independence between the input frames. Given a training set $\mathcal{T} = \{(X_n, C_n)\}_{n=1}^N$, the training criterion $\mathcal{L}$ to be minimized can be written as: $\mathcal{L} = -\sum_{n=1}^N \log P(C_n|X_n)$ (which is equivalent to maximizing the total conditional log-likelihood).

A. Alignment in Training

E2E models such as RNN-T and CTC introduce an additional blank token $\langle \text{b} \rangle$ for alignment. Therefore optimization implies marginalizing across all alignments, as follows:

$$\mathcal{L}_{\text{ex}} = -\sum_{n=1}^{N} \sum_{A_n} \log P(C_n, A_n|X_n)$$

This requires the forward-backward algorithm [118], [119] for efficient computation of the training criterion and its gradient, with minor modifications for CTC, RNN-T, and RNA models, as well as classical (full-sum) hybrid ANN/HMMs corresponding to the differences in alignments defined in each of these models. In comparison, AED models are based on implicit alignment modeling approaches, and the training criterion does not have a latent variable A for explicit alignment as:

$$\mathcal{L}_{\text{im}} = -\sum_{n=1}^{N} \log P(C_n|X_n)$$

We refer the interested reader to the individual papers for further details on the training algorithms [13], [14], [15], [16], [46], [48], [53], [71], [120]. As shown in Section III-A, in both explicit and implicit alignment cases, $P(C|X)$ is factorized with respect to input time t and output position i, respectively, and the factorized distribution is conditioned on the label context c_1^{i-1}, except for CTC. For example, in the AED case: $\log P(C|X) = \sum_{i=1}^L \log P(c_i|X, c_1^{i-1})$. During training, we use a teacher-forcing technique where the ground truth history is used as a label context.

As part of the training procedure, all E2E as well as classical hidden Markov models for ASR provide mechanisms to solve the underlying sequence alignment problem - either explicitly via corresponding latent variables, as in CTC, RNN-T or RNA, and also hybrid ANN/HMM, or implicitly, as in AED models. Also, the distinction between speech and silence needs to be considered, which may be handled explicitly by introducing silence as a latent label (hybrid ANN/HMM), or implicitly by not labeling silence at all, as currently is the standard in virtually all E2E models.

E2E models also may take advantage of hierarchical training schedules. These schedules may comprise several separate training passes and explicit, initially generated alignments that are kept fixed for some Viterbi-style [121], [122], [123] training epochs before re-enabling E2E-style full-sum training that marginalizes over all possible alignments. Such an alternative approach is employed by Zeyer et al. [52], where an initial full-sum RNN-T model is used to generate an alignment and continue with framewise cross-entropy training. This greatly simplifies the training process by replacing the summation over all possible alignments in (4) by a single term corresponding to the alignment sequence generated. Recently, a similar procedure has been introduced in [124] also employing E2E models, only. In this work, CTC is used to initialize the training and to generate an initial alignment, followed by intermediate Viterbi-style RNN-T training and final full-sum fine tuning, which improved convergence compared to full-sum-only training approaches.

It is interesting to note that in contrast to the RNN-T and RNA label-topologies, CTC does not require alignments with single label emissions per label position. However, training CTC models eventually does lead to single label emissions per hypothesized label. An analysis of this property of CTC training which is usually called *peaky behavior* can be found in [125] and references therein. Laptev et al. [126] even introduces a CTC variant without non-blank loop transitions.

B. Training With External Language Models

E2E ASR models generally are normalized on sequence level. Therefore, sequence training with the maximum mutual information criterion [127] is the same as standard cross entropy/conditional likelihood training. However, once external language models are included in the training phase, sequence normalization needs to be included explicitly, leading to MMI sequence discriminative training. This has been exploited as a further approach to combine E2E models with external language models trained on text-only data during the training phase itself [128], [129], [130].

C. Minimum Word Error Rate Training

Since the objective of speech recognition is to minimize word error rate (WER), there has been a growing number of research studies that incorporate this into the objective function by minimizing the model-based expectation of the number of word errors, as follows:

$$\mathcal{L}_{\text{mwer}} = \sum_{n=1}^{N} \sum_{C'_n} \mathcal{W}(C_n, C'_n) P(C'_n | X_n)$$

where $\mathcal{W}(C_n, C'_n)$ is the word error count in a hypothesis C'_n given a reference C_n, and n is an index which iterates over the entire training set. These methods, known as sequence or discriminative training, have shown great improvements for classical ASR [131], [132], [133], [134], [135], and have since been explored in E2E models. Typically these losses are constructed by running in 'beam-search' mode rather than teacher-forcing mode, and construct a loss from the errors made from the candidate hypotheses in the beam. Thus, this type of training first requires training the model to optimize $P(C|X)$ in order to initialize the model with a good set of parameters to run a beam search. However, also direct approaches have been introduced that avoid this separation to train discriminatively from scratch [69], [136].

Papers that explore penalizing word errors include, Minimum Word Error Rate (MWER) training [137], where the loss function is constructed such that the expected number of word errors are minimized. Further work includes MWER for RNN-T and self-attention-T [138], as well as MWER using prefix search instead of n-best [139]. Also, there have been studies that consider MWER in terms of reinforcement learning [140], [141]. Optimal Completion Distillation (OCD) [81] proposes to minimize the total edit distance using an efficient dynamic programming algorithm. Finally, another body of research with sequence training introduce a separate external language model at training time [142], which can also be done efficiently via approximate lattice recombination [129] and also lattice-free approaches [130].

D. Pretraining

All E2E models as well as classical hidden Markov models for ASR provide holistic models that in principle enable training from scratch. However, many strategies exist to initialize and guide the training process to reach optimal performance and/or to obtain efficient convergence by applying pretraining and model growing [143], [144]. Supervised layer-wise pretraining has been successfully applied for classical [5], [145], as well as attention-based ASR models [146], which can be combined with intermediate sub-sampling schemes [147], and model growing [148]. Pretraining approaches utilizing untranscribed audio, large-scale semi-supervised data and/or multilingual data [149], [150], [151], [152], [153], [154], [155], [156], [157], [158], [159], [160] would deserve a self-contained survey and they are applicable for hybrid DNN/HMM and E2E approaches likewise – they will not be further discussed here.

E. Training Schedules and Curricula

Dedicated training schedules have been developed to guide the optimization process and as part of that reach proper alignment behavior explicitly or implicitly [52], [124], [147]. Many approaches exist for learning rate control [161], [162]: NewBob [163], [164] and enhancements [162]; global versus parameter-wise learning rate control (exponential decay, power decay, etc.) [165]; learning rate warm-up [44]; warm restarts/cosine annealing [166]; weight decay versus gradually decreasing batch size [167]; fine-tuning [168] or population-based training [169]; etc. For a survey of meta learning cf. [170].

Sequence learning approaches also consider curriculum learning [171], [172], e.g., by considering short sequences first [173], [174]; interim increase of sub-sampling [147] initially more sub-sampling; or, for multi-speaker ASR training sort mixed speech by SNR and start with speakers of balanced energy and mixed gender [175].

F. Optimization and Regularization

Optimization usually is based on stochastic gradient descent [176], with momentum [177], [178], and a number of corresponding adaptive approaches, most prominently Adam [179] and variants thereof [145], [179], [180].

Investing more training epochs seems to provide improvements [52, Table 8], and also averaging over epochs has been reported to help [102]. For a discussion of the double descent effect and its relation to the amount of training data, label noise and early stopping cf. [181].

Regularization strongly contributes to training performance: e.g., L2 and weight decay [166], [182]; weight noise [183]; adaptive mean L1/L2 [184]; gradient noise [185]; dropout [186], [187], [188], layer dropout [189], [190], [191]; dropconnect [192]; zoneout [193]; smoothing of attention scores [15]; label smoothing [194]; scheduled sampling [195]; auxiliary loss [194], [196]; variable backpropagation through time [197], [198]; mixup [199]; increased frame rate [180]; or, batch normalization [200].

G. Data Augmentation

Training of E2E ASR models also benefit from data augmentation methods, which might also be viewed as regularization methods. However, their diversity and impact on performance justifies a separate overview.

Most data augmentation methods perform data perturbation by exploiting certain dimensions of speech signal variation: speed perturbation [201], [202], vocal tract length perturbation [201], [203], frequency axis distortion [201], sequence noise injection [204], SpecAugment [205], or semantic mask [206]. Also, text-only data may be used to generate data using text-to-speech (TTS) on feature [207] or signal level [208]. In a comparison of the effect of TTS-based data augmentation on different E2E ASR architectures in [208], AED seemed to be the only architecture that appeared to benefit significantly from the TTS data.

In a recent study [174] and corresponding follow-up work [180], many of the regularization and data augmentation methods listed here have been exploited jointly leading to state-of-the-art performance on the Switchboard task for a single-headed AED model.

Relationship to Classical ASR

E2E systems attempt to define ASR models that integrate all knowledge sources into a single global joint model that does not utilize secondary knowledge sources and avoids the classical separation into acoustic and language models. These global joint models are completely trained from scratch using a single global training criterion based on a single kind of (transcribed) training data and thus require less ASR domain-specific knowledge provided sufficient amounts of training data are available.

While standard hybrid ANN/HMM training for ASR using frame-wise cross entropy already is discriminative, it is not yet sequence discriminative, requires prior alignments and also lacks consideration of an (external) language model during training. However, these potential shortcomings may be remedied by using sequence discriminative training criteria [127] and lattice-free training approaches [69].

In contrast to strict E2E systems, the classical ASR architecture includes the use of secondary knowledge sources beyond the primary training data, i.e. (transcribed) speech audio for acoustic model training, and textual data for language model training. Most prominently, this includes the use of a pronunciation lexicon and the definition of a phoneme set. Secondary resources like pronunciation lexica may be helpful in low-resource scenarios. However, their generation often is costly and may even introduce errors, like pronunciations from a lexicon not reflecting the actual pronunciations observed. Therefore, for large enough training resources, secondary knowledge sources might become obsolete [209], or even harmful, in case of erroneous information introduced [210], [211].

Classical ASR models usually are trained successively, with knowledge derived from models trained earlier injected into later training stages, e.g. in the form of HMM state alignments. However, such approaches from classical ASR might also be interpreted as specific training schedules. Initializing deep learning models using HMM alignments obtained from acoustic models based on mixtures of Gaussians may be interpreted in this way, with the Gaussian mixtures serving as an initial shallow model. In classical ASR, also approaches training deep neural networks from scratch while avoiding intermediate training of Gaussians has been proposed [212], [213], [214], also in combination with character-level modeling [83]. Another step towards more integrated training of classical systems has been to apply discriminative training criteria avoiding intermediate (usually lattice-based) representations of competing word sequences [69], [136], [215], [216], [217].

The training of classical ASR systems usually applies secondary objectives to solve subtasks like phonetic clustering. The classification and regression trees (CART) approach is used to cluster triphone HMM states [27], [218]. More recent approaches proposed clustering within a neural network modeling framework, while still retaining secondary clustering objectives [213], [219]. However, also in E2E approaches secondary objectives are used, most prominently for subword generation, e.g. via byte-pair encoding [32]. Also, available pronunciation lexica can be utilized indirectly for assisting subword generation for E2E systems [35], [36], which are shown to outperform byte-pair encoding. Within classical ASR systems, phonetic clustering also can be avoided completely by modeling phonemes in context directly [220].

It is interesting to observe that specifically attention-based encoder-decoder models require larger numbers of training epochs to reach high performance, e.g. for a comparison of systems trained on Switchboard 300 h cf. Table 5 in [221]. Also, attention-based encoder-decoder models have been shown to suffer from low training resources [222], [223], which can be improved by a number of approaches, including regularization techniques [174] as well as data augmentation using SpecAugment [224] and text-to-speech (TTS) [29]. SpecAugment also is shown to improve classical hybrid HMM models [225]. TTS on the other hand considerably improved attention-based encoder-decoder models trained on limited resources, but did not reach the performance of other E2E approaches or hybrid HMM models, which in turn were not considerably improved by TTS [208]. Multilingual approaches also help improve ASR development for low resource tasks, again both for classical [226], as well as for E2E systems [227], [228].

VI. Decoding E2E Models

This section describes several decoding algorithms for end-to-end speech recognition. The basic decoding algorithm of end-to-end ASR tries to estimate the most likely sequence $\hat{C}$ among all possible sequences, as follows:

$$\hat{C} = \arg\max_{C \in \mathcal{U}^*} P(C|X)$$

The following section describes how to obtain the recognition result $\hat{C}$.

A. Greedy Search

The Greedy search algorithm is mainly used in CTC, which ignores the dependency of the output labels as follows:

$$\hat{A} = \prod_{t=1}^{T} \left(\arg\max_{a_t} P(a_t|X) \right)$$

where a_t is an alignment token introduced in Section III-B1. The original character sequence is obtained by converting alignment token sequence $\hat{A}$ to the corresponding token sequence $\hat{C}$. The argmax operation can be performed in parallel over input frame t, yielding fast decoding [13], [229], although the lack of the output dependency causes relatively poor performance than the attention and RNN-T based methods in general.

CTC's fast decoding is further boosted with transformer [44], [98], [102] and its variants [45], [103] since their entire computation across the frames is parallelized [190], [230]. For example, the non-autoregressive models, including Imputer [231], Mask-CTC [230], Insertion-based modeling [232], Continuous integrate-and-fire (CIF) [233] and other variants [234], [235] have been actively studied as an alternative non-autoregressive model to CTC. [235] shows that CTC greedy search and its variants achieve 0.06 real-time factor (RTF)[7] by using Intel(R) Xeon(R) Silver 4114 CPU, 2.20 GHz. The paper also shows that the degradation of the non-autoregressive models from the attention/RNN-T methods with beam search is not extremely large (19.7% with self-conditioned CTC [234] versus 18.5 and 18.9% with AED and RNN-T, respectively).

The greedy search algorithm is also used as approximate decoding for both implicit and explicit alignment modeling approaches, including AED, RNA, CTC, and RNN-T, as follows:

$$\hat{c}_i = \arg\max_{c_i} P(c_i|\hat{C}_{1:i-1}, X) \text{ for } i = 1, \ldots, N$$

$$\hat{a}_t = \arg\max_{a_t} P(a_t|\hat{A}_{1:t-1}, X) \text{ for } t = 1, \ldots, T$$

The greedy search algorithm does not consider alternate hypotheses in a sequence compared with the beam search algorithm described below. However, it is known that the degradation of the greedy search algorithm is not very large [16], [46], especially when the model is well trained in matched conditions.[8]

[7]The ratio of the actual decoding time to the duration of the input speech.

B. Beam Search

The beam search algorithm is introduced to approximately consider a subset of possible hypotheses $\tilde{\mathcal{C}}$ among all possible hypotheses $\mathcal{U}^*$ during decoding, i.e., $\tilde{\mathcal{C}} \subset \mathcal{U}^*$. A predicted output sequence $\hat{C}$ is selected among a hypothesis subset $\tilde{\mathcal{C}}$ instead of all possible hypotheses $\mathcal{U}^*$, i.e.,

$$\hat{C} = \arg\max_{C \in \tilde{\mathcal{C}}} P(C|X) \tag{6}$$

The beam search algorithm is to find a set of possible hypotheses $\tilde{\mathcal{C}}$, which can include promising hypotheses efficiently by avoiding the combinatorial explosion encountered with all possible hypotheses $\mathcal{U}^*$.

There are two major beam search categories: 1) *frame synchronous* beam search and 2) *label synchronous* beam search. The major difference between them is whether it performs hypothesis pruning for every input frame t or every output token i. The following sections describe these two algorithms in more detail.

C. Label Synchronous Beam Search

Suppose we have a set of partial hypotheses up to $(i-1)$th token $\tilde{\mathcal{C}}_{1:i-1}$. A set of all possible partial hypotheses up to ith token $\mathcal{C}_{1:i}$ is expanded from $\tilde{\mathcal{C}}_{1:i-1}$ as follows:

$$\mathcal{C}_{1:i} = \{(\tilde{\mathcal{C}}_{1:i-1}, c_i = c)\}_{c \in \mathcal{U}} \tag{7}$$

The number of hypotheses $|\mathcal{C}_{1:i}|$ would be $|\tilde{\mathcal{C}}_{1:i-1}| \times |\mathcal{U}|$, at most. The beam search algorithm prunes the low probability score hypotheses from $\mathcal{C}_{1:i}$ and only keeps a certain number (beam size Δ) of hypotheses at i among $\mathcal{C}_{1:i}$. This pruning step is represented as follows:

$$\tilde{\mathcal{C}}_{1:i} = \mathrm{NBEST}_{C_{1:i} \in \mathcal{C}_{1:i}} P(C_{1:i}|X), \text{ where } |\tilde{\mathcal{C}}_{1:i}| = \Delta \tag{8}$$

Note that $\mathrm{NBEST}(\cdot)$ is an operation to extract top Δ hypotheses in terms of the probability score $P(C_{1:i}|X)$ computed from an end-to-end neural network, or a fusion of multiple scores described in Section VII-B.

In the label synchronous beam search, the length of the output sequence (N) is unknown. Therefore, during this pruning process, we also add the hypothesis that reaches the end of an utterance (i.e., predict the end of sentence symbol $\langle \text{eos} \rangle$) to a set of hypotheses $\tilde{\mathcal{C}}$ in (6) as a promising hypothesis.

The label synchronous beam search does not explicitly depend on the alignment information; thus, it is often used in implicit alignment modeling approaches, including AED. Due to this nature, sequence hypotheses of the same length might cover a completely different number of encoder frames, unlike the frame synchronous beam search, as pointed out by [40]. As a result, we observe that the scores of very short and long segment hypotheses often become the same range, and the beam search

[8]On the other hand, in the AED models, increasing the search space does not consistently improve the speech recognition performance [77], [236] – a fact also observed in neural machine translation [237].

wrongly selects such hypotheses. [86] shows an example of such extreme cases, resulting in large deletion and insertion errors for short and long-segment hypotheses, respectively. Thus, the label synchronous beam search requires heuristics to limit the output sequence length to avoid extremely long/short output sequences. Usually, the minimum and maximum length thresholds are determined proportionally to the input frame length $|X|$ with tunable parameters $\rho_{\min}$ and $\rho_{\max}$ as $L_{\min} = \lfloor \rho_{\min}|X| \rfloor$, $L_{\max} = \lfloor \rho_{\max}|X| \rfloor$. Although these are quite intuitive ways to control the length of a hypothesis, the minimum and maximum output lengths depend on the token unit or type of script in each language. Another heuristic is to provide an additional score related to the output length or attention weights – e.g., a length penalty, and a coverage term [77], [238]. The end-point detection [239] is also used to estimate the hypothesis length automatically. [236] redefines the implicit length model of the attention decoder to take into account beam search, resulting in consistent behavior without degradation for increasing beam sizes.

Note that there are several studies on applying label synchronous beam search to explicit alignment modeling approaches. For example, label synchronous beam search algorithms for CTC are realized by marginalizing all possible alignments for each label hypothesis [13]. [240] extends CIF [233] to produce label-level encoder representation and realizes label synchronous beam search in RNN-T.

D. Frame Synchronous Beam Search

In contrast to the label synchronous case in (8), the frame synchronous beam search algorithm performs pruning at every input frame t, as follows:

$$\tilde{\mathcal{C}}_{1:i(t)} = \mathrm{NBEST}_{C_{1;i(t)}} P(C_{1;i(t)}|X), \text{ where } |\tilde{\mathcal{C}}_{1:i(t)}| = \Delta$$

where $C_{1;i(t)}$ is an $i(t)$-length label sequence obtained from the alignment $A_{1:t}$, which is introduced in Section III-B. $P(C_{1;i(t)}|X)$ is obtained by summing up all possible alignments $A_{1:t} \in \mathcal{A}_{(X,C_{1;i(t)})}$. Unlike the label synchronous beam search, frame synchronous beam search depends on explicit alignment $\mathcal{A}$; thus, it is often used for explicit alignment modeling approaches, including CTC, RNN-T, and RNA. $\mathcal{C}_{1:i(t)}$ is an expanded partial hypotheses up to input frame t, similar to (7).

Compared with the label synchronous algorithm, the frame synchronous algorithm needs to handle additional output token transitions inside the beam search algorithm. The frame synchronous algorithm can be easily extended in online and/or streaming decoding, thanks to the explicit alignment information with input frame and output token.

Classical approaches to beam search for HMM, but also CTC and RNN-T variants, are based on weighted finite state transducers (WFST) [38], [74], [241] or lexical prefix trees [106], [242], [243]. They are categorized as frame synchronous beam search. These methods are often combined with an N-gram language model or a full-context neural language model [244], [245]. RNN-T [14], [246] and CTC prefix search [247] can deal with a neural language model by incorporating the language model score in the label transition state. Interestingly, triggered attention approaches [248], [249] allow us to use implicit alignment modeling approaches, including AED, in frame-synchronous beam search together with CTC and neural LM, which applies on-the-fly rescoring to the hypotheses given by CTC prefix search using the AED and LM scores.

E. Block-Wise Decoding

Another beam search implementation uses a fixed-length block unit for the input feature. In this block processing, we can use the future context inside the block by using the non-causal encoder network based on the BLSTM, output-delayed unidirectional LSTM, or transformer (and its variants). This future context information avoids the degradation of the fully causal network. In this setup, the chunk size becomes the trade-off of controlling latency and accuracy. This technique is used in both RNN-T [100], [250], [251] and AED [61], [252], [253], [254]. Block-wise processing is especially important for implicit alignment modeling approaches, including AED, since it can provide block-wise monotonic alignment constraint between the input feature and output label, and realize block-wise streaming decoding.

F. Model Fusion During Decoding

Similar to the classical HMM-based beam search, we combine various scores obtained from different modules, including the main end-to-end ASR and LM scores.

1) Synchronous Score Fusion: The most simple score fusion is performed when the scores of multiple modules are synchronized. In this case, we can simply add the multiple scores at each frame t or label i. The most well-known score combination is LM shallow fusion.

LM shallow fusion: As discussed in Section VII, various neural LMs can be integrated with end-to-end ASR. The most simple integration is based on LM shallow fusion [255], [256], [257], as discussed in Section VII-B1, which (log-) linearly adds the LM score $P_{\mathsf{lm}}(C_{1:i})$ to E2E ASR scores $P(C_{1:i}|X)$ during beam search in (8) as follows:

$$\log P(C_{1:i}|X) \to \log P(C_{1:i}|X) + \gamma \log P_{\mathsf{lm}}(C_{1:i})$$

where γ is a language model weight. Of course, we can combine other scores, such as the length penalty and coverage terms, as discussed in Section VI-C.u

2) Asynchronous Score Fusion: If we combine the frame-dependent score functions, $P(a_t|\cdot)$, used in explicit alignment modeling approaches, e.g., CTC, RNN-T, and label-dependent score functions, $P(c_i|\cdot)$, used implicit alignment modeling approaches, e.g., AED, language model, we have to deal with the mismatch between the frame and label time indices t and i, respectively.

In the time-synchronous beam search, this fusion is performed by incorporating the language model score in the label transition state [22], [70], [258]. [247] also combines a word-based language model and token-based CTC model by incorporating the language model score triggered by the word delimiter (space) symbol.

In the label-synchronous beam search, we first compute the label-dependent scores from the frame-dependent score function by marginalizing all possible alignments given a hypothesis label sequence. CTC/attention joint decoding [86] is a typical example, where the CTC score is computed by marginalizing all possible alignments based on the CTC forward algorithm [229]. This approach eliminates the wrong alignment issues and difficulties of finding the correct end of sentences in the label-synchronous beam search [86].

Note that the model fusion method during beam search can realize simple one-pass decoding, while it limits the time unit of the models to be the same or it requires additional dynamic programming to adjust the different time units, especially for the label-synchronous beam search. This dynamic programming computation becomes significantly large when the length of the utterance becomes larger and requires some heuristics to reduce the computational cost [259].

G. Lexical Constraint During Score Fusion

Classically, we use a word-based language model to capture the contextual information with the word unit, and also consider the word-based lexical constraint for ASR. However, end-to-end ASR often uses a letter or token unit and it causes further unit mismatch during beam search. As described in previous sections, the classical approach of incorporating the lexical constraint from the token unit to the word unit is based on a WFST. This method first makes a TLG transducer composed of the token (T), word lexicon (L), and word-based language transducers (G) [74]. This TLG transducer has been used for both CTC [74] and attention-based [53] models.

Another approach used in the time synchronous beam search is to insert the word-based language model score triggered by the word delimiter (space) symbol [75]. To synchronize the word-based language model with a character-based end-to-end ASR, [260] combines the word and character-based LMs with the prefix tree representation, while [239], [261] uses look-ahead word probabilities to predict next characters instead of using the character-based LM. The prefix tree representation is also used for the sub-word token unit case [262], [263].

H. Multi-Pass Fusion

The previous fusion methods are performed during the beam search, which enables a one-pass algorithm. The popular alternative methods are based on multi-pass algorithms where we do not care about the synchronization and perform n-best or lattice scoring by considering the entire context within an utterance. [16] uses the N-best rescoring techniques to integrate a word-based language model. [55] combines forward and backward searches within a multi-pass decoding framework to combine bidirectional LSTM decoder networks. Recently two-pass algorithms of switching different end-to-end ASR systems have been investigated, including RNN-T → AED [264]; CTC → AED [265], [266]. This aims to provide streamed output in the first pass and re-scoring with AED in the second pass to refine the previous output, thus satisfying a real-time interface requirement while providing high recognition performance.

In addition to the N-best output in the above discussion, there is a strong demand for generating a lattice output for better multi-pass decoding thanks to richer hypothesis information in a lattice. The lattice output can also be used for spoken term detection, spoken language understanding, and word posteriors. However, due to the lack of Markov assumptions, RNN-T and AED cannot merge the hypothesis and cannot generate a lattice straightforwardly, unlike the HMM-based or CTC systems. To tackle this issue, there are several studies of modifying these models by limiting the output dependencies in the fixed length (i.e., finite-history) [47], [267], or keeping the original RNN-T structure but merging the similar hypotheses during beam search [107].

I. Vectorization Across Both Hypotheses and Utterances

We can accelerate the decoding process by vectorizing multiple hypotheses during the beam search, where we replace the score accumulation steps for each hypothesis with vector-matrix operations for the vectorized hypotheses. This has been studied in RNN-T [22], [258], [268] and attention-based [259] models. This modification leverages the parallel computing capabilities of multi-core CPUs, GPUs and TPUs, resulting in significant speedups, while enabling multiple utterances to be processed simultaneously in a batch. Major deep neural network and end-to-end ASR toolkits support this vectorization. For example, Tensorflow[9] [269], and FAIRESEQ[10] [270] provide a vectorized beam search interface for a generic sequence to sequence task, and it can be used for attention-based end-to-end ASR. End-to-end ASR toolkits including ESPnet[11] [259], ESPRESSO[12] [261], LINGVO [271], and, RETURNN[13] [272] also support the vectorized beam search algorithm.

Relationship to Classical ASR

One of the most prominent properties shared between E2E and classical statistical ASR systems is the use of a single-pass decoding strategy, which integrates all knowledge sources involved (models, components), before coming to a final decision [123]. This includes the use of full label context dependency both for E2E systems [51], [77], [174], [229], [262], [273], [274], [275], as well as classical systems via full-context language models [244], [245], [276], [277]. In classical ASR systems, even HMM alignment path summation may be retained in search [278]. Both E2E as well as classical ASR systems employ beam search in decoding. However, compared to classical search approaches, E2E beam search usually is highly simplified with very small beam sizes around 1 to 100 [15], [16], [77], [147]. Very small beam sizes also partly mask a length bias exhibited by E2E attention-based encoder-decoder models [279], [280], thus trading model errors against search errors [281]. An overview of approaches to handle the length bias beyond using small beam sizes in ASR is presented in [236].

[9][Online]. Available: https://www.tensorflow.org/api_docs/python/tf/contrib/seq2seq/BeamSearchDecoder

[10][Online]. Available: https://github.com/pytorch/fairseq/blob/master/fairseq/sequence_generator.py

[11][Online]. Available: https://github.com/espnet/espnet

[12][Online]. Available: https://github.com/freewym/espresso

[13][Online]. Available: https://github.com/rwth-i6/returnn

Many classical ASR search paradigms are based on multipass approaches that successively generate search space representations applying increasingly complex acoustic and/or language models [243], [282], [283]. However, multipass strategies also are employed using E2E models, which however softens the E2E concept. Decoder model combination is pursued in a two-pass approach, while even retaining latency constraints as in [87]. Further multipass approaches include E2E adaptation approaches [284], [285], [286], [287].

VII. LM Integration

This section discusses language models (LMs) used for E2E ASR. Hybrid ASR systems have long been using a pretrained LM [2], whereas most end-to-end (E2E) ASR systems employ a single E2E model that includes a network component acting as an LM.[14] For example, the prediction network of RNN-T and the decoder network of AED models take on the role of a LM covering label back-histories. Therefore, E2E ASR does not seem to require external LMs. Nevertheless, many studies have demonstrated that external LMs help improve the recognition accuracy in E2E ASR.

There are presumably three reasons that E2E ASR still requires an external LM:

a) Compensation for poor generalization: E2E models need to learn a more complicated mapping function than classical modular-based models such as acoustic models. Consequently, E2E models tend to face overfitting problems if the amount of training data is not sufficient. Pretrained LMs potentially compensate for the less generalized predictions made by E2E models.

b) Use of external text data: E2E models need to be trained using paired speech and text data, while LMs can be trained with only text data. Generally, text data can be collected more easily than the paired data. The training speed of an LM is also faster than that of E2E models for the same number of sentences. Accordingly, the LM can be improved more effectively with external text data, providing additional performance gain to the ASR system.

c) Domain adaptation: Domain adaptation helps improve recognition accuracy when the E2E model is applied to a specific domain. However, domain adaptation of the E2E model requires a certain amount of paired data in the target domain. Also, when multiple domains are assumed, it may be costly to maintain multiple E2E models for the domains the system supports. If a pretrained LM for the target domain is available, it may more easily improve recognition accuracy for domain-specific words and speaking styles without updating the E2E model.

This section reviews various types of LMs used for E2E ASR and fusion techniques to integrate LMs into E2E models.

A. Language Models

The LMs provide a prior probability distribution, $P(C)$. If the sentence, C, can be decomposed into a sequence of tokens such as characters, subwords, and single words, the probability distribution can be computed based on the chain rule as:

$$P(C) = \prod_{i=1}^{L+1} P(c_i|c_{0:i-1})$$

where c_i denotes the i-th token of C, and $c_{0:i-1}$ represents token sequence $c_0, c_1, \ldots, c_{i-1}$, assuming $c_0 = \langle\text{sos}\rangle$ and $c_{L+1} = \langle\text{eos}\rangle$.

Most LMs are designed to provide the conditional probability $P(c_i|c_{0:i-1})$, i.e., they are modeled to predict the next token given a sequence of the preceding tokens. We briefly review such LMs focusing on the different techniques to represent each token, c_i, and back-history, $c_{0:i-1}$.

1) N-Gram LM: N-gram LMs have long been used for ASR [2]. Early E2E systems in [53], [74], [77] also employed an N-gram LM. The N-gram models rely on the Markov assumption that the probability distribution of the next token depends only on the previous $N-1$ tokens, i.e., $P(c_i|c_{0:i-1}) \approx P(c_i|c_{i-N+1:i-1})$, where N is typically 3 to 5 for word-based models and higher for sub-word and character-based models. The maximum likelihood estimates of N-gram probabilities are determined based on the counts of N sequential tokens in the training data set as:

$$P(c_i|c_{i-N+1:i-1}) = \frac{\mathcal{K}(c_{i-N+1}, \ldots, c_i)}{\sum_{c_i} \mathcal{K}(c_{i-N+1}, \ldots, c_i)}$$

where, $\mathcal{K}(\cdot)$ denotes the count of each token sequence. Since the data size is finite, it is important to apply a smoothing technique to avoid estimating the probabilities based on zero or very small counts for rare token sequences. Those techniques compensate the N-gram probabilities with lower order models, e.g., $(N-1)$-gram models, according to the magnitude of the count [288]. However, since the N-gram probabilities still rely on the discrete representation of each token and the history, they suffer from data sparsity problems, leading to poor generalization.

The advantage of the N-gram models is their simplicity, although they underperform state-of-the-art neural LMs. In the training, the main step is to just count the N tuples in the data set, which is required only once. During decoding, the LM probabilities can be obtained very quickly by table lookup or can be attached to a decoding graph, e.g., WFST, in advance.

2) FNN-LM: The feed-forward neural network (FNN) LM was proposed in [9], which estimates N-gram probabilities using a neural network. The network accepts $N-1$ tokens, and predicts the next token as:

$$\begin{aligned} P(c_i|c_{i-N+1:i-1}) &= \text{softmax}(W_o h_i + b_o) \\ h_i &= \tanh(W_h e_i + b_h) \\ e_i &= \text{concat}(E(c_{i-N+1}), \ldots, E(c_{i-1})) \end{aligned}$$

where W_o and W_h are weight matrices, and b_o and b_h are bias vectors. $E(y)$ provides an embedding vector of c, and $\text{concat}(\cdot)$ operation concatenates given vectors.[15] This model first maps each input token to an embedding space, and then obtains hidden vector, h_i, as a context vector representing the previous

[14] In the simplest case of a CTC model as in Fig. 2, the included LM component however is limited to a label prior without label context.

[15] We omit the optional direct connection from the embedding layer to the softmax layer in [9] for simplicity.

$N-1$ tokens. Finally, it outputs the probability distribution of the next token through the $\mathrm{softmax}$ layer. Although this LM still relies on the Markov assumption, it outperforms classical N-gram LMs described in the previous section. The superior performance of FNN-LM is primarily due to the distributed representation of each token and the history. The LM learns to represent token/context vectors such that semantically similar tokens/histories are placed close to each other in the embedding space. Since this representation has a better smoothing effect than the count-based one used for N-gram LMs, FNN-LM can provide a better generalization than N-gram LMs for predicting the next token. Neural network-based LMs basically utilize this type of representation.

3) RNN-LM: A recurrent neural network (RNN) LM was introduced to exploit longer contextual information over $N-1$ previous tokens using recurrent connections [289]. Unlike FNN-LM, the hidden vector is computed as:

$$h_i = \mathrm{recurrence}(e_i, h_{i-1})$$
$$e_i = E(c_{i-1})$$

where, $\mathrm{recurrence}(e_i, h_{i-1})$ represents a recursive function, which accepts previous hidden vector h_{i-1} with input e_i, and outputs next hidden vector h_i. In the case of simple (Elman-type) RNN, the function can be computed as

$$\mathrm{recurrence}(e, h) = \tanh(W_h e + W_r h + b_h)$$

where, W_r is a weight matrix for the recurrent connection, which is applied to the previous hidden vector h. This recurrent loop makes it possible to hold the history information in the hidden vector without limiting the history to $N-1$ tokens. However, the history information decays exponentially as tokens are processed with this recursion. Therefore, currently stacked LSTM layers are more widely used for the recurrent network, which have separate internal memory cells and gating mechanisms to keep long-range history information [290]. With this mechanism, RNN-LMs outperform other N-gram-based models in many tasks.

4) ConvLM: Convolutional neural networks (ConvLM) have also been applied to LMs [291], [292], [293]. ConvLM [292] replace the recurrent connections used in RNN-LMs with gated temporal convolutions. The hidden vector is computed as

$$h_i = h'_i \otimes \sigma(g_i)$$
$$h'_i = e_{i-k+1:i} * W + b$$
$$g_i = e_{i-k+1:i} * V + c$$

where $\otimes$ is element-wise multiplication, $*$ is a temporal convolution operation, and k is the patch size. $\sigma(g_i)$ represents a gating function of convoluted activation h'_i, and is modeled as a sigmoid function. W and V are matrices for convolution and b and c are bias vectors. The convolution and gating blocks are typically stacked multiple times with residual connections. In [293], a ConvLM with 14 blocks has been applied for E2E ASR. Similar to FNN-LM, ConvLM allow us to use only a fixed history size, but they are more parameter efficient and easier to utilize longer histories than the FNN-LM by stacking the layers. Thus, they achieve competitive performance to that of RNN-LMs [292], even with the finite history consisting of short tokens such as characters [294]. Moreover, they are highly parallelizable and thus suitable for training the model with a large training data set.

5) Transformer LM: Transformer architecture [44] has been applied to LMs [295] and used for ASR [102], [296], where the LMs are designed as a Transformer decoder without any inputs from other modules such as encoders. The hidden vector is computed as:

$$h_i = \mathrm{FFN}(h'_i) + h'_i$$
$$h'_i = \mathrm{MHA}(e_i, e_{1:i}, e_{1:i}) + e_i$$

where $\mathrm{FFN}(\cdot)$ and $\mathrm{MHA}(\cdot,\cdot,\cdot)$ denote a feed forward network and a multi-head attention module, respectively. The multihead attention and feed-forward blocks are typically stacked multiple times, e.g., 6 times [102], to obtain the final hidden vector. The advantage of Transformer LMs is that they can take all tokens in the history into account through the self-attention mechanism without summarizing them into a fixed-size memory like RNN-LMs. Thus, the long history can be fully considered with attention to predict the next token, achieving better performance than RNN-LMs. However, the computational complexity increases quadratically as the length of the sequence. Therefore, the history length is typically limited to a fixed size or within every single sentence. To overcome this limitation, Transformer-XL [297] reuses already computed activations, which includes information on farther previous tokens, and the model is trained with a truncated back-propagation through time (BPTT) algorithm [298]. Compressive Transformer [299] extends this approach to utilize even longer contextual information by incorporating a compression step to keep older, but important, information in a fixed-size memory network.

B. *Fusion Approaches*

There are several ways to incorporate an external LM into E2E ASR, called *LM fusion*. Their purpose is to improve the recognition accuracy of E2E ASR by leveraging the benefits of the external LM described in the first part of this section. However, there can be a mismatch in the prediction between the E2E model and the LM when trained on different data sets, and therefore the LM may not collaborate well with the E2E model. Researchers have investigated various LM fusion approaches to reduce the mismatch between models in different situations.

1) Shallow Fusion: Shallow fusion is the most popular approach to combine the pretrained E2E model and LM in the inference time. As we described in Section VI-F, shallow fusion simply combines the E2E and LM scores by a log-linear combination as

$$\mathrm{Score}(C|X) = \log P(C|X) + \gamma \log P(C) \tag{9}$$

where γ is a scaling factor for the LM [255], [256], [257]. The advantage of this approach is that it is easy and effective when there are no major mismatches between the source and target domains.

2) Deep Fusion: Deep fusion [300] is an approach to combine an LM with an E2E model using a joint network. Given a pretrained E2E model and an LM, all the network parameters are fine-tuned jointly so that the models collaborate better to improve the recognition accuracy, where the joint network is used to combine the E2E and LM states through a gating mechanism that controls the contribution of the LM according to the current state.

3) Cold Fusion: Cold fusion [301] is another approach to combine a pretrained LM like deep fusion, but the E2E model is learned while freezing the LM parameters. Since the E2E model is aware of the LM throughout training, it learns to use the LM to reduce language specific information and capture only the relevant information to map the source to the target sequence. This mechanism reduces the role of LM in the E2E model and alleviates the language bias of the training data. Accordingly, the E2E model becomes more robust to domain mismatches between the training data and the target domain. Unlike deep fusion, cold fusion makes it possible to combine the E2E model with a pretrained LM for the target domain, improving the recognition accuracy. Component fusion [302] extends cold fusion to use a pretrained LM with transcriptions of the training data for the E2E model, more focusing on reducing the bias of the training data.

4) Internal LM Estimation: There is another approach to reduce language bias in training data through shallow fusion. The language bias is a problem when a big domain mismatch exists between the source domain (training data) and the target domain (test data) because the E2E model scores are strongly dependent on the language priors in the source domain. To remove such a bias from the score, we can explicitly estimate the LM that represents the language priors, called *Internal LM*, and subtract the LM score from the ASR score of (9):

$$\mathrm{Score}(C|X) = \log P_\varphi(C|X) - \gamma_\varphi \log P_\varphi(C) + \gamma_\tau \log P_\tau(C)$$

where subscripts φ and τ indicate the source and target domains, respectively. γ_φ and γ_τ are their scaling factors. Subtracting the internal LM score corresponds to approximating acoustic probability density $P_\varphi(X|C)$ because $P_\varphi(X|C) \propto P_\varphi(C|X)/P_\varphi(C)$ is satisfied for fixed X, where the ASR score can be seen as a classical hybrid ASR system. Accordingly, the subtracted E2E model score plays a role of acoustic model and makes it more domain independent in terms of language, achieving a higher recognition accuracy in combination with the external LM $P_\tau(C)$.

The density ratio method [303] trains an internal LM using the transcript of the training data. Hybrid autoregressive transducer (HAT) [47] extends RNN-T so that the model becomes the internal LM when the encoder output is eliminated, i.e., set to zero. This approach simplifies the framework by utilizing the prediction network as the internal LM, which avoids training an additional LM and using it in the inference time. In the work of [304], an approach similar to HAT has been proposed where the internal LM is formulated on top of standard RNN-T and attention-based encoder-decoder models, respectively. In [128], several techniques to estimate internal LMs have been proposed for AED models, where an estimated bias vector is fed to the LM instead of a zero vector. The bias vector can be estimated by averaging encoder states or context vectors, or by a small LSTM predicting the context vector based on the decoder label context, only. These techniques to estimate the internal LM were also evaluated for RNN-T in [305].

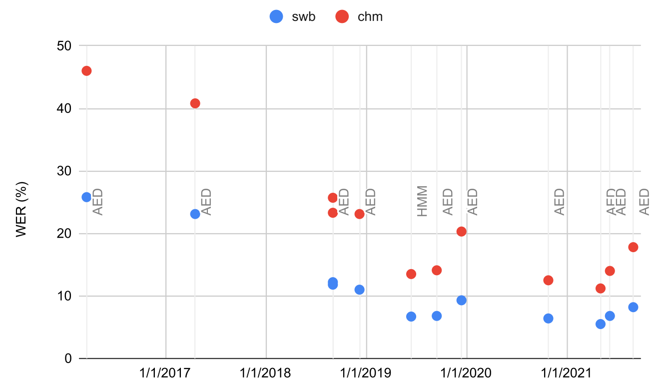

Fig. 9. E2E ASR performance improvement in the switchboard task.

C. *Use of Large-Scale Pretrained LMs*

In recent years, LMs trained with large-scale text data are available for different NLP tasks. BERT [306] and GPT-2 [307] are representative models based on Transformer LMs. Such LMs have also been applied to E2E ASR systems in different ways, e.g., N-best rescoring [308] and dialog context embedding [309].

Relationship to Classical ASR

The architecture of classical ASR systems provides a separation between the acoustic model and the language model. In contrast to this, E2E models avoid this separation and define a joint model. While this allows for training with a single objective, it limits training of the (implicit) prior to the transcriptions of the audio training data. To exploit further text-only training data, usually a separate LM is combined with E2E models, nonetheless. However, due to the implicit prior of E2E models, i.e. the internal language model, combination with separate language models is not straightforward and requires corresponding internal language model estimation and compensation approaches, e.g. [47], [128], [303], [304], [310]. At least from the recognition accuracy perspective, it remains unclear, if the clear separation of acoustic modeling and language modeling in the classical ASR architecture is a disadvantage because of separate training objectives, or rather an advantage, since text-only training data may be used easily. Also, the language model training objective, i.e. language model perplexity, is observed to correlate well with word error rate [311], [312], [313], [314]. Furthermore, discriminative approaches to language modeling [315] may be viewed as a step towards joint modeling.

VIII. Overall Performance Trends of E2E Approaches in Common Benchmarks

This section summarizes various techniques with the common ASR benchmarks based on switchboard (SWBD) [316] in Fig. 9 and Librispeech [317] in Fig. 10 to see the trajectory of the techniques developed in end-to-end ASR. We choose these two databases because they are widely used in speech and machine learning communities and cover spontaneous (SWBD) and read speech (Librispeech) speaking styles. Figs. 9 and 10 show that the performance improvement relative to the initial works [79],

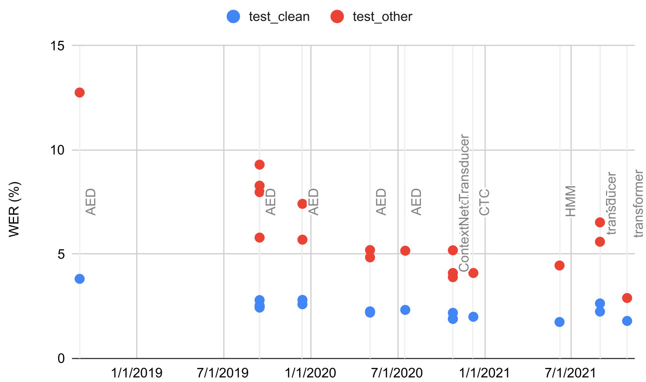

Fig. 10. E2E ASR performance improvement in the Librispeech task.

[147] based on the E2E models is significant, and the error rates of all tasks become less than half of the original error rates![16]

Although the overall trends show that the ASR performance has steadily improved over time, there are several remarkable gains. One significant gain observed in both benchmarks in the middle of 2019 comes from the data augmentation method represented by SpecAugment [205], [206], as discussed in Section V-G. The subsequent gains mostly come from the exploration of the new neural network architectures, including transformer [102], [318], conformer [45], [103], and contextnet [97] on top of SpecAugment, as discussed in Section IV-C. Such an exploration is also performed in language modeling to improve the ASR performance [102], [296]. The final gain observed in the Librispeech benchmark in 2021 is based on self-supervised learning [25], [319] and semi-supervised learning [320], [321]. These techniques utilize a considerable amount of unlabeled in-domain speech data (e.g., Libri-light 60 K hours [322]).

Relationship to Classical ASR

Speech recognition research has always been pushed by international evaluation campaigns (e.g. as lead by NIST) and corresponding benchmark tasks. The competition between classical and E2E approaches is nicely reflected in the widely used Librispeech [317] and Switchboard [316] tasks, showing that E2E models gain momentum. As shown in Fig. 10, on Librispeech, the current best-published classical hybrid systems range around 2.3% (test-clean) and 4.9% (test-other) word error rate [222], [323], while there already are a number of E2E systems providing similar performance [205], [206], [224], [320], with some E2E systems clearly outperforming former state-of-the-art results with word error rates down to 1.8% (test-clean) and 3.7% (test-other) [324] with similar results reported in [45], [97]. Merging insights from classical HMM-based and monotonic RNN-T provided similarly well results with a limited training budget [124]. Finally, when trained on Switchboard 300 h, the current best result, obtained with an E2E system [180] is 5.4% compared to 6.6% word error rate for the best hybrid system result [325] on the HUB5'00 Switchboard test set, in Fig. 9.

[16]For readers who want to know the latest update of these benchmarks can also check https://github.com/syhw/wer_are_we and https://github.com/thu-spmi/ASR-Benchmarks/blob/main/README.md.

IX. Deployment of E2E Models

Many of the ideas discussed in this paper have been explored by various industry research labs [265], [326], [327], [328], [329], [330], [331], *inter alia*. In this section, we review the development of on-device production-level systems at Google as a typical case study for deployment.

The first streaming E2E model, deployed to production, was launched in 2019 for the Pixel 4 smartphone [22], [332]. This model used a streaming RNN-T first-pass system, while re-scoring first-pass hypotheses with an AED system in the second pass. In addition, FST-based contextual biasing [92] was employed in the model, which was critical to obtain accurate results for diverse queries. This model ran on CPU and was much faster than real time.

In 2020, for the Pixel 5 smartphone [333], the system was improved further to reduce user-perceived latency (i.e., the time between when the user speaks, and when words appear on the device). This included advancements such as end-to-end endpointing [113] to encourage faster microphone closing; as well as FastEmit [91] to encourage the model to emit tokens earlier.

Finally, in 2021 the model was further improved for the Pixel 6 smartphone [334], to take advantage of the tensor processing unit (TPU) [85] on the device. Improvements include the use of conformer layers for the encoder [45]; a small embedding prediction network for the decoder [104]; a 2-pass cascaded encoder to run a 2nd-pass beam search [89]; and, a neural LM re-scorer to help improve accuracy long-tail named entities. This model is the best ASR system that Google has released to date, both in terms of quality and latency.

X. Areas for Future Work

Currently, E2E models dominate the academic debate on ASR. However, at least partly, this is not (yet?) reflected in the corresponding commercial deployment of E2E ASR architectures. E2E models are not yet the perfect match for all ASR conditions and further research is needed to take full advantage of the benefits of E2E modeling.

E2E models seem to perform really well when training data is abundant, while not scaling well to low-resource conditions. Similarly, domain change requires a flexible exchange of language models, which is natural for classical ASR models based on a separation of acoustic and language models. Ongoing research on the use of external language models in E2E models and internal language model estimation already is promising, but can be expected to see further improvements.

Top E2E ASR systems usually require orders of magnitude more training epochs than comparable classical ASR systems, and further research into efficient and robust optimization and training schedules is needed.

The high level of integration of E2E models also involves a loss in modularity, which might support the explainability and reusability of models. Also, more efficient training schedules might take advantage of modularity. One assumed advantage of E2E models is that everything is trained from data and secondary knowledge sources (e.g. pronunciation lexica and phoneme sets) are avoided. However, rare events, like rare words in ASR still provide a challenge, which needs further research.

With the missing separation of acoustic and language models, the question arises of how to exploit text-only resources in E2E model training - do we foresee solutions beyond training data generation using TTS? We note that a number of recent works have explored approaches to combine speech and text modalities by attempting to implicitly or explicitly map them into a shared space [159], [335], [336], [337], [338], [339], [340], [341]. Furthermore, high-performance E2E solutions exist for both discriminative problems like ASR, as well as generative problems like TTS, how can both be exploited jointly to support semi-supervised training based on text-only and/or audio-only data on top of transcribed speech audio [28], [342]?

For AED architectures, we observe a length bias, which complicates the decoding process. Although many heuristics are known to tackle length bias in AED, we are still missing a well-founded explanation for it, as well as a corresponding remedy of the original model.

Other open research problems include speaker adaptation and robustness to recording conditions, especially in mismatch situations. The E2E principle also provides a promising candidate to solve multichannel ASR by providing an E2E solution jointly tackling the source separation, speaker diarization and speech recognition problem [26], [343].

Finally, we need to investigate, if E2E is a suitable guiding principle, and how different E2E ASR models relate to each other as well as to classical ASR approaches. The most important guiding principle of ASR research and development has been performance, and ASR has been boosted strongly by widely used benchmark tasks and international evaluation campaigns. With the current diversity of classical and E2E models, we also need to resolve the question of what constitutes state-of-the-art in ASR today, and can we expect a common state-of-the-art ASR architecture in the future?

XI. Conclusion

In this work, we presented a detailed overview of end-to-end approaches to ASR. Such models, which have grown in popularity over the last few years, propose to use highly integrated neural network components which allow input speech to be converted directly into output text sequences through character-based output units. Thus, such models eschew the classical modular ASR architecture consisting of an acoustic model, a pronunciation model, and a language model, in favor of a single compact structure, and rely on the data to learn effectively. These design choices enable the deployment of highly accurate on-device speech recognition models (see Section IX), but also come with a number of downsides which are still areas of active research (see Section X).

Finally, we direct interested readers to Li's excellent contemporaneous overview article on end-to-end ASR [344], which offers a complementary perspective to our own. In particular, readers of [344] may find a more detailed exposition on the choice of encoder structure, and the applications of E2E approaches to allied ASR areas (e.g., multi-speaker recognition; multilingual ASR; adaptation to new application domains, and speakers; etc.), which we do not cover due to space limitations.

Acknowledgment

The authors would like to thank Julian Dierkes, Yifan Peng, Zoltán Tüske, Albert Zeyer, and Wei Zhou for their help on refining our manuscript.

References

[1] T. Bayes, "An essay towards solving a problem in the doctrine of chances," *Philos. Trans. Roy. Soc. London*, vol. 53, pp. 370–418, 1763.
[2] F. Jelinek, *Statistical Methods for Speech Recognition*. Cambridge, MA, USA: MIT Press, 1997.
[3] L. R. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition," *Proc. IEEE*, vol. 77, no. 2, pp. 257–286, Feb. 1989.
[4] H. A. Bourlard and N. Morgan, *Connectionist Speech Recognition: A Hybrid Approach*. Norwell, MA, USA: Kluwer Academic Publishers, 1993.
[5] F. Seide, G. Li, and D. Yu, "Conversational speech transcription using context-dependent deep neural networks," in *Proc. Interspeech*, 2011, pp. 437–440.
[6] V. Fontaine, C. Ris, and H. Leich, "Nonlinear discriminant analysis for improved speech recognition," in *Proc. Eurospeech*, 1997, pp. 1–4.
[7] H. Hermansky, D. P. W. Ellis, and S. Sharma, "Tandem connectionist feature extraction for conventional HMM systems," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2000, vol. 3, pp. 1635–1638.
[8] M. Nakamura and K. Shikano, "A study of english word category prediction based on neural networks," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 1989, pp. 731–734.
[9] Y. Bengio, R. Ducharme, and P. Vincent, "A neural probabilistic language model," in *Proc. Neural Inf. Process. Syst.*, 2000, pp. 932–938.
[10] H. Schwenk and J.-L. Gauvain, "Connectionist language modeling for large vocabulary continuous speech recognition," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2002, pp. 765–768.
[11] Z. Tüske, P. Golik, R. Schlüter, and H. Ney, "Acoustic modeling with deep neural networks using raw time signal for LVCSR," in *Proc. Interspeech*, 2014, pp. 890–894.
[12] T. N. Sainath, R. J. Weiss, K. W. Wilson, A. Narayanan, M. Bacchiani, and A. Senior, "Speaker location and microphone spacing invariant acoustic modeling from raw multichannel waveforms," in *Proc. IEEE Autom. Speech Recognit. Understanding*, 2015, pp. 30–36.
[13] 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.
[14] A. Graves, "Sequence transduction with recurrent neural networks," in *Proc. Int. Conf. Mach. Learn.*, Edinburgh, Scotland, Jun. 2012.
[15] 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.
[16] W. Chan, N. Jaitly, Q. Le, and O. Vinyals, "Listen, attend and spell: A neural network for large vocabulary conversational speech recognition," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2016, pp. 4960–4964.
[17] P. Liang, A. Bouchard-Côté, D. Klein, and B. Taskar, "An end-to-end discriminative approach to machine translation," in *Proc. Assoc. Comput. Linguistics*, 2006, pp. 761–768.
[18] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa, "Natural language processing (Almost) from scratch," *J. Mach. Learn. Res.*, vol. 12, pp. 2493–2537, 2011.
[19] 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.
[20] "Cambridge Dictionary," Accessed: Feb. 21, 2020. [Online]. Available: https://dictionary.cambridge.org/dictionary/english/end-to-end
[21] R. Pang et al., "Compression of end-to-end models," in *Proc. Interspeech*, 2018, pp. 27–31.
[22] Y. He et al., "Streaming end-to-end speech recognition for mobile devices," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, Brighton, U.K., 2019, pp. 6381–6385.

[23] R. Schlüter and H. Ney, "Model-based MCE bound to the true bayes' error," *IEEE Signal Process. Lett.*, vol. 8, no. 5, pp. 131–133, May 2001.
[24] H. Ney, "On the relationship between classification error bounds and training criteria in statistical pattern recognition," in *Proc. Iberian Conf. Pattern Recognit. Image Anal.*, 2003, pp. 636–645.
[25] A. Baevski, Y. Zhou, A. Mohamed, and M. Auli, ""wav2vec 2.0: A framework for self-supervised learning of speech representations," in *Proc. Adv. Neural Inf. Process. Syst.*, 2020, pp. 12449–12460.
[26] X. Chang, W. Zhang, Y. Qian, J. Le Roux, and S. Watanabe, "MIMO-Speech: End-to-end multi-channel multi-speaker speech recognition," in *Proc. IEEE Autom. Speech Recognit. Understanding*, 2019, pp. 237–244.
[27] L. Breiman, J. Friedman, C. Stone, and R. Olshen, *Classication and Regression Trees*. Belmont, CA, USA: Taylor & Francis, 1984.
[28] A. Tjandra, S. Sakti, and S. Nakamura, "Listening while speaking: Speech chain by deep learning," in *Proc. IEEE Autom. Speech Recognit. Understanding Workshop*, Dec. 2017, pp. 301–308.
[29] M. K. Baskar, S. Watanabe, R. Astudillo, T. Hori, L. Burget, and J. Černocký, "Semi-supervised sequence-to-sequence ASR using unpaired speech and text," in *Proc. Interspeech*, 2019, pp. 3790–3794.
[30] 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.
[31] G. K. Zipf, *Human Behavior and the Principle of Least Effort*. Boston, MA, USA: Addison-Wesley Press, 1949.
[32] R. Sennrich, B. Haddow, and A. Birch, "Neural machine translation of rare words with subword units," in *Proc. Assoc. Comput. Linguistics*, 2015, pp. 1715–1725.
[33] W. Chan, Y. Zhang, Q. Le, and N. Jaitly, "Latent sequence decompositions," in *Proc. Int. Conf. Learn. Representations*, 2017. [Online]. Available: https://openreview.net/forum?id=SyQq185lg
[34] H. Liu, Z. Zhu, X. Li, and S. Satheesh, "Gram-CTC: Automatic unit selection and target decomposition for sequence labelling," in *Proc. Int. Conf. Mach. Learn.*, Aug. 2017, pp. 2188–2197.
[35] H. Xu, S. Ding, and S. Watanabe, "Improving end-to-end speech recognition with pronunciation-assisted sub-word modeling," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2019, pp. 7110–7114.
[36] W. Zhou, M. Zeineldeen, Z. Zheng, R. Schlüter, and H. Ney, "Acoustic data-driven subword modeling for end-to-end speech recognition," in *Proc. Interspeech*, 2021, pp. 2886–2890.
[37] M. Schuster and K. Nakajima, "Japanese and Korean voice search," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2012, pp. 5149–5152.
[38] M. Mohri, F. Pereira, and M. Riley, "Weighted finite-state transducers in speech recognition," *Comput. Speech Lang.*, vol. 16, no. 1, pp. 69–88, 2002.
[39] E. Beck, M. Hannemann, P. Doetsch, R. Schlüter, and H. Ney, "Segmental encoder-decoder models for large vocabulary automatic speech recognition," in *Proc. Interspeech*, 2018, pp. 766–770.
[40] W. Zhou, A. Zeyer, A. Merboldt, R. Schlüter, and H. Ney, "Equivalence of segmental and neural transducer modeling: A proof of concept," in *Proc. Interspeech*, 2021, pp. 2891–2895.
[41] R. Prabhavalkar, K. Rao, T. N. Sainath, B. Li, L. Johnson, and N. Jaitly, "A comparison of sequence-to-sequence models for speech recognition," in *Proc. Interspeech*, 2017, pp. 939–943.
[42] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.
[43] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," in *Proc. Int. Conf. Learn. Representations*, 2015.
[44] A. Vaswani et al., "Attention is all you need," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 5998–6008.
[45] A. Gulati et al., "Conformer: Convolution-augmented transformer for speech recognition," in *Proc. Interspeech*, 2020, pp. 5036–5040.
[46] H. Sak, M. Shannon, K. Rao, and F. Beaufays, "Recurrent neural aligner: An encoder-decoder neural network model for sequence to sequence mapping," in *Proc. Interspeech*, 2017, pp. 1298–1302.
[47] E. Variani, D. Rybach, C. Allauzen, and M. Riley, "Hybrid autoregressive transducer (HAT)," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2020, pp. 6139–6143.
[48] A. Graves, A.-r. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2013, pp. 6645–6649.
[49] N. Moritz, T. Hori, S. Watanabe, and J. Le Roux, "Sequence transduction with graph-based supervision," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2022, pp. 7212–7216.
[50] Y. Bengio, N. Léonard, and A. Courville, "Estimating or propagating gradients through stochastic neurons for conditional computation," 2013, *arXiv:1308.3432*.
[51] A. Tripathi, H. Lu, H. Sak, and H. Soltau, "Monotonic recurrent neural network transducer and decoding strategies," in *Proc. IEEE Autom. Speech Recognit. Understanding*, 2019, pp. 944–948.
[52] A. Zeyer, A. Merboldt, R. Schlüter, and H. Ney, "A new training pipeline for an improved neural transducer," in *Proc. Interspeech*, 2020, pp. 2812–2816.
[53] 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.
[54] Y. Wu et al., "Google's neural machine translation system: Bridging the gap between human and machine translation," 2016, *arXiv:1609.08144*.
[55] M. Mimura, S. Sakai, and T. Kawahara, "Forward-backward attention decoder," in *Proc. Interspeech*, 2018, pp. 2232–2236.
[56] A. Graves, "Generating sequences with recurrent neural networks," 2013, *arXiv:1308.0850*.
[57] J. Hou, S. Zhang, and L.-R. Dai, "Gaussian prediction based attention for online end-to-end speech recognition," in *Proc. Interspeech*, 2017, pp. 3692–3696, doi: 10.21437/Interspeech.2017-751.
[58] C.-C. Chiu et al., "A comparison of end-to-end models for long-form speech recognition," in *Proc. IEEE Autom. Speech Recognit. Understanding*, 2019, pp. 889–896.
[59] N. Jaitly, Q. V. Le, O. Vinyals, I. Sutskever, D. Sussillo, and S. Bengio, "An online sequence-to-sequence model using partial conditioning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, pp. 5067–5075.
[60] C. Raffel, M.-T. Luong, P. J. Liu, R. J. Weiss, and D. Eck, "Online and linear-time attention by enforcing monotonic alignments," in *Proc. Int. Conf. Mach. Learn.*, 2017, pp. 2837–2846.
[61] C.-C. Chiu and C. Raffel, "Monotonic Chunkwise attention," in *Proc. Int. Conf. Learn. Representations*, 2018.
[62] N. Arivazhagan et al., "Monotonic infinite lookback attention for simultaneous machine translation," in *Proc. Assoc. Comput. Linguistics*, 2019, pp. 1313–1323.
[63] T. N. Sainath et al., "Improving the performance of online neural transducer models," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2018, pp. 5864–5868.
[64] N. Moritz, T. Hori, and J. Le Roux, "Triggered attention for end-to-end speech recognition," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2019, pp. 5666–5670.
[65] A. Merboldt, A. Zeyer, R. Schlüter, and H. Ney, "An analysis of local monotonic attention variants," in *Proc. Interspeech*, 2019, pp. 1398–1402.
[66] A. Zeyer, R. Schlüter, and H. Ney, "A Study of Latent Monotonic Attention Variants," Mar. 2021, *arXiv:2103.16710*.
[67] A. Zeyer, R. Schmitt, W. Zhou, R. Schlüter, and H. Ney, "Monotonic segmental attention for automatic speech recognition," in *Proc. IEEE Spoken Lang. Technol. Workshop*, 2023, pp. 229–236.
[68] Z. Tian, J. Yi, Y. Bai, J. Tao, S. Zhang, and Z. Wen, "Synchronous transformers for end-to-end speech recognition," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2020, pp. 7884–7888.
[69] D. Povey et al., "Purely sequence-trained neural networks for ASR based on lattice-free MMI," in *Proc. Interspeech*, 2016, pp. 2751–2755, doi: 10.21437/Interspeech.2016-595.
[70] R. Collobert, C. Puhrsch, and G. Synnaeve, "Wav2Letter: An end-to-end convnet-based speech recognition system," 2016, *arXiv:1609.03193*.
[71] P. Haffner, "Connectionist speech recognition with a global MMI algorithm," in *Proc. Eurospeech*, 1993, pp. 1929–1932.
[72] A. Zeyer, E. Beck, R. Schlüter, and H. Ney, "CTC in the context of generalized full-sum HMM training," in *Proc. Interspeech*, 2017, pp. 944–948.
[73] T. Raissi, W. Zhou, S. Berger, R. Schlüter, and H. Ney, "HMM vs. CTC for automatic speech recognition: Comparison based on full-sum training from scratch," in *Proc. IEEE Spoken Lang. Technol. Workshop*, 2023, pp. 287–294.
[74] Y. Miao, M. Gowayyed, and F. Metze, "EESEN: End-to-end speech recognition using deep RNN models and WFST-Based decoding," in *Proc. IEEE Autom. Speech Recognit. Understanding*, 2015, pp. 167–174.
[75] A. Hannun et al., "Deep Speech: Scaling up End-to-End Speech Recognition," 2014, *arXiv:1412.5567*.
[76] 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.

[77] J. Chorowski and N. Jaitly, "Towards better decoding and language model integration in sequence to sequence models," in *Proc. Interspeech*, 2017, pp. 523–527.
[78] Y. Zhang, W. Chan, and N. Jaitly, "Very deep convolutional networks for end-to-end speech recognition," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2017, pp. 4845–4849.
[79] S. Toshniwal, H. Tang, L. Lu, and K. Livescu, "Multitask learning with low-level auxiliary tasks for encoder-decoder based speech recognition," in *Proc. Interspeech*, 2017, pp. 3532–3536.
[80] A. Renduchintala, S. Ding, M. Wiesner, and S. Watanabe, "Multi-modal data augmentation for end-to-end ASR," in *Proc. Interspeech*, 2018, pp. 2394–2398.
[81] S. Sabour, W. Chan, and M. Norouzi, "Optimal completion distillation for sequence learning," in *Proc. Int. Conf. Learn. Representations*, 2019.
[82] C. Weng et al., "Improving attention based sequence-to-sequence models for end-to-end english conversational speech recognition," in *Proc. Interspeech*, 2018, pp. 761–765.
[83] D. Le, X. Zhang, W. Zheng, C. Fügen, G. Zweig, and M. L. Seltzer, "From senones to chenones: Tied context-dependent graphemes for hybrid speech recognition," in *Proc. IEEE Autom. Speech Recognit. Understanding Workshop*, 2019, pp. 457–464.
[84] S. Kanthak and H. Ney, "Context-dependent acoustic modeling using graphemes for large vocabulary speech recognition," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2002, pp. 845–848.
[85] N. P. Jouppi et al., "In-datacenter performance analysis of a tensor processing unit," in *Proc. 44th Annu. Int. Symp. Comput. Architecture*, 2017, pp. 1–12.
[86] S. Watanabe, T. Hori, S. Kim, J. R. Hershey, and T. Hayashi, "Hybrid CTC attention architecture for end-to-end speech recognition," *IEEE J. Sel. Topics Signal Process.*, vol. 11, no. 8, pp. 1240–1253, Dec. 2017.
[87] T. N. Sainath et al., "Two-pass end-to-end speech recognition," in *Proc. Interspeech*, 2019, pp. 2773–2777.
[88] K. Hu, T. N. Sainath, R. Pang, and R. Prabhavalkar, "Deliberation model based two-pass end-to-end speech recognition," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2020, pp. 7799–7803.
[89] A. Narayanan et al., "Cascaded encoders for unifying streaming and non-streaming ASR," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2021, pp. 5629–5633.
[90] A. Tripathi, J. Kim, Q. Zhang, H. Lu, and H. Sak, "Transformer transducer: One model unifying streaming and non-streaming speech recognition," 2020, *arXiv:2010.03192*.
[91] J. Yu et al., "Universal ASR: Unify and improve streaming asr with full-context modeling," in *Proc. Int. Conf. Learn. Representations*, 2021.
[92] D. Zhao et al., "Shallow-fusion end-to-end contextual biasing," in *Proc. Interspeech*, 2019, pp. 1418–1422.
[93] G. Pundak, T. N. Sainath, R. Prabhavalkar, A. Kannan, and D. Zhao, "Deep context: End-to-end contextual speech recognition," in *Proc. IEEE Spoken Lang. Technol. Workshop*, 2018, pp. 418–425.
[94] S. Kim and F. Metze, "Dialog-context aware end-to-end speech recognition," in *Proc. IEEE Spoken Lang. Technol. Workshop*, 2018, pp. 434–440.
[95] A. Bruguier, R. Prabhavalkar, G. Pundak, and T. N. Sainath, "Phoebe: Pronunciation-aware contextualization for end-to-end speech recognition," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2019, pp. 6171–6175.
[96] M. Delcroix, S. Watanabe, A. Ogawa, S. Karita, and T. Nakatani, "Auxiliary feature based adaptation of end-to-end ASR systems," in *Proc. Interspeech*, 2018, pp. 2444–2448.
[97] W. Han et al., "ContextNet: Improving convolutional neural networks for automatic speech recognition with global context," in *Proc. Interspeech*, 2020, pp. 3610–3614.
[98] L. Dong, S. Xu, and B. Xu, "Speech-transformer: A. no-recurrence sequence-to-sequence model for speech recognition," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2018, pp. 5884–5888.
[99] Q. Zhang et al., "Transformer transducer: A streamable speech recognition model with transformer encoders and RNN-T loss," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2020, pp. 7829–7833.
[100] C.-F. Yeh et al., "Transformer-transducer: End-to-snd speech recognition with self-attention," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2019, pp. 7829–7833.
[101] Y. Peng, S. Dalmia, I. Lane, and S. Watanabe, "Branchformer: Parallel MLP-attention architectures to capture local and global context for speech recognition and understanding," in *Proc. Int. Conf. Mach. Learn.*, 2022, pp. 17627–17643.
[102] S. Karita et al., "A comparative study on transformer vs RNN in speech applications," in *Proc. IEEE Autom. Speech Recognit. Understanding Workshop*, 2019, pp. 449–456.
[103] P. Guo et al., "Recent Developments on ESPNET Toolkit Boosted by Conformer," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2021, pp. 5874–5878.
[104] R. Botros, T. Sainath, R. David, E. Guzman, W. Li, and Y. He, "Tied & reduced RNN-T decoder," in *Proc. Interspeech*, 2021, pp. 4563–4567.
[105] M. Ghodsi, X. Liu, J. Apfel, R. Cabrera, and E. Weinstein, "RNN-Transducer with stateless prediction network," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2020, pp. 7049–7053.
[106] W. Zhou, S. Berger, R. Schlüter, and H. Ney, "Phoneme based neural transducer for large vocabulary speech recognition," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2021, pp. 5644–5648.
[107] R. Prabhavalkar et al., "Less is more: Improved RNN-T decoding using limited label context and path merging," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2021, pp. 5659–5663.
[108] X. Chen, Z. Meng, S. Parthasarathy, and J. Li, "Factorized neural transducer for efficient language model adaptation," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2022, pp. 8132–8136.
[109] Z. Meng et al., "Modular hybrid autoregressive transducer," in *Proc. IEEE Spoken Lang. Technol. Workshop*, 2023, pp. 197–204.
[110] T. Wang et al., "VioLA: Unified codec language models for speech recognition, synthesis, and translation," 2023, *arXiv:2305.16107*.
[111] P. K. Rubenstein et al., "AudioPaLM: A large language model that can speak and listen," 2023, *arXiv:2306.12925*.
[112] S.-Y. Chang, B. Li, and G. Simko, "A unified endpointer using multitask and multidomain training," in *Proc. IEEE Autom. Speech Recognit. Understanding Workshop*, 2019, pp. 100–106.
[113] B. Li et al., "Towards fast and accurate streaming end-to-end ASR," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2020, pp. 6069–6073.
[114] T. Yoshimura, T. Hayashi, K. Takeda, and S. Watanabe, "End-to-End automatic speech recognition integrated with CTC-Based voice activity detection," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2020, pp. 6999–7003.
[115] Y. Fujita, T. Wang, S. Watanabe, and M. Omachi, "Toward streaming ASR with non-autoregressive insertion-based model," in *Proc. Interspeech*, 2021, pp. 3740–3744.
[116] Y. Bengio, "Practical recommendations for gradient-based training of deep architectures," in *Neural Networks: Tricks of the Trade*, 2nd ed., Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, *arXiv:1206.5533*.
[117] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural Netw.*, vol. 61, pp. 85–117, Jan. 2015.
[118] L. Baum, "An inequality and associated maximization technique in statistical estimation for probabilistic functions of Markov processes," *Inequalities*, vol. 3, pp. 1–8, 1972.
[119] L. Rabiner and B.-H. Juang, "An introduction to hidden Markov models," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. 3, no. 1, pp. 4–16, Jan. 1986.
[120] Y. Bengio, R. De Mori, G. Flammia, and R. Kompe, "Neural network-Gaussian mixture hybrid for speech recognition or density estimation," in *Proc. Adv. Neural Inf. Process. Syst.*, 1991, pp. 175–182.
[121] R. E. Bellman, *Dynamic Programming*. Princeton, NJ, USA: Princeton Univ. Press, 1957.
[122] A. Viterbi, "Error bounds for convolutional codes and an asymptotically optimal decoding algorithm," *IEEE Trans. Inf. Theory*, vol. 13, pp. 260–269, Apr. 1967.
[123] H. Ney, "The use of a one-stage dynamic programming algorithm for connected word recognition," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. 32, no. 2, pp. 263–271, Apr. 1984.
[124] W. Zhou, W. Michel, R. Schlüter, and H. Ney, "Efficient training of neural transducer for speech recognition," in *Proc. Interspeech, Incheon, Korea*, 2022, pp. 2058–2062.
[125] A. Zeyer, R. Schlüter, and H. Ney, "Why does CTC result in Peaky behavior?," 2021, *arXiv:2105.14849*.
[126] A. Laptev, S. Majumdar, and B. Ginsburg, "CTC variations through new WFST topologies," in *Proc. Interspeech*, 2022, pp. 1041–1045, doi: 10.21437/interspeech.2022-10854.
[127] X. He, L. Deng, and W. Chou, "Discriminative learning in sequential pattern recognition–a unifying review for optimization-oriented speech recognition," *IEEE Signal Process. Mag.*, vol. 25, no. 5, pp. 14–36, Sep. 2008.

[128] M. Zeineldeen, A. Glushko, W. Michel, A. Zeyer, R. Schlüter, and H. Ney, "Investigating methods to improve language model integration for attention-based encoder-decoder ASR models," in *Proc. Interspeech*, 2021, pp. 2856–2860.

[129] N.-P. Wynands, W. Michel, J. Rosendahl, R. Schlüter, and H. Ney, "Efficient sequence training of attention models using approximative recombination," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2022, pp. 8002–8006.

[130] Z. Yang, W. Zhou, R. Schlüter, and H. Ney, "Lattice-free sequence discriminative training for phoneme-based neural transducers," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2023, pp. 1–5.

[131] V. Valtchev, J. J. Odell, P. C. Woodland, and S. J. Young, "MMIE training of large vocabulary recognition systems," *Speech Commun.*, vol. 22, no. 4, pp. 303–314, 1997.

[132] D. Povey and P. Woodland, "Improved discriminative training techniques for large vocabulary continuous speech recognition," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2001, pp. 45–48.

[133] R. Schlüter, W. Macherey, B. Müller, and H. Ney, "Comparison of discriminative training criteria and optimization methods for speech recognition," *Speech Commun.*, vol. 34, no. 3, pp. 287–310, May 2001.

[134] B. Kingsbury, "Lattice-based optimization of sequence classification criteria for neural-network acoustic modeling," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2009, pp. 3761–3764.

[135] G. Heigold, R. Schlüter, H. Ney, and S. Wiesler, "Discriminative training for automatic speech recognition: Modeling, criteria, optimization, implementation, and performance," *IEEE Signal Process. Mag.*, vol. 29, no. 6, pp. 58–69, Nov. 2012.

[136] W. Michel, R. Schlüter, and H. Ney, "Comparison of lattice-free and lattice-based sequence discriminative training criteria for LVCSR," in *Proc. Interspeech*, 2019, pp. 1601–1605.

[137] R. Prabhavalkar et al., "Minimum word error rate training for attention-based sequence-to-sequence models," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2018, pp. 4839–4843.

[138] C. Weng, C. Yu, J. Cui, C. Zhang, and D. Yu, "Minimum bayes risk training of RNN-Transducer for end-to-end speech recognition," in *Proc. Interspeech*, 2020, pp. 966–970, doi: 10.21437/Interspeech.2020-1221.

[139] M. K. Baskar, L. Burget, S. Watanabe, M. Karafiát, T. Hori, and J. H. Černockỳ, "Promising Accurate Prefix Boosting for Sequence-to-Sequence ASR," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, May 2019, pp. 5646–5650.

[140] A. Tjandra, S. Sakti, and S. Nakamura, "Sequence-to-sequence asr optimization via reinforcement learning," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2018, pp. 5829–5833.

[141] S. Karita, A. Ogawa, M. Delcroix, and T. Nakatani, "Sequence training of encoder-decoder model using policy gradient for end-to-end speech recognition," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2018, pp. 5839–5843.

[142] W. Michel, R. Schlüter, and H. Ney, "Early stage LM integration using local and global log-linear combination," in *Proc. Interspeech*, 2020, pp. 3605–3609.

[143] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets," *Neural Comput.*, vol. 18, no. 7, pp. 1527–1554, Jul. 2006.

[144] Y. Bengio, P. Lamblin, D. Popovici, and H. Larochelle, "Greedy layer-wise training of deep networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2006, pp. 153–160.

[145] A. Zeyer, P. Doetsch, P. Voigtlaender, R. Schlüter, and H. Ney, "A comprehensive study of deep bidirectional LSTM RNNs for acoustic modeling in speech recognition," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2017, pp. 2462–2466.

[146] A. Zeyer, T. Alkhouli, and H. Ney, "RETURNN as a generic flexible neural toolkit with application to translation and speech recognition," in *Proc. Assoc. Comput. Linguistics*, 2018, pp. 128–133.

[147] A. Zeyer, K. Irie, R. Schlüter, and H. Ney, "Improved training of end-to-end attention models for speech recognition," in *Proc. Interspeech*, 2018, pp. 7–11.

[148] A. Zeyer, A. Merboldt, R. Schlüter, and H. Ney, "A comprehensive analysis on attention models," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018.

[149] Y. Chung, C. Wu, C. Shen, H. Lee, and L. Lee, "Audio Word2Vec: Unsupervised learning of audio segment representations using sequence-to-sequence autoencoder," in *Proc. Interspeech*, 2016, pp. 765–769.

[150] Y.-C. Chen, S.-F. Huang, H.-y. Lee, Y.-H. Wang, and C.-H. Shen, "Audio Word2vec: Sequence-to-sequence autoencoding for unsupervised learning of audio segmentation and representation," *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, vol. 27, no. 9, pp. 1481–1493, Sep. 2019, doi: 10.1109/TASLP.2019.2922832.

[151] S. Scanzio, P. Laface, L. Fissore, R. Gemello, and F. Mana, "On the use of a multilingual neural network front-end," in *Proc. Interspeech*, 2008, pp. 2711–2714.

[152] Z. Tüske, J. Pinto, D. Willett, and R. Schlüter, "Investigation on cross- and multilingual MLP features under matched and mismatched acoustical conditions," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2013, pp. 7349–7353.

[153] S. Zhou, S. Xu, and B. Xu, "Multilingual end-to-end speech recognition with a single transformer on low-resource languages," 2018, *arXiv:1806.05059*.

[154] O. Adams, M. Wiesner, S. Watanabe, and D. Yarowsky, "Massively multilingual adversarial speech recognition," in *Proc. North Amer. Chapter Assoc. Comput. Linguistics: Hum. Lang. Technol.*, 2019, pp. 96–108.

[155] W. Hou, Y. Dong, B. Zhuang, L. Yang, J. Shi, and T. Shinozaki, "Large-scale end-to-end multilingual speech recognition and language identification with multi-task learning," in *Proc. Interspeech*, 2020, pp. 1037–1041.

[156] V. Pratap et al., "Massively multilingual ASR: 50 languages, 1 model, 1 billion parameters," in *Proc. Interspeech*, 2020, pp. 4751–4755.

[157] B. Li et al., "Scaling end-to-end models for large-scale multilingual ASR," in *Proc. IEEE Autom. Speech Recognit. Understanding*, 2021, pp. 1011–1018.

[158] Y. Zhang et al., "BigSSL: Exploring the frontier of large-scale semi-supervised learning for automatic speech recognition," *IEEE J. Sel. Topics Signal Process.*, vol. 16, no. 6, pp. 1519–1532, Oct. 2022.

[159] Z. Chen et al., "MAESTRO: Matched speech text representations through modality matching," in *Proc. Interspeech*, 2022, pp. 4093–4097.

[160] A. Radford et al., "Introducing whisper - robust speech recognition via large-scale weak supervision," Sep. 2022. [Online]. Available: https://openai.com/blog/whisper/

[161] T. P. Vogl, J. Mangis, A. Rigler, W. Zink, and D. Alkon, "Accelerating the convergence of the back-propagation method," *Biol. Cybern.*, vol. 59, no. 4, pp. 257–263, 1988.

[162] N. S. Keskar and G. Saon, "A nonmonotone learning rate strategy for SGD training of deep neural networks," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2015, pp. 4974–4978.

[163] S. Renals, N. Morgan, H. Bourlard, C. Wooters, and P. Kohn, "Connectionist speech recognition: Status and prospects," *Int. Conf. Sci. Inst.*, 1991, Tech. Rep. TR-OI-070.

[164] D. Johnson, D. Ellis, C. Oei, C. Wooters, and P. Faerber, "QuickNet," *Int. Conf. Swarm Intell.*, Berkeley, 2004. [Online]. Available: http://www.icsi.berkeley.edu/Speech/qn.html

[165] A. Senior, G. Heigold, M. Ranzato, and K. Yang, "An empirical study of learning rates in deep neural networks for speech recognition," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2013, pp. 6724–6728.

[166] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," in *Proc. Int. Conf. Learn. Representations*, 2019.

[167] S. L. Smith, P.-J. Kindermans, C. Ying, and Q. V. Le, "Don't decay the learning rate, increase the batch size," in *Proc. Int. Conf. Learn. Representations*, 2018.

[168] J. Howard and S. Ruder, "Universal language model fine-tuning for text classification," in *Proc. Assoc. Comput. Linguistics*, 2018, pp. 328–339.

[169] M. Jaderberg et al., "Population based training of neural networks," 2017, *arXiv:1711.09846*.

[170] T. Hospedales, A. Antoniou, P. Micaelli, and A. Storkey, "Meta-learning in neural networks: A survey," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 9, pp. 5149–5169, Sep. 2021.

[171] J. L. Elman, "Learning and development in neural networks: The importance of starting small," *Cognition*, vol. 48, no. 1, pp. 71–99, 1993, doi: 10.1016/0010-0277(93)90058-4.

[172] Y. Bengio, J. Louradour, R. Collobert, and J. Weston, "Curriculum learning," in *Proc. Int. Conf. Mach. Learn.*, 2009, pp. 41–48.

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

[174] Z. Tüske, G. Saon, K. Audhkhasi, and B. Kingsbury, "Single headed attention based sequence-to-sequence model for state-of-the-art results on switchboard," in *Proc. Interspeech*, 2020, pp. 551–555.

[175] W. Zhang, X. Chang, Y. Qian, and S. Watanabe, “Improving end-to-end single-channel multi-talker speech recognition,” *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, vol. 28, pp. 1385–1394, 2020.
[176] B. Polyak, “Some methods of speeding up the convergence of iteration methods,” *USSR Comput. Math. Math. Phys.*, vol. 4, no. 5, pp. 1–17, 1964, doi: 10.1016/0041-5553(64)90137-5.
[177] Y. Nesterov, “A method of solving a convex programming problem with convergence rate $O(\frac{1}{k^2})$,” *Sov. Math. Doklady*, vol. 27, pp. 372–376, 1983.
[178] I. Sutskever, J. Martens, G. Dahl, and G. Hinton, “On the importance of initialization and momentum in deep learning,” in *Proc. Int. Conf. Mach. Learn.*, 2013, pp. 1139–1147.
[179] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” in *Proc. Int. Conf. Learn. Representations*, 2015.
[180] Z. Tüske, G. Saon, and B. Kingsbury, “On the limit of english conversational speech recognition,” in *Proc. Interspeech*, 2021, pp. 2062–2066.
[181] P. Nakkiran, G. Kaplun, Y. Bansal, T. Yang, B. Barak, and I. Sutskever, “Deep double descent: Where bigger models and more data hurt,” in *Proc. Int. Conf. Learn. Representations*, 2020.
[182] A. Krogh and J. Hertz, “A simple weight decay can improve generalization,” in *Proc. Neural Inf. Process. Syst.*, 1991, pp. 950–957.
[183] A. F. Murray and P. J. Edwards, “Enhanced MLP performance and fault tolerance resulting from synaptic weight noise during training,” *IEEE Trans. Neural Netw.*, vol. 5, no. 5, pp. 792–802, Sep. 1994.
[184] A. Graves, “Practical variational inference for neural networks,” in *Proc. Adv. Neural Inf. Process. Syst.*, 2011.
[185] A. Neelakantan et al., “Adding gradient noise improves learning for very deep networks,” Nov. 2015, *arXiv:1511.06807*.
[186] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov, “Improving neural networks by preventing co-adaptation of feature detectors,” Jul. 2012, *arXiv:1207.0580*.
[187] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “ImageNet classification with deep convolutional neural networks,” in *Proc. Adv. Neural Inf. Process. Syst.*, 2012.
[188] Y. Gal and Z. Ghahramani, “Dropout as a bayesian approximation: Representing model uncertainty in deep learning,” in *Proc. Int. Conf. Mach. Learn.*, 2016, pp. 1050–1059.
[189] G. Huang, Y. Sun, Z. Liu, D. Sedra, and K. Q. Weinberger, “Deep networks with stochastic depth,” in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 646–661.
[190] N.-Q. Pham, T.-S. Nguyen, J. Niehues, M. Müller, and A. Waibel, “Very deep self-attention networks for end-to-end speech recognition,” in *Proc. Interspeech*, 2019, pp. 66–70.
[191] J. Lee and S. Watanabe, “Intermediate loss regularization for CTC-Based speech recognition,” in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2021, pp. 6224–6228.
[192] L. Wan, M. Zeiler, S. Zhang, Y. Le Cun, and R. Fergus, “Regularization of neural networks using DropConnect,” in *Proc. Int. Conf. Mach. Learn.*, 2013, pp. 1058–1066.
[193] D. Krueger et al., “Zoneout: Regularizing RNNs by randomly preserving hidden activations,” in *Proc. Int. Conf. Learn. Representations*, 2017.
[194] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, “Rethinking the inception architecture for computer vision,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 2818–2826.
[195] S. Bengio, O. Vinyals, N. Jaitly, and N. Shazeer, “Scheduled sampling for sequence prediction with recurrent neural networks,” in *Proc. Adv. Neural Inf. Process. Syst.*, 2015.
[196] T. Trinh, A. Dai, T. Luong, and Q. Le, “Learning longer-term dependencies in RNNs with auxiliary losses,” in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 4965–4974.
[197] R. J. Williams and J. Peng, “An efficient gradient-based algorithm for on-line training of recurrent network trajectories,” *IEEE Neural Comput.*, vol. 2, no. 4, pp. 490–501, Dec. 1990.
[198] S. Merity, N. S. Keskar, and R. Socher, “An analysis of neural language modeling at multiple scales,” 2018, *arXiv:1803.08240*.
[199] L. Meng, J. Xu, X. Tan, J. Wang, T. Qin, and B. Xu, “MixSpeech: Data augmentation for low-resource automatic speech recognition,” in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2021, pp. 7008–7012.
[200] S. Ioffe and C. Szegedy, “Batch normalization: Accelerating deep network training by reducing internal covariate shift,” in *Proc. Int. Conf. Mach. Learn.*, 2015, pp. 448–456.
[201] N. Kanda, R. Takeda, and Y. Obuchi, “Elastic spectral distortion for low resource speech recognition with deep neural networks,” in *Proc. IEEE Autom. Speech Recognit. Understanding*, 2013, pp. 309–314.
[202] T. Ko, V. Peddinti, D. Povey, and S. Khudanpur, “Audio augmentation for speech recognition,” in *Proc. Interspeech*, 2015, pp. 3586–3589.
[203] N. Jaitly and G. E. Hinton, “Vocal tract length perturbation (VTLP) improves speech recognition,” in *Proc. Int. Conf. Mach. Learn.*, 2013.
[204] G. Saon, Z. Tüske, K. Audhkhasi, and B. Kingsbury, “Sequence noise injected training for end-to-end speech recognition,” in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2019, pp. 6261–6265.
[205] D. S. Park et al., “SpecAugment on large scale datasets,” in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2020, pp. 6879–6883.
[206] C. Wang et al., “Semantic mask for transformer based end-to-end speech recognition,” in *Proc. Interspeech*, 2020, pp. 971–975.
[207] T. Hayashi et al., “Back-translation-style data augmentation for end-to-end ASR,” in *Proc. IEEE Spoken Lang. Technol. Workshop*, 2018, pp. 426–433.
[208] N. Rossenbach, M. Zeineldeen, B. Hilmes, R. Schlüter, and H. Ney, “Comparing the benefit of synthetic training data for various automatic speech recognition architectures,” in *Proc. IEEE Autom. Speech Recognit. Understanding Workshop*, 2021, pp. 788–795.
[209] T. N. Sainath et al., “No need for a lexicon? Evaluating the value of the pronunciation lexica in end-to-end models,” in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2018, pp. 5859–5863, doi: 10.1109/ICASSP.2018.8462380.
[210] C. Wooters and A. Stolcke, “Multiple-pronunciation lexical modeling in a speaker independent speech understanding system,” in *Proc. Int. Conf. Spoken Lang. Process.*, 1994, pp. 1363–1366.
[211] I. McGraw, I. Badr, and J. R. Glass, “Learning lexicons from speech using a pronunciation mixture model,” *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, vol. 21, no. 2, pp. 357–366, Feb. 2013.
[212] A. Senior, G. Heigold, M. Bacchiani, and H. Liao, “GMM-Free DNN acoustic model training,” in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2014, pp. 5602–5606, doi: 10.1109/ICASSP.2014.6854675.
[213] G. Gosztolya, T. Grósz, and L. Tóth, “GMM-Free flat start sequence-discriminative DNN training,” in *Proc. Interspeech*, 2016, pp. 3409–3413, doi: 10.21437/Interspeech.2016-391.
[214] H. Hadian, H. Sameti, D. Povey, and S. Khudanpur, “Flat-start single-stage discriminatively trained HMM-based models for ASR,” *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, vol. 26, no. 11, pp. 1949–1961, Nov. 2018.
[215] H. Soltau, B. Kingsbury, L. Mangu, D. Povey, G. Saon, and G. Zweig, “The IBM 2004 conversational telephony system for rich transcription,” in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2005, pp. 205–208.
[216] H. Hadian, D. Povey, H. Sameti, J. Trmal, and S. Khudanpur, “Improving LF-MMI using unconstrained supervisions for ASR,” in *Proc. IEEE Spoken Lang. Technol. Workshop*, 2018, pp. 43–47, doi: 10.1109/SLT.2018.8639684.
[217] N. Kanda, Y. Fujita, and K. Nagamatsu, “Lattice-free state-level minimum Bayes risk training of acoustic models,” in *Proc. Interspeech*, 2018, pp. 2923–2927, doi: 10.21437/Interspeech.2018-79.
[218] S. J. Young and P. C. Woodland, “The use of state tying in continuous speech recognition,” in *Proc. Eurospeech*, 1993, pp. 2203–2206.
[219] S. Wiesler, G. Heigold, M. Nußbaum-Thom, R. Schlüter, and H. Ney, “A discriminative splitting criterion for phonetic decision trees,” in *Proc. Interspeech*, 2010, pp. 54–57.
[220] T. Raissi, E. Beck, R. Schlüter, and H. Ney, “Towards consistent hybrid HMM acoustic modeling,” 2021, *arXiv:2104.02387*.
[221] M. Zeineldeen, A. Zeyer, W. Zhou, T. Ng, R. Schlüter, and H. Ney, “A systematic comparison of grapheme-based vs. phoneme-based label units for encoder-decoder-attention models,” Nov. 2020, *arXiv:2005.09336*.
[222] C. Lüscher et al., “RWTH ASR systems for LibriSpeech: Hybrid vs attention,” in *Proc. Interspeech*, 2019, pp. 231–235.
[223] D. Park et al., “Improved noisy student training for automatic speech recognition,” in *Proc. Interspeech*, 2020, pp. 2817–2821.
[224] D. S. Park et al., “SpecAugment: A simple data augmentation method for automatic speech recognition,” in *Proc. Interspeech*, 2019, pp. 2613–2617.
[225] W. Zhou, W. Michel, K. Irie, M. Kitza, R. Schlüter, and H. Ney, “The RWTH ASR System for TED-LIUM Release 2: Improving Hybrid HMM with SpecAugment,” in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2020, pp. 7839–7843.
[226] J. Cui et al., “Multilingual representations for low resource speech recognition and keyword search,” in *Proc. IEEE Autom. Speech Recognit. Understanding*, 2015, pp. 259–266.

[227] O. Adams, M. Wiesner, S. Watanabe, and D. Yarowsky, "Massively multilingual adversarial speech recognition," in *Proc. North Amer. Chapter Assoc. Comput. Linguistics: Hum. Lang. Technol.*, 2019, pp. 96–108.

[228] A. Kannan et al., "Large-scale multilingual speech recognition with a streaming end-to-end model," in *Proc. Interspeech*, 2019, pp. 2130–2134.

[229] A. Graves, "Connectionist Temporal Classification," in *Supervised Sequence Labelling with Recurrent Neural Networks*. Heidelberg, Germany: Springer, 2012, pp. 61–93.

[230] Y. Higuchi, S. Watanabe, N. Chen, T. Ogawa, and T. Kobayashi, "Mask CTC: Non-autoregressive end-to-end ASR with CTC and mask predict," in *Proc. Interspeech*, 2020, pp. 3655–3659.

[231] W. Chan, C. Saharia, G. Hinton, M. Norouzi, and N. Jaitly, "Imputer: Sequence modelling via imputation and dynamic programming," in *Proc. Int. Conf. Mach. Learn.*, 2020, pp. 1403–1413.

[232] Y. Fujita, S. Watanabe, M. Omachi, and X. Chang, "Insertion-based modeling for end-to-end automatic speech recognition," in *Proc. Interspeech*, 2020, pp. 3660–3664.

[233] L. Dong and B. Xu, "Cif: Continuous integrate-and-fire for end-to-end speech recognition," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2020, pp. 6079–6083.

[234] J. Nozaki and T. Komatsu, "Relaxing the conditional independence assumption of CTC-Based ASR by conditioning on intermediate predictions," in *Proc. Interspeech*, 2021, pp. 3735–3739.

[235] Y. Higuchi et al., "A comparative study on non-autoregressive modelings for speech-to-text generation," in *Proc. IEEE Autom. Speech Recognit. Understanding*, 2021, pp. 47–54.

[236] W. Zhou, R. Schlüter, and H. Ney, "Robust beam search for encoder-decoder attention based speech recognition without length bias," in *Proc. Interspeech*, 2020, pp. 1768–1772.

[237] P. Koehn and R. Knowles, "Six challenges for neural machine translation," in *Proc. 1st Workshop Neural Mach. Transl.*, 2017, pp. 28–39.

[238] Z. Tu, Z. Lu, Y. Liu, X. Liu, and H. Li, "Modeling coverage for neural machine translation," in *Proc. Assoc. Comput. Linguistics*, 2016, pp. 76–85.

[239] T. Hori, J. Cho, and S. Watanabe, "End-to-end speech recognition with word-based rnn language models," in *Proc. IEEE Spoken Lang. Technol. Workshop*, Dec. 2018, pp. 389–396.

[240] K. Deng and P. C. Woodland, "Label-synchronous neural transducer for end-to-end ASR," 2023, *arXiv:2307.03088*.

[241] T. Hori and A. Nakamura, *Speech Recognition Algorithms Using Weighted Finite-State Transducers*. San Rafael, CA, USA: Morgan & Claypool Publishers, 2013.

[242] R. Haeb-Umbach and H. Ney, "Improvements in beam search for 10000-Word continuous-speech recognition," *IEEE Speech Audio Process.*, vol. 2, no. 2, pp. 353–356, Apr. 1994.

[243] H. Ney and S. Ortmanns, "Progress in dynamic programming search for LVCSR," *Proc. IEEE*, vol. 88, no. 8, pp. 1224–1240, Aug. 2000, doi: 10.1109/5.880081.

[244] T. Hori, Y. Kubo, and A. Nakamura, "Real-time one-pass decoding with recurrent neural network language model for speech recognition," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2014, pp. 6364–6368.

[245] E. Beck, W. Zhou, R. Schlüter, and H. Ney, "LSTM language models for LVCSR in first-pass decoding and lattice-rescoring," 2019, *arXiv:1907.01030*.

[246] G. Saon, Z. Tüske, and K. Audhkhasi, "Alignment-length synchronous decoding for RNN transducer," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2020, pp. 7804–7808.

[247] A. Y. Hannun, A. L. Maas, D. Jurafsky, and A. Y. Ng, "First-pass large vocabulary continuous speech recognition using bi-directional recurrent DNNs," 2014, *arXiv:1408.2873*.

[248] N. Moritz, T. Hori, and J. Le Roux, "Triggered Attention for End-to-End Speech Recognition," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, Brighton, U.K., May 2019, pp. 5666–5670.

[249] N. Moritz, T. Hori, and J. Le, "Streaming automatic speech recognition with the transformer model," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2020, pp. 6074–6078.

[250] M. Jain et al., "RNN-T for latency controlled ASR with improved beam search," 2019, *arXiv:1911.01629*.

[251] L. Lu, C. Liu, J. Li, and Y. Gong, "Exploring transformers for large-scale speech recognition," in *Proc. Interspeech*, 2020, pp. 5041–5045.

[252] T. Wang, Y. Fujita, X. Chang, and S. Watanabe, "Streaming end-to-end ASR based on blockwise non-autoregressive models," in *Proc. Interspeech*, 2021, pp. 3755–3759.

[253] H. Miao, G. Cheng, P. Zhang, T. Li, and Y. Yan, "Online hybrid CTC/Attention architecture for end-to-end speech recognition," in *Proc. Interspeech*, 2019, pp. 2623–2627, doi: 10.21437/Interspeech.2019-2018.

[254] E. Tsunoo, Y. Kashiwagi, and S. Watanabe, "streaming transformer ASR with blockwise synchronous beam search," in *Proc. IEEE Spoken Lang. Technol. Workshop*, 2021, pp. 22–29.

[255] K. Hwang and W. Sung, "Character-level language modeling with hierarchical recurrent neural networks," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2017, pp. 5720–5724.

[256] 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.

[257] A. Kannan, Y. Wu, P. Nguyen, T. N. Sainath, Z. Chen, and R. Prabhavalkar, "An analysis of incorporating an external language model into a sequence-to-sequence model," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2018, pp. 5824–5828, doi: 10.1109/ICASSP.2018.8462682.

[258] G. Saon, Z. Tüske, D. Bolanos, and B. Kingsbury, "Advancing RNN transducer technology for speech recognition," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2021, pp. 5654–5658.

[259] H. Seki, T. Hori, S. Watanabe, N. Moritz, and J. Le Roux, "Vectorized beam search for CTC-Attention-Based speech recognition," in *Proc. Interspeech*, 2019, pp. 3825–3829.

[260] T. Hori, S. Watanabe, and J. R. Hershey, "Multi-level language modeling and decoding for open vocabulary end-to-end speech recognition," in *Proc. IEEE Autom. Speech Recognit. Understanding*, 2017, pp. 287–293.

[261] Y. Wang et al., "Espresso: A fast end-to-end neural speech recognition toolkit," in *Proc. IEEE Autom. Speech Recognit. Understanding Workshop*, 2019, pp. 136–143.

[262] Z. Tüske, K. Audhkhasi, and G. Saon, "Advancing sequence-to-sequence based speech recognition," in *Proc. Interspeech*, 2019, pp. 3780–3784.

[263] J. Drexler and J. Glass, "Subword regularization and beam search decoding for end-to-end automatic speech recognition," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2019, pp. 6266–6270.

[264] T. N. Sainath et al., "Two-pass end-to-end speech recognition," in *Proc. Interspeech*, 2019, pp. 2773–2777.

[265] Z. Yao et al., "WeNet: Production oriented streaming and non-streaming end-to-end speech recognition toolkit," in *Proc. Interspeech*, 2021, pp. 4054–4058.

[266] D. Wu et al., "U2++: Unified two-pass bidirectional end-to-end model for speech recognition," 2021, *arXiv:2106.05642*.

[267] M. Zapotoczny, P. Pietrzak, A. Lancucki, and J. Chorowski, "Lattice generation in attention-based speech recognition models," in *Proc. Interspeech*, 2019, pp. 2225–2229.

[268] J. Kim, Y. Lee, and E. Kim, "Accelerating RNN transducer inference via adaptive expansion search," *IEEE Signal Process. Lett.*, vol. 27, pp. 2019–2023, 2020.

[269] M. Abadi et al., "TensorFlow: A system for large-scale machine learning," in *Proc. 12th USENIX Symp. Operating Syst. Des. Implementation*, 2016, pp. 265–283.

[270] M. Ott et al., "FAIRSEQ: A fast, extensible toolkit for sequence modeling," in *Proc. North Amer. Chapter Assoc. Comput. Linguistics: Hum. Lang. Technol.*, 2019, pp. 48–53.

[271] J. Shen et al., "Lingvo: A modular and scalable framework for sequence-to-sequence modeling," 2019, *arXiv:1902.08295*.

[272] P. Doetsch, A. Zeyer, P. Voigtlaender, I. Kulikov, R. Schlüter, and H. Ney, "RETURNN: The RWTH extensible training framework for universal recurrent neural networks," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2017, pp. 5345–5349.

[273] A. Hannun, A. Lee, Q. Xu, and R. Collobert, "Sequence-to-sequence speech recognition with time-depth separable convolutions," in *Proc. Interspeech*, 2019, pp. 3785–3789.

[274] M. Li, M. Liu, and H. Masanori, "End-to-end speech recognition with adaptive computation steps," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2019, pp. 6246–6250.

[275] P. Bahar, N. Makarov, A. Zeyer, R. Schüter, and H. Ney, "Exploring a zero-order direct HMM based on latent attention for automatic speech recognition," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2020, pp. 7854–7858.

[276] Z. Huang, G. Zweig, and B. Dumoulin, "Cache based recurrent neural network language model inference for first pass speech recognition," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2014, pp. 6354–6358.

[277] J. Jorge, A. Giménez, J. Iranzo-Sánchez, J. Civera, A. Sanchis, and A. Juan, "Real-time one-pass decoder for speech recognition using LSTM language models," in *Proc. Interspeech*, 2019, pp. 3820–3824.
[278] W. Zhou, R. Schlüter, and H. Ney, "Full-sum decoding for hybrid HMM based speech recognition using LSTM language model," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2020, pp. 7834–7838.
[279] P. Sountsov and S. Sarawagi, "Length bias in encoder decoder models and a case for global conditioning," in *Proc. Empirical Methods Natural Lang. Process.*, 2016, pp. 1516–1525.
[280] K. Murray and D. Chiang, "Correcting length bias in neural machine translation," in *Proc. WMT*, 2018, pp. 212–223.
[281] F. Stahlberg and B. Byrne, "On NMT search errors and model errors: Cat got your tongue?," in *Proc. Empirical Methods Natural Lang. Process.*, 2019, pp. 3354–3360.
[282] N. Deshmukh, A. Ganapathiraju, and J. Picone, "Hierarchical search for large-vocabulary conversational speech recognition: Working toward a solution to the decoding problem," *IEEE Signal Process. Mag.*, vol. 16, no. 5, pp. 84–107, Sep. 1999.
[283] L. Nguyen and R. Schwartz, "Single-tree method for grammar-directed search," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 1999, pp. 613–616.
[284] L. Sarı, N. Moritz, T. Hori, and J. L. Roux, "Unsupervised speaker adaptation using attention-based speaker memory for end-to-end ASR," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2020, pp. 7384–7388.
[285] F. Weninger, J. Andrés-Ferrer, X. Li, and P. Zhan, "Listen, attend, spell and adapt: Speaker adapted sequence-to-sequence ASR," in *Proc. Interspeech*, 2019, pp. 3805–3809.
[286] Z. Meng, Y. Gaur, J. Li, and Y. Gong, "Speaker adaptation for attention-based end-to-end speech recognition," in *Proc. Interspeech*, 2019, pp. 241–245.
[287] N. Tomashenko and Y. Estève, "Evaluation of feature-space speaker adaptation for end-to-end acoustic models," in *Proc. 11th Int. Conf. Lang. Resour. Eval.*, 2018, pp. 3163–3170.
[288] S. F. Chen and J. Goodman, "An empirical study of smoothing techniques for language modeling," in *Proc. Assoc. Comput. Linguistics*, 1996, pp. 310–318.
[289] T. Mikolov, M. Karafiát, L. Burget, J. Černockỳ, and S. Khudanpur, "Recurrent neural network based language model," in *Proc. Interspeech*, 2010, pp. 1045–1048.
[290] M. Sundermeyer, R. Schlüter, and H. Ney, "LSTM neural networks for language modeling," in *Proc. Interspeech*, 2012, pp. 194–197.
[291] N.-Q. Pham, G. Kruszewski, and G. Boleda, "Convolutional neural network language models," in *Proc. Joint Conf. Empirical Methods Natural Lang. Process.*, 2016, pp. 1153–1162.
[292] Y. N. Dauphin, A. Fan, M. Auli, and D. Grangier, "Language modeling with gated convolutional networks," in *Proc. 34th Int. Conf. Mach. Learn.*, 2017, pp. 933–941.
[293] N. Zeghidour, Q. Xu, V. Liptchinsky, N. Usunier, G. Synnaeve, and R. Collobert, "Fully convolutional speech recognition," 2018, *arXiv:1812.06864*.
[294] T. Likhomanenko, G. Synnaeve, and R. Collobert, "Who needs words? lexicon-free speech recognition," in *Proc. Interspeech*, 2019, pp. 3915–3919.
[295] R. Al-Rfou, D. Choe, N. Constant, M. Guo, and L. Jones, "Character-level language modeling with deeper self-attention," in *Proc. AIII*, 2019, pp. 3159–3166.
[296] K. Irie, A. Zeyer, R. Schlüter, and H. Ney, "Language modeling with deep transformers," in *Proc. Interspeech*, 2019, pp. 3905–3909.
[297] Z. Dai, Z. Yang, Y. Yang, J. G. Carbonell, Q. Le, and R. Salakhutdinov, "Transformer-XL: Attentive language models beyond a fixed-length context," in *Proc. Assoc. Comput. Linguistics*, 2019, pp. 2978–2988.
[298] P. J. Werbos, "Backpropagation through time: What it does and how to do it," *Proc. IEEE*, vol. 78, no. 10, pp. 1550–1560, Oct. 1990, doi: 10.1109/5.58337.
[299] J. W. Rae, A. Potapenko, S. M. Jayakumar, and T. P. Lillicrap, "Compressive transformers for long-range sequence modelling," in *Proc. Adv. Neural Inf. Process. Syst.*, 2020, pp. 6154–6158.
[300] C. Gulcehre et al., "On using monolingual corpora in neural machine translation," 2015, *arXiv:1503.03535*.
[301] A. Sriram, H. Jun, S. Satheesh, and A. Coates, "Cold fusion: Training Seq2Seq models together with language models," in *Proc. Interspeech*, 2018, pp. 387–391.
[302] C. Shan et al., "Component fusion: Learning replaceable language model component for end-to-end speech recognition system," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2019, pp. 5361–5635.
[303] E. McDermott, H. Sak, and E. Variani, "A density ratio approach to language model fusion in end-to-end automatic speech recognition," in *Proc. IEEE Autom. Speech Recognit. Understanding Workshop*, 2019, pp. 434–441.
[304] Z. Meng et al., "Internal language model estimation for domain-adaptive end-to-end speech recognition," in *Proc. IEEE Spoken Lang. Technol. Workshop*, 2020, pp. 243–250.
[305] W. Zhou, Z. Zheng, R. Schlüter, and H. Ney, "On language model integration for RNN transducer based speech recognition," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2022, pp. 8407–8411.
[306] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proc. Assoc. Comput. Linguistics*, 2019, pp. 4171–4186.
[307] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever, "Language models are unsupervised multitask learners," 2019, openAI blog. [Online]. Available: https://cdn.openai.com/better-language-models/language_models_are_unsupervised_multitask_learners.pdf
[308] J. Salazar, D. Liang, T. Q. Nguyen, and K. Kirchhoff, "Masked language model scoring," in *Proc. Assoc. Comput. Linguistics*, 2020, pp. 2699–2712.
[309] S. Kim, S. Dalmia, and F. Metze, "Gated embeddings in end-to-end speech recognition for conversational-context fusion," in *Proc. Assoc. Comput. Linguistics*, 2019, pp. 1131–1141.
[310] A. Zeyer, A. Merboldt, W. Michel, R. Schlüter, and H. Ney, "Librispeech transducer model with internal language model prior correction," in *Proc. Interspeech*, 2021, pp. 2052–2056.
[311] L. R. Bahl, F. Jelinek, and R. L. Mercer, "A maximum likelihood approach to continuous speech recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. PAMI-5, no. 2, pp. 179–190, Mar. 1983.
[312] J. Makhoul and R. Schwartz, "State of the art in continuous speech recognition," *Proc. Nat. Acad. Sci.*, vol. 92, no. 22, pp. 9956–9963, Oct. 1995.
[313] D. Klakow and J. Peters, "Testing the correlation of word error rate and perplexity," *Speech Commun.*, vol. 38, no. 1, pp. 19–28, 2002.
[314] M. Sundermeyer, H. Ney, and R. Schlüter, "From feedforward to recurrent LSTM neural networks for language modeling," *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, vol. 23, no. 3, pp. 517–529, Mar. 2015.
[315] T. Hori, C. Hori, S. Watanabe, and J. R. Hershey, "Minimum word error training of long short-term memory recurrent neural network language models for speech recognition," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2016, pp. 5990–5994.
[316] J. Godfrey, E. Holliman, and J. McDaniel, "SWITCHBOARD: Telephone speech corpus for research and development," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 1992, pp. 517–520.
[317] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, "Librispeech: An ASR corpus based on public domain audio books," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2015, pp. 5206–5210.
[318] A. Zeyer, P. Bahar, K. Irie, R. Schlüter, and H. Ney, "A comparison of transformer and LSTM encoder decoder models for ASR," in *Proc. IEEE Autom. Speech Recognit. Understanding*, 2019, pp. 8–15.
[319] W.-N. Hsu et al., "HuBERT: Self-supervised speech representation learning by masked prediction of hidden units," *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, vol. 19, pp. 3451–3460, 2021.
[320] G. Synnaeve et al., "End-to-end ASR: From supervised to semi-supervised learning with modern architectures," in *Proc. Int. Conf. Mach. Learn.*, 2020.
[321] E. G. Ng, C.-C. Chiu, Y. Zhang, and W. Chan, "Pushing the limits of non-autoregressive speech recognition," in *Proc. Interspeech*, 2021, pp. 3725–2729.
[322] J. Kahn et al., "Libri-light: A benchmark for ASR with limited or no supervision," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2020, pp. 7669–7673.
[323] Y. Wang et al., "Transformer-based acoustic modeling for hybrid speech recognition," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2020, pp. 6874–6878.
[324] K. Kim et al., "E-branchformer: Branchformer with enhanced merging for speech recognition," in *Proc. IEEE Spoken Lang. Technol. Workshop*, 2023, pp. 84–91.
[325] M. Kitza, P. Golik, R. Schlüter, and H. Ney, "Cumulative adaptation for BLSTM acoustic models," in *Proc. Interspeech*, 2019, pp. 754–758.
[326] C.-C. Chiu et al., "State-of-the-art speech recognition with sequence-to-sequence models," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2018, pp. 4774–4778.

[327] K. Kim et al., "Attention based on-device streaming speech recognition with large speech corpus," in *Proc. IEEE Autom. Speech Recognit. Understanding Workshop*, 2019, pp. 956–963.
[328] J. Li et al., "Developing RNN-T models surpassing high-performance hybrid models with customization capability," in *Proc. Interspeech,*2020, pp. 3590–3594.
[329] R. Hsiao, D. Can, T. Ng, R. Travadi, and A. Ghoshal, "Online automatic speech recognition with listen, attend and spell model," *IEEE Signal Process. Lett.*, vol. 27, pp. 1889–1893, 2020.
[330] Y. Shi et al., "Emformer: Efficient memory transformer based acoustic model for low latency streaming speech recognition," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2021, pp. 6783–6787.
[331] X. Chen, Y. Wu, Z. Wang, S. Liu, and J. Li, "Developing real-time streaming transformer transducer for speech recognition on large-scale dataset," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2021, pp. 5904–5908.
[332] T. N. Sainath et al., "A streaming on-device end-to-end model surpassing server-side conventional model quality and latency," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2020, pp. 6059–6063.
[333] B. Li et al., "A better and faster end-to-end model for streaming ASR," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2021, pp. 5634–5638.
[334] T. N. Sainath et al., "An efficient streaming non-recurrent on-device end-to-end model with improvements to rare-word modeling," in *Proc. Interspeech*, 2021, pp. 1777–1781.
[335] A. Bapna et al., "SLAM: A unified encoder for speech and language modeling via speech-text joint pre-training," 2021, *arXiv:2110.10329*.
[336] A. Bapna et al., "mSLAM: Massively Multilingual Joint Pre-Training for Speech and text," 2022, *arXiv:2202.01374*.
[337] Y. Tang et al., "Unified speech-text pre-training for speech translation and recognition," in *Proc. Assoc. Comput. Linguistics*, 2022, pp. 1488–1499.
[338] Y.-A. Chung, C. Zhu, and M. Zeng, "SPLAT: Speech-language joint pre-training for spoken language understanding," in *Proc. North Amer. Chapter Assoc. Comput. Linguistics: Hum. Lang. Technol.*, 2021, pp. 1897–1907.
[339] J. Ao et al., "SpeechT5: Unified-modal encoder-decoder pre-training for spoken language processing," in *Proc. Assoc. Comput. Linguistics*, 2022, pp. 5723–5738.
[340] S. Thomas, H. K. J. Kuo, B. Kingsbury, and G. Saon, "Towards reducing the need for speech training data to build spoken language understanding systems," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2022, pp. 7932–7936.
[341] T. N. Sainath et al., "JOIST: A joint speech and text streaming model for ASR," in *Proc. IEEE Spoken Lang. Technol. Workshop*, 2023, pp. 52–59.
[342] T. Hori, R. Astudillo, T. Hayashi, Y. Zhang, S. Watanabe, and J. Le Roux, "Cycle-consistency training for end-to-end speech recognition," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2019, pp. 6271–6275.
[343] T. Ochiai, S. Watanabe, T. Hori, and J. R. Hershey, "Multichannel end-to-end speech recognition," in *Proc. Int. Conf. Mach. Learn.*, 2017, pp. 2632–2641.
[344] J. Li, "Recent advances in end-to-end automatic speech recognition," *APSIPA Trans. Signal Inf. Process*, vol. 11, no. 1, Nov. 2021, doi: 10.1561/116.00000050.

Rohit Prabhavalkar (Member, IEEE) received the Ph.D. degree in computer science and engineering from The Ohio State University, Columbus, OH, USA, in 2013. Following his Ph.D., he joined the Speech Technologies group, Google, where he is currently a Staff Research Scientist. At Google, his research has focused primarily on developing compact acoustic models which can run efficiently on mobile devices, and on developing improved end-to-end automatic speech recognition systems. He has coauthored more than 50 refereed papers, which have received two best paper awards (ASRU 2017; ICASSP 2018). He is currently a Member of the IEEE Speech and Language Processing Technical Committee during 2018–2024, and an Associate Editor for the IEEE/ACM TRANSACTIONS ON AUDIO, SPEECH, AND LANGUAGE PROCESSING.

Takaaki Hori (Senior Member, IEEE) received the Ph.D. degree in system and information engineering from Yamagata University, Yonezawa, Japan, in 1999. From 1999 to 2015, he was engaged in researches on speech recognition and spoken language processing with Cyber Space Laboratories and Communication Science Laboratories, Nippon Telegraph and Telephone Corporation, Tokyo, Japan. From 2015 to 2021, he was a Senior Principal Research Scientist with Mitsubishi Electric Research Laboratories, Cambridge, MA, USA. He is currently a Machine Learning Researcher with Apple. His research interests include automatic speech recognition, spoken language understanding, and language modeling. During 2020–2022, he was a Member of the IEEE Speech and Language Processing Technical Committee.

Tara N. Sainath (Fellow, IEEE) received the Ph.D. degree in electrical engineering and computer science from the Massachusetts Institute of Technology, Cambridge, MA, USA, in 2009. The main focus of her Ph.D. work was in acoustic modeling for noise robust speech recognition. After her Ph.D., she spent five years with Speech and Language Algorithms Group, IBM Thomas J. Watson Research Center, before joining Google Research. She was the Program Chair for ICLR in 2017 and 2018. She has co-organized numerous special sessions and workshops, including Interspeech 2010, ICML 2013, Interspeech 2016 and ICML 2017. In addition, she is a Member of the IEEE Speech and Language Processing Technical Committee and an Associate Editor for IEEE/ACM TRANSACTIONS ON AUDIO, SPEECH, AND LANGUAGE PROCESSING.

Ralf Schlüter (Senior Member, IEEE) received the Dr.rer.nat. degree in computer science and the Habilitation degree in computer science from RWTH Aachen University, Aachen, Germany, in 2000 and 2009, respectively. In May 1996, he joined the Computer Science Department, RWTH Aachen University, where he is currently a Lecturer and Academic Director, leading the Automatic Speech Recognition Group, the Chair Computer Science 6 – Machine Learning and Human Language Technology. In 2019, he also joined AppTek GmbH Aachen as Senior Researcher. His research interests include sequence classification, specifically all aspects of automatic speech recognition, decision theory, stochastic modeling, and signal analysis. During 2013–2019, he was the Subject Editor of *Speech Communication*.

Shinji Watanabe (Fellow, IEEE) received the B.S., M.S., and Ph.D. (Dr. Eng.) degrees from Waseda University, Tokyo, Japan. He is currently an Associate Professor with Carnegie Mellon University, Pittsburgh, PA, USA. He was a research Scientist with NTT Communication Science Laboratories, Kyoto, Japan, from 2001 to 2011, a Visiting Scholar with the Georgia Institute of Technology, Atlanta, GA, USA, in 2009, and a Senior Principal Research Scientist with Mitsubishi Electric Research Laboratories, Cambridge, MA, USA, from 2012 to 2017. Before Carnegie Mellon University, he was an Associate Research Professor with Johns Hopkins University, Baltimore, MD, USA, from 2017 to 2020. His research interests include automatic speech recognition, speech enhancement, spoken language understanding, and machine learning for speech and language processing. He is an ISCA Fellow.