Towards Maximum Likelihood Training for Transducer-Based Streaming Speech Recognition

Hyeonseung Lee, Ji Won Yoon, Sungsoo Kim, and Nam Soo Kim, *Senior Member, IEEE*

***Abstract*—Transducer neural networks have emerged as the mainstream approach for streaming automatic speech recognition (ASR), offering state-of-the-art performance in balancing accuracy and latency. In the conventional framework, streaming transducer models are trained to maximize the likelihood function based on non-streaming recursion rules. However, this approach leads to a mismatch between training and inference, resulting in the issue of deformed likelihood and consequently suboptimal ASR accuracy. We introduce a mathematical quantification of the gap between the actual likelihood and the deformed likelihood, namely forward variable causal compensation (FoCC). We also present its estimator, FoCCE, as a solution to estimate the exact likelihood. Through experiments on the LibriSpeech dataset, we show that FoCCE training improves the accuracy of the streaming transducers.**

***Index Terms*—RNN-transducer (RNN-T), transducer neural network, automatic speech recognition (ASR), streaming ASR.**

I. Introduction

MODERN automatic speech recognition (ASR) has witnessed substantial accuracy improvements, primarily attributed to advances in deep learning. While the pursuit of higher accuracy in general ASR remains a priority, recent studies have increasingly emphasized the need to maintain accuracy in challenging scenarios, including spoken named entities [1], [2], multi-lingual speech [3], [4], and streaming ASR [5], [6], [7]. Notably, the rising demand for on-device and real-time ASR underscores the importance of streaming ASR.

The accuracy of streaming ASR is degraded compared to its non-streaming counterpart, especially when the model is restricted to have low latency. Two separate causes induce the accuracy degradation of the streaming model. The primary cause is *information deficiency*; the streaming model predicts an output based on a limited input context, whilst the non-streaming model has the advantage of being aware of the entire context. Information deficiency is an inevitable intrinsic property of streaming ASR. The second cause is *deformed likelihood*. Deep learning ASR models are usually trained based on the maximum likelihood criterion [8], [9], [10], [11], [12], [13], [14], [15], [16]. Mainstream ASR approaches [9], [10], [12] commonly define likelihood by breaking it into local probability terms that are estimated using neural networks with a softmax output layer. These local probabilities are designed to sum up to the likelihood when all local probabilities are modeled with the entire input context, which is the case for non-streaming ASR but not valid for streaming ASR [16]. Consequently, naively utilizing mainstream ASR likelihood functions for training the streaming models leads to the deformed likelihood problem.

Several approaches [14], [16] adopt a globally normalized likelihood definition, which bypasses the issue of deformed likelihood in streaming ASR. These methods define globally normalized likelihood as a ratio between the score-sum of accepting paths (corresponding to ground truth data) and the score-sum of all possible paths. By excluding local probability terms, these approaches avoid the deformed likelihood problem. However, they exhibit lower accuracy compared to local probability-based mainstream ASR methods in both streaming and non-streaming scenarios. This limitation restricts the application of globally normalized likelihood in ASR.

Transducer neural networks [10], often called RNN-T, have recently emerged as the dominant approach for streaming ASR, offering a state-of-the-art tradeoff between accuracy and latency. The prevalent paradigm in streaming transducer models relies on a likelihood function derived from the non-streaming recursion rule. Training a transducer with this naive likelihood induces the deformed likelihood problem, resulting in sub-optimal ASR accuracy.

This letter introduces a mathematical perspective on the deformed likelihood problem in streaming transducer training and proposes a novel solution to mitigate it. The key contributions of this letter are as follows:

- We reframe the dynamic programming for the non-streaming transducer likelihood [10] using detailed probabilistic notation and demonstrate its mismatch to a streaming model due to the *deformed likelihood* problem.
- We quantify the gap between the deformed and the actual likelihood in streaming transducer training, namely "**Fo**rward Variable **C**ausal **C**ompensation" (FoCC).
- We propose the FoCC estimator (FoCCE) network, which estimates the actual likelihood instead of the deformed likelihood in streaming transducer training.

Received 1 July 2024; accepted 29 October 2024. Date of publication 4 November 2024; date of current version 16 December 2024. This work was supported in part by COMPA funded by the Korea Government (MSIT and Police) under Grant RS2023-00235082, and in part by the Institute of Information & Communications Technology Planning & Evaluation (IITP) funded by the Korea Government (MSIT) under Grant 2021-0-01341 through Artificial Intelligence Graduate School Program (Chung-Ang University). The associate editor coordinating the review of this article and approving it for publication was Dr. Yu Tsao. *(Corresponding author: Nam Soo Kim.)*

Hyeonseung Lee is with the XL8 Inc., Seoul 08846, Republic of Korea (e-mail: swigls1@gmail.com).

Ji Won Yoon is with the Department of Artificial Intelligence, Chung-Ang University, Seoul 06974, Republic of Korea (e-mail: jiwonyoon@cau.ac.kr).

Sungsoo Kim and Nam Soo Kim are with the Institute of New Media and Communications, Department of Electrical and Computer Engineering, Seoul National University, Seoul 08826, Republic of Korea (e-mail: sskim@hi.snu.ac.kr; nkim@snu.ac.kr).

Digital Object Identifier 10.1109/LSP.2024.3491019

- We experimentally show that FoCCE training improves the streaming transducers' ASR accuracy on the LibriSpeech dataset, reducing the accuracy gap between streaming and non-streaming transducers.

II. Transducer Neural Networks

A. Non-Streaming Transducer

Given an input sequence $\mathbf{x}_{1:T}$ of length T and the corresponding target sequence $\mathbf{y}_{0:U}$ of length U ($y_u \in \mathbb{N}$, y_0 is the start-of-sequence token $\langle sos \rangle$), a transducer neural network [10] parametrized by θ computes the conditional likelihood $L_\theta(\mathbf{x}_{1:T}, \mathbf{y}_{0:U})$ as

$$L_\theta(\mathbf{x}_{1:T}, \mathbf{y}_{0:U}) := \log P_\theta(\mathbf{y}_{0:U}, z_U \le T < z_{U+1} | \mathbf{x}_{1:T}), \quad (1)$$

$$\mathbf{f}_{1:T} = \text{Encoder}_\theta(\mathbf{x}_{1:T}) \quad (2)$$

where the latent alignment variable $z_u \in \mathbb{N}$ is defined such that $z_u = t$ means that the target y_u is aligned to the encoded input f_t. $\text{Encoder}_\theta(\cdot)$ is a neural network that extracts abstract information from the entire input sequence. The encoder may subsample its input, i.e., $\mathbf{f}_{1:T} = \text{Encoder}_\theta(\mathbf{x}_{1:T'})$ where $T < T'$, to reduce the length mismatch between input and target. As any long input sequence can be reshaped into a chunked sequence that has the same length as the encoded sequence, i.e., $x_{1:T'} = x'_{1:T}$, we denote the input as $x_{1:T}$ for simplicity.

Since the alignments between the inputs and targets are assumed to be monotonic, $\mathbf{z}_{0:U}$ is constrained such that

$$z_0 \le 1 \le z_1 \le z_2 \le z_3 \le \cdots \le z_U. \quad (3)$$

The end condition $z_U \le T < z_{U+1}$ in (1) indicates that the entire target $\mathbf{y}_{1:U}$ is aligned to the input $\mathbf{x}_{1:T}$ whilst the next target y_{U+1} is not (i.e., only $\mathbf{y}_{0:U}$ is included in the inputs).

Dynamic programming is used to obtain $L_\theta(\mathbf{x}_{1:T}, \mathbf{y}_{0:U})$ in $\mathcal{O}(TU)$ computations. The forward variable is defined as

$$\alpha_\theta(t,u) := P_\theta(\mathbf{y}_{0:u}, z_u \le t \le z_{u+1} | \mathbf{x}_{1:T}). \quad (4)$$

From (3) and (4), at the initial point $\alpha_\theta(1,0) = P_\theta(y_0 = \langle sos \rangle | \mathbf{x}_{1:T}) = 1$ and at the boundaries $\alpha_\theta(t,-1) = \alpha_\theta(0,u) = 0$ for $t \in [1,T]$ and $u \in [0,U]$. Local probabilities in x-axis and y-axis are respectively defined as

$$\begin{aligned} \phi_\theta(t,u) &:= P_\theta(\ z_{u+1} \ge t+1 | \quad \mathbf{x}_{1:T}, \mathbf{y}_{0:u}, z_u \le t \le z_{u+1}), \\ Y_\theta(t,u) &:= P_\theta(y_{u+1}, z_{u+1} = t | \quad \mathbf{x}_{1:T}, \mathbf{y}_{0:u}, z_u \le t \le z_{u+1}), \end{aligned} \quad (5)$$

for $t \in [1,T], u \in [0,U]$, which is to be estimated by neural networks. With blank probability $\phi_\theta(\cdot,\cdot)$ and label probability $Y_\theta(\cdot,\cdot)$, the forward variable can be recursively computed as

$$\begin{aligned} \alpha_\theta(t,u) = {} & P_\theta(\mathbf{y}_{0:u}, z_u < t \le z_{u+1} | \mathbf{x}_{1:T}) \\ & + P_\theta(\mathbf{y}_{0:u}, z_u = t \le z_{u+1} | \mathbf{x}_{1:T}) \\ = {} & P_\theta(\mathbf{y}_{0:u}, z_u \le t-1 \le z_{u+1} | \mathbf{x}_{1:T}) \\ & \cdot P_\theta(z_{u+1} \ge t | \mathbf{x}_{1:T}, \mathbf{y}_{0:u}, z_u \le t-1 \le z_{u+1}) \\ & + P_\theta(\mathbf{y}_{0:u-1}, z_{u-1} \le t \le z_u | \mathbf{x}_{1:T}) \\ & \cdot P_\theta(y_u, z_u = t | \mathbf{x}_{1:T}, \mathbf{y}_{0:u-1}, z_{u-1} \le t \le z_u) \\ = {} & \alpha_\theta(t-1,u)\phi_\theta(t-1,u) \\ & + \alpha_\theta(t,u-1)Y_\theta(t,u-1), \end{aligned} \quad (6)$$

according to (3). The conditional likelihood in (1) can be obtained by

$$L_\theta(\mathbf{x}_{1:T}, \mathbf{y}_{0:U}) = \log \alpha_\theta(T,U)\phi_\theta(T,U), \quad (7)$$

which is used as a training objective.

In this subsection, we introduce the alignment variable z_u separately from the target label variable y_u. This separation clarifies the Bayes' rule within the transducer recursion, as shown in (6). We found that the forward-backward algorithm [10], which is a training method for transducer networks in most prior studies, does not obey Bayes' rule. Therefore this paper focuses on the training based on the forward variable recursion rather than the forward-backward algorithm.

B. Streaming Transducer

In the streaming case, as depicted on the left of Fig. 1, a transducer network θ estimates the local probabilities at the t-th timestep using only limited input context $\mathbf{x}_{1:e(t)}$:

$$\mathbf{f}_{1:t} = \text{CausalEncoder}_\theta(\mathbf{x}_{1:e(t)}), \quad (8)$$

$$\begin{aligned} \tilde{\phi}_\theta(t,u) &:= P_\theta(\ z_{u+1} \ge t+1 | \mathbf{x}_{1:e(t)}, \mathbf{y}_{0:u}, z_u \le t \le z_{u+1}), \\ \tilde{Y}_\theta(t,u) &:= P_\theta(y_{u+1}, z_{u+1} = t | \mathbf{x}_{1:e(t)}, \mathbf{y}_{0:u}, z_u \le t \le z_{u+1}), \end{aligned} \quad (9)$$

where the $\text{CausalEncoder}_\theta(\cdot)$ is similar to the $\text{Encoder}_\theta(\cdot)$ in (2), except that its input context is limited by a context-end function $e : \mathbb{N} \to \mathbb{N}$. In general, $e(t) = \min(T, C\lceil t/C \rceil + R)$ with chunk size C and right context offset R.

Conventional methods train streaming transducer models by maximizing the likelihood given by (7), with local probabilities in (6) being substituted by (9). This naive approach breaks Bayes' rule, thereby training models with deformed likelihood. To obtain the actual likelihood, we introduce a probability ratio, namely **Fo**rward Variable **C**ausal **C**ompensation (FoCC):

$$\begin{aligned} \gamma_\theta(t,u) &:= \frac{P_\theta(\mathbf{y}_{0:u}, z_u \le t < z_{u+1} | \mathbf{x}_{1:e(t+1)})}{P_\theta(\mathbf{y}_{0:u}, z_u \le t < z_{u+1} | \mathbf{x}_{1:e(t)})} \\ &= \frac{P_\theta(\mathbf{x}_{e(t)+1:e(t+1)} | \mathbf{x}_{1:e(t)}, \mathbf{y}_{0:u}, z_u \le t < z_{u+1})}{P_\theta(\mathbf{x}_{e(t)+1:e(t+1)} | \mathbf{x}_{1:e(t)})}. \end{aligned} \quad (10)$$

Note that if t and $t+1$ are in the same encoder chunk, i.e., $e(t) = e(t+1)$, then $\gamma_\theta(t,\cdot) = 1$ as shown in (10). Thus, FoCC needs to be calculated only for $t : e(t) < e(t+1)$.

The modified recursion rule can be formulated with the streaming forward variable, which is defined as

$$\begin{aligned} \tilde{\alpha}_\theta(t,u) &:= P_\theta(\mathbf{y}_{0:u}, z_u \le t \le z_{u+1} | \mathbf{x}_{1:e(t)}) \\ &= P_\theta(\mathbf{y}_{0:u}, z_u < t \le z_{u+1} | \mathbf{x}_{1:e(t-1)})\gamma_\theta(t-1,u) \\ &\quad + P_\theta(\mathbf{y}_{0:u}, z_u = t \le z_{u+1} | \mathbf{x}_{1:e(t)}) \\ &= \tilde{\alpha}_\theta(t-1,u)\tilde{\phi}_\theta(t-1,u)\gamma_\theta(t-1,u) \\ &\quad + \tilde{\alpha}_\theta(t,u-1)\tilde{Y}_\theta(t,u-1). \end{aligned} \quad (11)$$

According to (1), (9), and (11), the actual likelihood is

$$L_\theta(\mathbf{x}_{1:T}, \mathbf{y}_{0:U}) = \log \tilde{\alpha}_\theta(T,U)\tilde{\phi}_\theta(T,U). \quad (12)$$

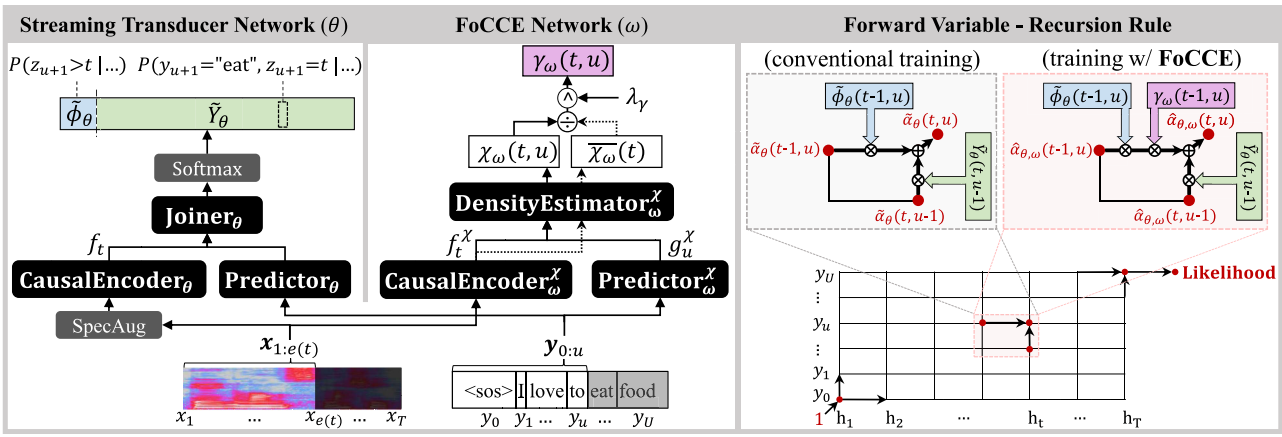

Fig. 1. An illustration of the proposed FoCCE training. The streaming transducer network (left) and the FoCCE network (middle) respectively estimate the local probabilities and FoCC values, which are used to estimate the actual likelihood by the modified forward variable recursion rule (red boxes on the right).

III. Forward Variable Causal Compensation Estimator (FoCCE)

FoCC values $\gamma_\theta(\cdot,\cdot)$ defined in (10) are necessary for determining the actual likelihood in streaming transducers, though we could not find any analytic solution to access the probability terms in (10) based on the transducer network outputs $\tilde{Y}_\theta(\cdot,\cdot)$ and $\tilde{\phi}_\theta(\cdot,\cdot)$. For this reason, we propose to approximate $\gamma_\theta(\cdot,\cdot)$ with $\gamma_\omega(\cdot,\cdot)$ using a separate FoCC estimator (FoCCE) network parametrized by ω.

Learning the probability ratio $\gamma_\omega(\cdot,\cdot)$ with a neural network is challenging, so we split it into two probability densities $\chi_\omega(\cdot,\cdot)$, $\bar{\chi}_\omega(\cdot)$ and let the FoCCE network estimate them:

$$\gamma_\omega(t,u) := \left(\frac{\chi_\omega(t,u)}{\bar{\chi}_\omega(t)}\right)^{\lambda_\gamma} \tag{13}$$

where

$$\begin{aligned}\chi_\omega(t,u) &:= P_\omega(\mathbf{x}_{e(t)+1:e(t+1)}|\mathbf{x}_{1:e(t)}, \mathbf{y}_{0:u}, z_u \le t < z_{u+1}),\\ \bar{\chi}_\omega(t) &:= P_\omega(\mathbf{x}_{e(t)+1:e(t+1)}|\mathbf{x}_{1:e(t)})\end{aligned} \tag{14}$$

for $t \in [1,T)$, $u \in [0,U]$, and λ_γ is a non-negative scaling factor for FoCCE. Intuitively, $\bar{\chi}_\omega(\cdot)$ is a probability density of next-chunk input features given the current history of the input sequence (similar to autoregressive predictive coding [17]), while the density $\chi_\omega(\cdot,\cdot)$ is also conditioned on the target sequence history. With enough model capacity, it is assumed that both $\gamma_\theta(\cdot,\cdot)$ and $\gamma_\omega(\cdot,\cdot)$ converge to the true probability ratio $\gamma(\cdot,\cdot)$ as training progresses, therefore $\gamma_\theta(\cdot,\cdot) \approx \gamma_\omega(\cdot,\cdot)$ for well-trained models θ and ω.

The probability densities $\chi_\omega(\cdot,\cdot)$, $\bar{\chi}_\omega(\cdot)$ in (14) are estimated by the FoCCE network ω, as illustrated in the middle of Fig. 1. The architecture of this network is adapted from the conventional transducer networks, as shown below:

$$\begin{aligned}\chi_\omega(t,u) &= \text{DensityEstimator}^\chi_\omega([f^\chi_t; g^\chi_u]),\\ \bar{\chi}_\omega(t) &= \text{DensityEstimator}^\chi_\omega([f^\chi_t; \vec{0}]),\end{aligned} \tag{15}$$

$$\begin{aligned}f^\chi_t &= \text{CausalEncoder}^\chi_\omega(x_{1:e(t)}),\\ g^\chi_u &= \text{Predictor}^\chi_\omega(y_{0:u}),\end{aligned} \tag{16}$$

where $[\cdot;\cdot]$ denotes the concatenation, and $\vec{0}$ stands for a zero vector. The $\text{CausalEncoder}^\chi_\omega(\cdot)$ and $\text{Predictor}^\chi_\omega(\cdot)$ respectively mean an encoder and a prediction network that function similarly to those of a transducer network. The $\text{CausalEncoder}^\chi_\omega(\cdot)$ operates causally, just like the $\text{CausalEncoder}_\theta(\cdot)$ described in (8), using the same context-end function $e(\cdot)$. In contrast to the joiner of a transducer network, the $\text{DensityEstimator}^\chi_\omega(\cdot)$ models probability densities in continuous space. To model arbitrary densities, we utilized normalizing flows [18] to implement $\text{DensityEstimator}^\chi_\omega(\cdot)$.

The FoCCE network ω is trained independently from the transducer model, maximizing the objective

$$L^\chi_\omega(\mathbf{x}_{1:T}, \mathbf{y}_{0:U}) := \sum_{t=1}^{T-1}\left(\bar{\chi}_\omega(t) + \frac{1}{U}\sum_{u=1}^{U}\chi_\omega(t,u)\right). \tag{17}$$

Based on the FoCC estimation $\gamma_\omega(\cdot,\cdot)$, a streaming transducer network θ is trained to maximize the modified likelihood:

$$L^{mod}_{\theta,\omega}(\mathbf{x}_{1:T}, \mathbf{y}_{0:U}) := \log \hat{\alpha}_{\theta,\omega}(T,U)\tilde{\phi}_\theta(T,U), \tag{18}$$

$$\begin{aligned}\hat{\alpha}_{\theta,\omega}(t,u) :=& \hat{\alpha}_{\theta,\omega}(t-1,u)\tilde{\phi}_\theta(t-1,u)\text{sg}\big(\gamma_\omega(t-1,u)\big)\\ &+ \hat{\alpha}_{\theta,\omega}(t,u-1)\tilde{Y}_\theta(t,u-1),\end{aligned} \tag{19}$$

with the initial and boundary values the same as $\alpha_\theta(t,u)$. The modified likelihood in (18) approximates the actual likelihood in (12), mitigating the deformed likelihood problem. The right side of Fig. 1 depicts the modified forward variable recursion rule described in (19). In this rule, the stop-gradient operator $\text{sg}(\cdot)$ is applied to $\gamma_\omega(\cdot,\cdot)$ to prevent the divergence of parameter values. Note that from (13), the modified streaming recursion in (19) can be smoothly transitioned into the conventional recursion in (6) by setting λ_γ close to 0.

The whole training objective is given by

$$\begin{aligned}L^{tot}_{\theta,\omega}(\mathbf{x}_{1:T}, \mathbf{y}_{0:U}) :=\ & \lambda_{mod} L^{mod}_{\theta,\omega}(\mathbf{x}_{1:T}, \mathbf{y}_{0:U})\\ &+ \lambda_\chi L^\chi_\omega(\mathbf{x}_{1:T}, \mathbf{y}_{0:U}).\end{aligned} \tag{20}$$

Both the transducer network θ and the FoCCE network ω are learned to maximize $\mathbb{E}_{\mathbf{x},\mathbf{y}\sim\mathbb{D}} L^{tot}_{\theta,\omega}(\mathbf{x},\mathbf{y})$ given a training set $\mathbb{D}$. Note that the gradient backpropagated from the FoCCE network does not directly affect the transducer network.

TABLE I
WORD ERROR RATES (WERS) COMPARISON OF TRANSDUCER MODELS ON THE LIBRISPEECH AND TED-LIUM3 DATASETS

Transducer model	Attention chunk size	FoCCE network hyperparam.			# param. (train)	LibriSpeech WER [%]			
		λ_γ	CausalEncoder$^{\chi}_{\omega}(\cdot)$ conv. module			dev		test	
			# module stacks	module dim.		clean	other	clean	other
Zipformer (non-streaming)	full	-			25.6M	2.42	5.96	2.54	6.00
Zipformer (streaming)		-			25.6M	3.27	9.41	3.53	9.17
+ FoCCE (proposed)		0.01	8	320	29.1M	3.20	9.31	3.47	9.06
	8	0.05	”	”	”	**3.13**	8.95	**3.27**	**8.76**
	(160 ms)	0.25	”	”	”	3.32	9.40	3.60	9.20
		0.05	4	256	27.3M	3.26	9.25	3.41	9.00
		”	8	512	33.2M	3.14	**8.90**	3.34	8.78

Transducer model	Attention chunk size	λ_r	CausalEncoder$^{\chi}_{\omega}(\cdot)$ conv. module		# param. (train)	TED-LIUM3 WER [%]	
			# module stacks	module dim.		dev	test
Zipformer (non-streaming)	full	-			25.6M	6.46	5.91
Zipformer (streaming)		-			25.6M	9.43	8.57
+ FoCCE (proposed)	8	0.01	8	320	29.1M	9.28	8.41
	(160 ms)	0.05	”	”	”	**9.06**	**8.10**
		0.25	”	”	”	9.35	8.51

IV. EXPERIMENTAL RESULTS

A. Experimental Setting

We followed the icefall [19] framework to train and evaluate the transducer models.

1) Data Preparation: We conducted experiments on the LibriSpeech [20] and the TED-LIUM3 [21] datasets, with the designated training, validation, and evaluation sets. We transformed speech waveforms into 80-D log mel filterbank energies using a 25ms Hanning window with a 10ms stride, which were applied as input for the encoders. The text data was encoded using a byte-pair encoding (BPE) [22], [23], resulting in 500 subword units, which were used as input for predictors.

2) Neural Network Architecture: For the transducer network, we employed the Zipformer [24] from an existing recipe.[1] The Zipformer consists of an encoder with a $4\times$ total subsampling rate, a stateless prediction network [25], and a joint network followed by a softmax layer. From the recipe, we modified only a few parameters regarding small blocks: the number of small blocks in each block to 2, feedforward dim to 768, encoder dim to 256, and encoder unmasked dim to 192. For the CausalEncoder$_\theta(\cdot)$, we used an attentional block chunk size of 8, resulting in 160ms of encoder latency.

The FoCCE network ω comprises three main components:

- CausalEncoder$^{\chi}_{\omega}(\cdot)$: eight stacks of chunk-wise causal convolution modules, which are identical to those in the Zipformer encoder1 but with a kernel size of 9.
- Predictor$^{\chi}_{\omega}(\cdot)$: a 128-D LSTM layer.
- DensityEstimator$^{\chi}_{\omega}(\cdot)$: masked autoregressive flow (MAF) [26] with a flow depth of 1, two neural blocks per each depth, and a hidden dimension of 160.

To align the chunk boundaries of encoders of the FoCCE network and the transducer network, we stacked the acoustic features into $4\times$ stacked features along the time-axis so that their dimension is 320, which were processed by CausalEncoder$^{\chi}_{\omega}(\cdot)$ to generate a causal context vector. This vector, in combination with the output from Predictor$^{\chi}_{\omega}(\cdot)$, acts as the condition for the MAF network DensityEstimator$^{\chi}_{\omega}(\cdot)$; Our model employs a standard Gaussian distribution as the prior.

3) Training: We trained the models for 40 epochs to ensure convergence. We applied SpecAugment [27] 1 to the acoustic features exclusively for the transducer encoder input. We experimentally determined the FoCCE hyperparameters that minimized WERs such that $\lambda_\gamma = 0.05$, $\lambda_{mod} = 1$, and $\lambda_\chi = 0.01$. For all the experiments of streaming ASR, we used the identical parameter setting for the transducer network. Therefore all transducers have the same computational footprints at inference, and their performances are affected only by the likelihood estimation at training.

4) Evaluation: We assessed the performance of the transducer models in terms of word error rates (WER) using a beam search algorithm with a beam size of 4. We incorporated a left context of 256 frames while evaluating streaming models.

B. Accuracy Improvement by FoCCE Training

Table I displays the WERs for Zipformer transducers trained using different methods. In LibriSpeech, FoCCE training on streaming transducers resulted in lower WERs, which amounts to 26.3% in test-clean and 12.3% in test-other of WER gaps between non-streaming and streaming baselines. FoCCE training also reduced such a gap by 17.7% in TED-LIUM3 test set. The extent of WER improvement was sensitive to the hyperparameter λ_γ, which arises from the fact that λ_γ is calculated based on the division of two probability densities in continuous feature space, rather than two probabilities in discrete output space.

V. FUTURE WORK

We introduce FoCCE, the estimator for FoCC, that helps mitigate the deformed likelihood problem. Our experiments show that FoCCE reduces WERs in streaming transducer training. Future research should focus on refining the estimation process for FoCCE.

[1]The Zipformer architecture and the training recipe can be found at https://github.com/k2-fsa/icefall/blob/master/egs/librispeech/ASR/zipformer. The **small-scaled model** recipe is used in this paper.

REFERENCES

[1] C. Peyser, T. N. Sainath, and G. Pundak, "Improving proper noun recognition in end-to-end ASR by customization of the MWER loss criterion," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, 2020, pp. 7789–7793.

[2] K. C. Sim et al., "Personalization of end-to-end speech recognition on mobile devices for named entities," in *Proc. IEEE Autom. Speech Recognit. Understanding Workshop*, 2019, pp. 23–30.

[3] Y. Zhang et al., "Google USM: Scaling automatic speech recognition beyond 100 languages," 2023, *arXiv:2303.01037*.

[4] V. Pratap et al., "Scaling speech technology to 1,000 languages," *J. Mach. Learn. Res.*, vol. 25, pp. 1–52, 2024.

[5] 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.

[6] 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.

[7] J. Yu et al., "Dual-mode ASR: Unify and improve streaming ASR with full-context modeling," in *Proc. Int. Conf. Learn. Representations*, 2021.

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

[9] 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.

[10] A. Graves, "Sequence transduction with recurrent neural networks," 2012, *arXiv:1211.3711*.

[11] 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.

[12] 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.

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

[14] H. Xiang and Z. Ou, "CRF-based single-stage acoustic modeling with CTC topology," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, 2019, pp. 5676–5680.

[15] 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.

[16] E. Variani, K. Wu, M. D. Riley, D. Rybach, M. Shannon, and C. Allauzen, "Global normalization for streaming speech recognition in a modular framework," in *Proc. Adv. Neural Inf. Process. Syst.*, 2022, pp. 4257–4269.

[17] Y.-A. Chung, W.-N. Hsu, H. Tang, and J. Glass, "An unsupervised autoregressive model for speech representation learning," in *Proc. Int. Speech Commun. Assoc.*, 2019, pp. 146–150.

[18] D. Rezende and S. Mohamed, "Variational inference with normalizing flows," in *Proc. Int. Conf. Mach. Learn.*, 2015, pp. 1530–1538.

[19] K2-FSA, "Icefall [Source Code]," 2021. [Online]. Available: https://github.com/k2-fsa/icefall

[20] 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.

[21] F. Hernandez, V. Nguyen, S. Ghannay, N. Tomashenko, and Y. Esteve, "TED-LIUM 3: Twice as much data and corpus repartition for experiments on speaker adaptation," in *Proc. 20th Int. Conf. Speech Comput.*, Leipzig, Germany, 2018, pp. 198–208.

[22] P. Gage, "A new algorithm for data compression," *C Users J.*, vol. 12, no. 2, pp. 23–38, 1994.

[23] R. Sennrich, B. Haddow, and A. Birch, "Neural machine translation of rare words with subword units," in *Proc. ACL*, 2016, pp. 1715–1725.

[24] Z. Yao et al., "Zipformer: A faster and better encoder for automatic speech recognition," in *Proc. Int. Conf. Learn. Representations*, 2024.

[25] 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.

[26] M. Germain, K. Gregor, I. Murray, and H. Larochelle, "Made: Masked autoencoder for distribution estimation," in *Proc. Int. Conf. Mach. Learn.*, 2015, pp. 881–889.

[27] D. S. Park et al., "SpecAugment: A simple data augmentation method for automatic speech recognition," in *Proc. Int. Speech Commun. Assoc.*, 2019, pp. 2613–2617.