Label-Synchronous Neural Transducer for Adaptable Online E2E Speech Recognition

Keqi Deng, *Graduate Student Member, IEEE*, and Philip C. Woodland, *Fellow, IEEE*

***Abstract*—Although end-to-end (E2E) automatic speech recognition (ASR) has shown state-of-the-art recognition accuracy, it tends to be implicitly biased towards the training data distribution which can degrade generalisation. This paper proposes a label-synchronous neural transducer (LS-Transducer), which provides a natural approach to domain adaptation based on text-only data. The LS-Transducer extracts a label-level encoder representation before combining it with the prediction network output. Since blank tokens are no longer needed, the prediction network performs as a standard language model, which can be easily adapted using text-only data. An Auto-regressive Integrate-and-Fire (AIF) mechanism is proposed to generate the label-level encoder representation while retaining low latency operation that can be used for streaming. In addition, a streaming joint decoding method is designed to improve ASR accuracy while retaining synchronisation with AIF. Experiments show that compared to standard neural transducers, the proposed LS-Transducer gave a 12.9% relative WER reduction (WERR) for intra-domain LibriSpeech data, as well as 21.4% and 24.6% relative WERRs on cross-domain TED-LIUM 2 and AESRC2020 data with an adapted prediction network.**

***Index Terms*—Domain adaptation, E2E ASR, neural transducer.**

I. Introduction

THE hybrid deep neural network and hidden Markov model (DNN-HMM) [1], [2], [3] approach for automatic speech recognition (ASR) is a widely-used deep learning-based framework, which contains several separately optimised modules, including the acoustic model, pronunciation lexicon, context dependency model [4], and language model (LM). However, the separately optimised models make it hard to achieve a globally optimised system [5]. End-to-end (E2E) ASR simplifies the modelling pipeline and integrates the separate modules used by the DNN-HMM approach [6]. Notable E2E ASR methods include connectionist temporal classification (CTC) [7], the neural transducer [8], and the attention-based encoder-decoder (AED) [9], [10]. Among these techniques, the neural transducer provides a natural approach for streaming ASR that can give a high accuracy, and has become popular for industrial applications [11].

The hybrid DNN-HMM framework is, however, still used in many industrial ASR systems [11]. Several practical advantages of hybrid systems contribute to this, including low-latency streaming and domain adaptation capabilities [11]. In the past few years, the streaming features of E2E ASR have been extensively explored [12] and neural transducers can replace hybrid DNN-HMMs in some cases [13]. However, the hybrid DNN-HMM approach is highly modular and contains an explicit independent LM, making it straightforward to bias the recognition system to unseen domains using text-only data. The neural transducer, which is the most commonly deployed E2E model, is still weaker than HMM-based systems in this regard. This is because E2E ASR models jointly learn acoustic and linguistic information [14] and do not have an explicit LM that can be used for flexible domain adaptation with text-only data. For the standard neural transducer, in which speech is decoded on a per-frame basis i.e. frame-synchronously, blank tokens are used to augment the output sequence thus allowing the frame-level encoder output to be combined with the label-level prediction network output [8]. However, blank token generation means that the prediction network cannot be viewed as an explicit LM [15] due to the inconsistency with the LM task [15], [16] and thus poses a challenge to text-only domain adaptation.

The motivation of this paper is to modify the standard neural transducer by enabling an acoustic encoder representation to be directly combined with the prediction network output at the label level and hence not need blank tokens. Therefore, operation is label-synchronous and the prediction network performs as a standard LM. This makes it straightforward to biased the model to previously unseen domains using text-only data. In this paper, a label-synchronous neural transducer (LS-Transducer) is proposed[1] that provides a natural approach to domain adaptation, while retaining the valuable streaming property and E2E training simplicity.

The main contributions of this work can be summarised in four key parts:

- *LS-Transducer:* A label-synchronous neural transducer (LS-Transducer) is proposed that extracts a label-level encoder representation before combining it with the prediction network output. The LS-Transducer improves E2E

Manuscript received 10 November 2023; revised 21 March 2024 and 10 June 2024; accepted 13 June 2024. Date of publication 26 June 2024; date of current version 26 July 2024. The work of Keqi Deng was supported by the Cambridge Trust. This work has been performed using resources provided by the Cambridge Tier-2 system operated by the University of Cambridge Research Computing Service (www.hpc.cam.ac.uk) funded by EPSRC Tier-2 capital under Grant EP/T022159/1. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Rohit Prabhavalkar. *(Corresponding author: Philip C. Woodland.)*

The authors are with the Department of Engineering, University of Cambridge, CB2 1TN Cambridge, U.K. (e-mail: kd502@cam.ac.uk; pw117@cam.ac.uk).

This article has supplementary downloadable material available at https://doi.org/10.1109/TASLP.2024.3419421, provided by the authors.

Digital Object Identifier 10.1109/TASLP.2024.3419421

[1] An earlier and less complete description of the work is available at [17].

ASR domain adaptation while maintaining low ASR error rates and good streaming properties.

- *Auto-regressive Integrate-and-Fire (AIF):* In order to extract the label-level encoder representation for the LS-Transducer, an AIF mechanism is proposed, which is extended from the Continuous Integrate-and-Fire (CIF) [18] approach but with improved efficiency and robustness to inaccurate unit boundaries.
- *Prediction Network Adaptation:* The LS-Transducer prediction network performs as an explicit LM, so can be easily fine-tuned on target-domain text.
- *Streaming Joint Decoding:* This paper also proposes a streaming joint decoding method to enhance the accuracy of the LS-Transducer by utilising an online CTC prefix score, which is synchronised with the AIF alignment through a simple but effective modification to the standard CTC prefix score.

Experiments with ASR models trained on LibriSpeech data [19] show that the proposed LS-Transducer gives reduced WERs over standard neural transducer models for both intra-domain and cross-domain scenarios.

The rest of this paper is organised as follows. Section II reviews related works including CIF on which the AIF technique is based. Section III describes the proposed LS-Transducer framework. Section IV details the experimental setups and Section V presents the results. Finally, Section VI concludes.

II. Related Work

A. Neural Transducer Models

The neural transducer [8] has gained widespread interest and is the leading E2E model deployed in industry [11]. The neural transducer removes the independence assumption in CTC between output tokens by conditioning on previous non-blank output tokens [20]. When aligning input speech and output token sequences, the neural transducer aligns the sequences at the frame level by inserting a blank token to augment output sequences. Consequently, the neural transducer output is conditioned on the speech sequence up to the current time step, providing a natural approach for streaming ASR [11]. This is in contrast to the AED [9] which relies on a global attention mechanism that hampers streaming processing or incurs significant latency [21],

The neural transducer contains an encoder network, a prediction network, and a joint network. The encoder network extracts an acoustic representation $\boldsymbol{h}_t^{\text{enc}}$ from input speech $\boldsymbol{x}$. The encoder network can use a long short-term memory (LSTM) [22], Transformer [10], or Conformer [23] structure. However, when aimed at streaming, strategies like the chunk-based or lookahead-based method [13] need to be employed to achieve a streaming Transformer/Conformer encoder.

The prediction network allows the neural transducer to capture causal dependencies in the output by generating a representation $\boldsymbol{h}_n^{\text{pre}}$ from previous non-blank tokens $y_{1:n-1}$. The prediction network is an auto-regressive structure and can employ an RNN [8], unidirectional Transformer [24] or even only an embedding layer [15]. The prediction network normally has a similar structure to an LM, but it does not perform as an explicit LM because it also needs to predict blank tokens, which is inconsistent with the LM task [16].

The joint network combines $\boldsymbol{h}_t^{\text{enc}}$ and $\boldsymbol{h}_n^{\text{pre}}$ at the frame level with fully-connected (FC) networks and the output logits $\boldsymbol{l}_{t,n}$ can be computed as:

$$\boldsymbol{l}_{t,n} = \text{FC}(\Psi(\text{FC}(\boldsymbol{h}_t^{\text{enc}}) + \text{FC}(\boldsymbol{h}_n^{\text{pre}}))) \tag{1}$$

where Ψ is a non-linear activation function and the predicted probability of the k-th token is obtained by applying a softmax function to the logits $\boldsymbol{l}_{t,n}$:

$$p(y_n = k|\boldsymbol{x}_{1:t}, y_{1:n-1}) = \text{softmax}(\boldsymbol{l}_{t,n}) \tag{2}$$

where $\boldsymbol{x}_{1:t}$ denotes the speech sequence up to frame t. The neural transducer loss function $\mathcal{L}_{\text{nt}}$ is defined as the negative log-likelihood of the target text sequence $\boldsymbol{y}$ of length N:

$$p(\boldsymbol{a}|\boldsymbol{x}) \approx \prod_{u=1}^{T+N} p(a_u|\text{A}(a_{1:u-1}), \boldsymbol{x}) \tag{3}$$

$$\mathcal{L}_{\text{nt}} = -\ln \sum_{\boldsymbol{a} \in A^{-1}} p(\boldsymbol{a}|\boldsymbol{x}) \tag{4}$$

where T is the total length of $\boldsymbol{x}$ and A is a collapsing function that maps all alignment paths $\boldsymbol{a}$ to the target text sequence.

B. Continuous Integrate-and-Fire (CIF)

CIF [18] is presented as background since the AIF mechanism in the LS-Transducer is an extension of CIF. CIF estimates a monotonic alignment for streaming ASR. In CIF, the first step involves learning a weight α_t for each encoder output frame $\boldsymbol{e}_t$. To obtain this weight, a one-dimensional scalar is generated from each encoder output $\boldsymbol{e}_t$ through convolutional or fully-connected layers [18] or even directly using a particular element of $\boldsymbol{e}_t$ [25]. Using a sigmoid function, the weight α_t is then computed from this scalar. Next, the CIF mechanism accumulates the weights over time and integrates the acoustic representation via a weighted sum. The accumulation continues until the accumulated weight is above a threshold of 1.0. At this point, the current weight α_t is split into two parts: one part is used to make the current accumulated weight be exactly 1.0, while the remainder is used for the integration of the next label. CIF then "fires" the integrated acoustic representation $\boldsymbol{c_j}$ corresponding to label y_j and resets the accumulation.

As shown the example in Fig. 1, if the weights $(\alpha_1, \ldots, \alpha_T)$ generated by CIF are $(0.2, 0.9, 0.2, 0.3, 0.6, 0.1 \cdots)$, the $\alpha_2 = 0.9$ needs to be divided into $\alpha_{2,1} = 0.8$ and $\alpha_{2,2} = 0.1$, so that $\alpha_1 + \alpha_{2,1} = 1.0$ and $\boldsymbol{c}_1 = 0.2\boldsymbol{e}_1 + 0.8\boldsymbol{e}_2$ can be emitted. The same situation will also occur when $\alpha_5 = 0.6$, which needs to be divided into $\alpha_{5,1} = 0.4$ and $\alpha_{5,2} = 0.2$, so that $\alpha_{2,2} + \alpha_3 + \alpha_4 + \alpha_{5,1} = 1.0$ and $\boldsymbol{c}_2 = 0.1\boldsymbol{e}_2 + 0.2\boldsymbol{e}_3 + 0.3\boldsymbol{e}_4 + 0.4\boldsymbol{e}_5$ can be emitted. The calculation of $\boldsymbol{c}_3$, $\boldsymbol{c}_4$, etc., are similar and applied until the end of the encoder output.

During training, a scaling strategy is used to ensure that the integrated acoustic representations $\mathbf{C}=(\boldsymbol{c}_1, \ldots, \boldsymbol{c}_L)$ have the same length L as the target sequence. This strategy involves computing a scaled weight, $\hat{\alpha}_t$, which is obtained by $\hat{\alpha}_t = \alpha_t \cdot (L / \sum_{i=1}^{T} \alpha_i)$. By using $\hat{\alpha}_t$ instead of α_t to extract $\mathbf{C}$,

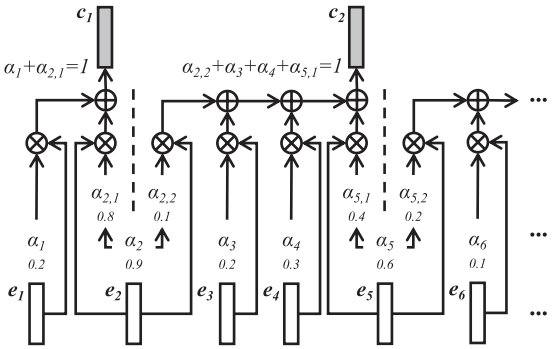

Fig. 1. Example of CIF [18]. $\oplus$ and $\otimes$ denote addition and multiplication. $\mathbf{E}=(\boldsymbol{e_1},\ldots,\boldsymbol{e_T})$ denotes encoder output and $\boldsymbol{\alpha}=(\alpha_1,\ldots,\alpha_T)$ represents predicted weights whose example values are $(0.2, 0.9, 0.2, 0.3, 0.6, 0.1\cdots)$.

the length is effectively controlled. However, during decoding, the length of integrated representations is solely determined by the weight accumulation $\sum_{i=1}^{T}\alpha_i$. Hence, a quantity loss $\mathcal{L}_{\text{qua}}=|\sum_{i=1}^{T}\alpha_i - L|$, defined as the absolute difference between $\sum_{i=1}^{T}\alpha_i$ and the target length L, is used to supervise CIF to extract a number of integrated representations close to L.

Note that CIF doesn't always locate the real acoustic boundaries and thus accurately estimate the text length [26], [27]. This is especially true for English ASR tasks using units such as BPE [28].[2] In addition, since the scaling strategy is used during training, a mismatch exists between training and decoding. Moreover, CIF is a sequential method [27], [29] since it needs to know the time step that the previous label representation was emitted, before the weight accumulation mechanism is reset for the next label. This can lead to reduced training efficiency.

C. Text Domain Adaptation

Various methods have been developed for E2E ASR domain adaptation using text-only data. One solution involves LM fusion where an external LM is integrated into the E2E ASR system [30], [31], of which the most commonly used is shallow fusion [30]. However, the E2E ASR model implicitly learns an internal LM that characterises the source domain training data distribution [32], which causes a mismatch when decoded on unseen domains. To solve this issue, this internal LM can be estimated [32], [33], [34], [35], [36], [37]. For example, HAT [34] was proposed to estimate the internal LM by removing the acoustic encoder effect from the neural transducer. Nonetheless, estimating the internal LM score increases the complexity of decoding and achieving accurate estimation is not always feasible due to domain mismatch [38]. More recent studies [16], [39], [40] such as the factorised neural transducer [16], have investigated fine-tuning the internal LM using target-domain text. However, this approach can lead to intra-domain performance degradation [16]. The use of Kullback-Leibler divergence regularisation mitigates this issue but limits how much the internal

[2]CIF gained popularity in Mandarin ASR tasks because Chinese characters correspond to clear syllable boundaries. However, CIF suffers from performance degradation on English ASR due to difficulty in locating boundaries for BPE units, as mentioned in [26]. While preliminary experiments show that the proposed LS-Transducer is very effective on Mandarin data, this paper focuses on English ASR.

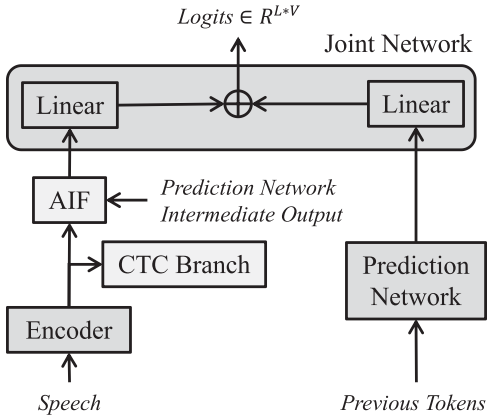

Fig. 2. Illustration of the proposed LS-Transducer. Linear denotes linear classifier. The output $logits$ is a label-level two-dimensional matrix, where L and V are the length and vocabulary size. $\oplus$ denotes addition.

LM learns the target domain [39], [40]. Another solution involves using Text-to-Speech (TTS) to synthesise speech from the target-domain text, which is then employed for fine-tuning the ASR models [41], [42]. However, this method incurs significant computational cost and lacks flexibility for fast adaptation [16].

III. Label-Synchronous Neural Transducer

As shown in Fig. 2, the proposed LS-Transducer includes the AIF mechanism to extract a label-level encoder representation before combining it with the prediction network output. This is the main difference with the standard neural transducer which directly combines the frame-level encoder output with the label-level prediction network output. AIF ensures the extracted label-level encoder representation is strictly synchronised with the prediction network output by querying its intermediate output, enabling the prediction network to work as a standard LM. The joint network in the LS-Transducer then adds the logits obtained from the AIF and prediction network outputs through linear fully-connected layers. This design enables the prediction network to be flexibly biased to unseen domains on text-only data, without affecting other parts of the model. Note that the joint network output, as shown in Fig. 2, is a 2-dimensional matrix $\mathbb{R}^{L\cdot V}$, which differs from the standard neural transducer where the output is a 3-dimensional tensor $\mathbb{R}^{T\cdot L\cdot V}$ with an extra time dimension.

During training, with the help of the proposed AIF mechanism, the logits (as in Fig. 2) in the LS-Transducer will have the same length as the target sequence, therefore the cross-entropy (CE) loss $\mathcal{L}_{\text{ce}}$ can be computed between them and used as the training objective. Computing this CE loss can also help save a considerable amount of memory compared to the RNN-T loss that is computed based on three-dimensional tensors [11]. In addition, the AIF mechanism calculates the quantity loss $\mathcal{L}_{\text{qua}}$ to learn an explicit speech-text alignment, which will be described in detail in Section III-A. Inspired by [43], which shows that CTC [7] always helps model training for both AED and neural transducers, CTC-based supervision $\mathcal{L}_{\text{ctc}}$ is also used by the encoder in the LS-Transducer. Therefore, the overall training

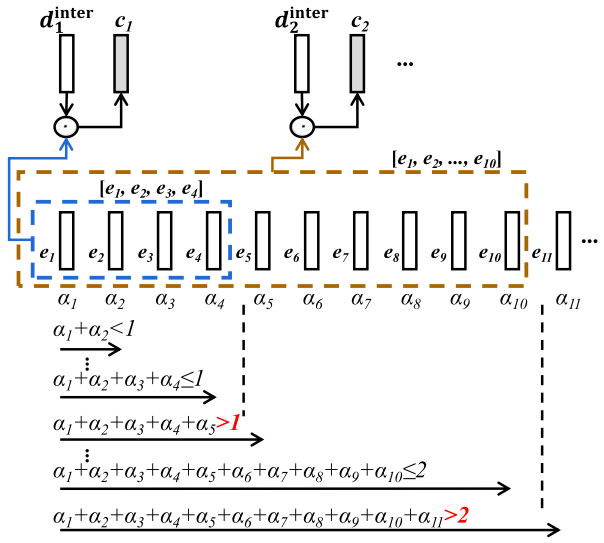

Fig. 3. Illustration of the proposed AIF. $\odot$ denotes dot-product attention, whose query $\boldsymbol{d}_j^{\text{inter}}$ is the intermediate output of the prediction network.

objective $\mathcal{L}_{\text{lst}}$ of the LS-Transducer is:

$$\mathcal{L}_{\text{lst}} = \gamma\mathcal{L}_{\text{ctc}} + (1-\gamma)\mathcal{L}_{\text{ce}} + \mu\mathcal{L}_{\text{qua}} \cdot L \tag{5}$$

where L is the target length, and γ and μ are hyper-parameters.

A. Auto-Regressive Integrate-and-Fire (AIF)

The AIF mechanism is proposed in this paper, which extracts label-level representations $\mathbf{C} = (\boldsymbol{c}_1, \ldots, \boldsymbol{c}_L)$ from the acoustic encoder output $\mathbf{E} = (\boldsymbol{e}_1, \ldots, \boldsymbol{e}_T)$. Extended from CIF [18], AIF retains the streaming property and also uses accumulated weights α_t to locate boundaries and thus decide when to fire a label-level representation $\boldsymbol{c}_j$. However, AIF resolves a number of issues present in CIF, thus improving the LS-Transducer recognition accuracy. The main difference to CIF is that AIF uses dot-product attention instead of the weights α_t to extract the $\boldsymbol{c}_j$, after locating the boundaries based on the accumulated weights α_t.

To be more specific, as shown in Fig. 3, AIF first computes a weight α_t for each encoder output frame $\boldsymbol{e}_t$, and this weight can be obtained using a sigmoid function, after transforming $\boldsymbol{e}_t$ into a scalar by neural networks or even simply selecting a particular element. The next step is locating the boundaries corresponding to the ASR modelling unit. To decide when to fire the label-level representation $\boldsymbol{c}_j$ where $j \in (1, L)$, the weight α_t will be accumulated from left to right until it exceeds j (j is both the index and threshold for $\boldsymbol{c}_j$), and then the time step of the located boundary for $\boldsymbol{c}_j$ is recorded as T_j+1. If the j isn't reached until all T frames have been read (i.e. $\sum_{i=1}^{T} \alpha_i \leq j$), $T_j = T$. When firing the label-level representation $\boldsymbol{c}_j$, AIF employs dot-product attention, where $\mathbf{E}_{1:T_j}$ is used as the keys and values:

$$\boldsymbol{c_j} = \text{softmax}(\boldsymbol{d}_j^{\text{inter}} \cdot \mathbf{E}_{1:T_j}{}^{\top}) \cdot \mathbf{E}_{1:T_j} \tag{6}$$

where the query $\boldsymbol{d}_j^{\text{inter}}$ is the prediction network intermediate output at the j-th step. Note the use of $\mathbf{E}_{1:T_j}$ is an important difference from AED-based [9] models, in which all speech input (i.e. $\mathbf{E}_{1:T}$) is used by the attention module. As shown in Fig. 3, after the $\boldsymbol{c}_j$ is extracted, the accumulation of the weight α_t will continue to locate the boundaries of $\boldsymbol{c}_{j+1}$, this process is carried out incrementally until the last $\boldsymbol{c}_L$ is obtained. When finding $\boldsymbol{c}_L$, AIF is closer to the AED since the complete $\mathbf{E}_{1:T}$ is available to be used as the keys and values.

As shown in the example in Fig. 3, when generating the first representation $\boldsymbol{c}_1$, the accumulated weight α_t exceeds 1 at the 5-th time step (i.e. $\sum_{i=1}^{5} \alpha_i > 1$ and $\sum_{i=1}^{4} \alpha_i \leq 1$), thus $\mathbf{E}_{1:4}$ is used as the keys and values to extract $\boldsymbol{c}_1$ with $\boldsymbol{d}_1^{\text{inter}}$ as the query. After that, the weights α_t continue to be accumulated to find the time step when the accumulation exceeds 2, which is at the 11-th time step in this example. Therefore, $\mathbf{E}_{1:10}$ is used as the keys and values to extract $\boldsymbol{c_2}$ with query $\boldsymbol{d}_2^{\text{inter}}$. Subsequent extraction for $\boldsymbol{c}_3$, $\boldsymbol{c}_4$, etc., follows a similar rule.

The joint network combines the logits obtained from AIF and the prediction network at the label level, and the output logits $\boldsymbol{l}_j$ can be computed as:

$$\boldsymbol{l}_j = \text{FC}(\boldsymbol{c_j}) + \text{FC}(\boldsymbol{d}_j^{\text{final}}) \tag{7}$$

where $\boldsymbol{d}_j^{\text{final}}$ denotes the final output of the prediction network. FC denotes linear output layers that map the dimension to the vocabulary size.

During training, in order to encourage AIF to learn accurate speech-text alignments and thus locate the correct boundaries, an explicit objective i.e. quantity loss $\mathcal{L}_{\text{qua}}$ will be computed:

$$\mathcal{L}_{\text{qua}} = |\sum_{i=1}^{T} \alpha_i - L| \tag{8}$$

This alignment learning is explicit and independent of recognition, as these weights $(\alpha_1, \ldots, \alpha_T)$ are not used to extract label-level representations $\mathbf{C} = (\boldsymbol{c}_1, \ldots, \boldsymbol{c}_L)$ and thus predict the target labels, hence distinguishing it from other E2E methods such as standard neural transducers, CTC, and AED.

In general, AIF generates the label-level representation in an auto-regressive manner, which has many advantages compared to conventional CIF. First of all, as mentioned in Section II-B, the length of label-level representation $\mathbf{C}$ extracted by CIF is solely determined by the value of accumulated weights $\sum_{i=1}^{T} \alpha_i$ and thus needs to employ a scaling strategy during training. However, this scaling strategy relies on the ground truth of target length and is not accessible during decoding, causing a mismatch between training and decoding. However, in AIF, the length of $\mathbf{C}$ is decided by the number of queries, so the scaling strategy is not used and the mismatch issue does not arise. Second, AIF has a higher training speed due to its ability to generate label-level representations in parallel with teacher forcing by masking certain attention weights, while, CIF, as discussed in Section II-B, is a sequential approach. Third, although the boundaries located by the accumulated weights α_t might not always be accurate, as shown in the dashed box in Fig. 3, AIF shows flexibility in tackling this issue by taking the first frame as the left boundary when extracting the $\boldsymbol{c}_j$.

B. Streaming Joint Decoding

Due to the proposed AIF mechanism, the LS-Transducer is naturally equipped with streaming decoding. In addition, considering the LS-Transducer uses the CTC branch to help model training, a streaming joint decoding method is proposed which computes an online CTC prefix score synchronously

with the LS-Transducer predictions to refine the search space and eliminate irrelevant alignments. The standard CTC prefix score is inapplicable in streaming scenarios because it requires a complete speech utterance. Suppose g is a partial hypothesis, q is a token appended to g, and $h = g \cdot q$ is a new hypothesis. When q is a normal vocabulary token (i.e. not end-of-sentence [eos]), the CTC prefix score S_{ctc} of h in [44] is calculated as:

$$p_{\text{ctc}}(h, \cdots|\mathbf{E}) = \sum_{\nu \in (U \cup [\text{eos}])} p_{\text{ctc}}(h \cdot \nu|\mathbf{E}) \tag{9}$$

$$S_{\text{ctc}}(h, \mathbf{E}) = \log(p_{\text{ctc}}(h, \cdots|\mathbf{E})) \tag{10}$$

where p_{ctc} refers to the sequence probability[3] given by CTC, e.g. $p_{\text{ctc}}(h \cdot \nu|\mathbf{E})$ represents the probability of the sequence $h \cdot \nu$ given the entire encoder output $\mathbf{E}$. In this context, ν represents all possible non-empty tokens (with U denoting normal tokens), and $h \cdot \nu$ means appending ν to h. Therefore, the CTC prefix score is calculated as the accumulated probability of all sequences with h as their prefix [44]. However, if q (i.e. the last token of h) is [eos], the CTC score is computed as:

$$S_{\text{ctc}}(h, \mathbf{E}) = \log(\gamma_T^{(n)}(g) + \gamma_T^{(b)}(g)) \tag{11}$$

where $\gamma_T^{(n)}(g)$ and $\gamma_T^{(b)}(g)$ are the forward probabilities [44] of the partial hypothesis g over T frames, with CTC paths ending with a non-blank or blank label, respectively. These CTC prefix scores are computed based on whole encoder output $\mathbf{E}$ with T frames, hampering streaming decoding.

To compute an online CTC prefix score for streaming joint decoding, inspired by [45], this paper uses $S_{\text{ctc}}(h, \mathbf{E}_{1:T_h})$ to approximate $S_{\text{ctc}}(h, \mathbf{E})$, where T_h is the maximum number of encoder output frames accessible when predicting the new hypothesis h, which is decided by the accumulated weights α_t of AIF. Therefore, the online CTC prefix score and LS-Transducer prediction are strictly synchronised.

However, since CTC has too much flexibility when learning alignments, there is no theoretical guarantee that the CTC spikes and AIF-located boundaries will be synchronised. When the corresponding CTC spike for token q doesn't appear during $\mathbf{E}_{1:T_h}$, preliminary experiments showed that the online CTC prefix score $S_{\text{ctc}}(h, \mathbf{E}_{1:T_h})$ would be very likely to predict [eos] because h would be considered complete for $\mathbf{E}_{1:T_h}$, which could greatly degrade the performance. Previous work tackles this problem by waiting until the corresponding CTC spike appeared before starting decoding [46] or switching to decoding the next block of speech if the [eos] label is predicted [45]. However, these methods are not feasible for the proposed LS-Transducer that needs to compute online CTC scores synchronously.

To solve this issue, a proposed streaming joint decoding method modifies the computation of online CTC prefix scores for [eos], which is shown as follows where $h{=}g \cdot [\text{eos}]$:

$$S_{\text{ctc}}(h, \mathbf{E}_{1:T_h}) = \begin{cases} \log(\text{p}_{\text{ctc}}(\text{h}, \cdots|\mathbf{E}_{1:\text{T}_\text{h}})), & T_h < T \\ \log(\gamma_{T_h}^{(n)}(g) + \gamma_{T_h}^{(b)}(g)), & T_h = T \end{cases} \tag{12}$$

This means that if the speech has not been completely loaded (i.e. $T_h < T$), h will not be considered complete, leading to an extremely low score for [eos] because CTC training never encounters the [eos] label. This makes sense because the online CTC prefix score should not consider ending prediction before reading the whole speech utterance.

[3]See [44] for detailed computation of the CTC-based sequence probability.

Algorithm 1: Modified Online CTC Prefix Score.

Input: $h, \mathbf{E}_{1:T_h}$
Output: S_{ctc}
1: $g, q \leftarrow h$: Split h into the last label q and the rest g
2: **if** $q = [\text{eos}]$ **and** $T_h = T$ **then**
3: **return** $\log(\gamma_{T_h}^{(n)}(g) + \gamma_{T_h}^{(b)}(g))$
4: **else**
5: $\gamma_1^{(n)}(h) \leftarrow \begin{cases} p(z_1 = q|\mathbf{E}_{1:T_h})), & \text{if } g = [\text{sos}] \\ 0, & \text{otherwise} \end{cases}$
6: $\gamma_1^{(b)}(h) \leftarrow 0$
7: $\Psi \leftarrow \gamma_1^{(n)}(h)$
8: **for** $t = 2 \cdots T_h$ **do**
9: $\Phi \leftarrow \gamma_{t-1}^{(b)}(g) + \begin{cases} 0, & \text{if last}(g) = q \\ \gamma_{t-1}^{(n)}(g), & \text{otherwise} \end{cases}$
10: $\gamma_t^{(n)}(h) \leftarrow (\gamma_{t-1}^{(n)}(h) + \Phi) p(z_t = q|\mathbf{E}_{1:T_h})$
11: $\gamma_t^{(b)}(h) \leftarrow (\gamma_{t-1}^{(b)}(h) + \gamma_{t-1}^{(n)}(h)) p(z_t = \text{blank}|\mathbf{E}_{1:T_h})$
12: $\Psi \leftarrow \Psi + \Phi \cdot p(z_t = q|\mathbf{E}_{1:T_h})$
13: **return** $\log(\Psi)$

The detailed procedure for the modified online CTC prefix score is shown in Algorithm 1, which modifies the condition in line 2 compared to the standard CTC prefix score [44]. z_t and $p(z_t = q|\mathbf{E}_{1:T_h})$ are the label and probability for the t-th frame. [sos] is start-of-sentence. Other details follow [44].

During streaming joint decoding, score S_{lst} assigned by the LS-Transducer is computed synchronously with $S_{\text{ctc}}(h, \mathbf{E}_{1:T_h})$ and follows the chain rule:

$$S_{\text{lst}}(h, \mathbf{E}_{1:T_h}) = \sum_{i=1}^{n} \log(p_{\text{lst}}(h_i|h_1, \ldots, h_{i-1}, \mathbf{E}_{1:T_i})) \tag{13}$$

where p_{lst} denotes the predicted probabilities obtained from the final logits output by the joint network, as shown in Fig. 2, n is the length of hypothesis $h = g \cdot q$, and T_i is the corresponding right boundary of the i-th label as decided by AIF. The overall streaming score S is computed as:

$$S(h, \mathbf{E}_{1:T_h}) = \beta S_{\text{ctc}}(h, \mathbf{E}_{1:T_h}) + (1 - \beta) S_{\text{lst}}(h, \mathbf{E}_{1:T_h}) \tag{14}$$

where β denotes the weight of online CTC scores. Hence, the streaming scores of the LS-Transducer $S_{\text{lst}}(h, \mathbf{E}_{1:T_h})$ and the CTC branch $S_{\text{ctc}}(h, \mathbf{E}_{1:T_h})$ are strictly synchronised.

C. Prediction Network Adaptation

With the prediction network of the LS-Transducer performing as an explicit LM, fine-tuning it on text-only data when encountering a domain shift is straightforward. Therefore, after ASR training and before decoding on an unseen domain, when target-domain text data set $\mathcal{D}$ available, the fine-tuning objective is:

$$\mathcal{L}_{\text{finetune}} = -\sum_{\boldsymbol{Y} \in \mathcal{D}} \sum_{n=1}^{N} \log\, p_{\text{pred}}(y_n|\boldsymbol{Y}_{0:n-1}; \theta_{\text{pred}}) \tag{15}$$

where $\boldsymbol{Y} = ([sos], y_1, \ldots, y_N)$ is a text sequence belonging to $\mathcal{D}$ and y_n denotes the n-th token. θ_{pred} represents the parameters of the prediction network, and p_{pred} denotes the predicted probabilities of the prediction network, which are obtained by applying the softmax function to its output logits, i.e. linear classifier output as shown in Fig. 2.

D. Specific Implementation

In this paper, inspired by [25], a simple method is used to compute the weight α_t in AIF by applying a sigmoid function to the last $e_{t,d}$ of each encoder output frame $\boldsymbol{e_t}$:[4]

$$\alpha_t = \text{sigmoid}(e_{t,d}) \tag{16}$$

where d is the dimension of $\boldsymbol{e}_t$. Preliminary experiments showed that an auxiliary phone-based quantity loss helps model training, and the effectiveness of this specific implementation is evaluated in the Supplemental Materials. Hence, (8) is written as $\mathcal{L}_{\text{qua}} = |\sum_{i=1}^{T} \alpha_i - L| + |\sum_{i=1}^{T} w_i - P|$, where $w_t = \text{sigmoid}(e_{t,d-1})$ and P corresponds to the number of phone units in the utterance. Correspondingly, when AIF extracts the label-level representations $\boldsymbol{c}_j$, only the other elements of $\boldsymbol{e}_t$ are used, i.e. $\mathbf{e}_{t,1:d-2}$, so (6) of AIF can be expressed as follows in this specific implementation.

$$\boldsymbol{c_j} = \text{softmax}(\boldsymbol{d}_j^{\text{inter}} \cdot \text{FC}(\mathbf{E}_{1:T_j,1:d-2})^{\top}) \cdot \text{FC}(\mathbf{E}_{1:T_j,1:d-2}) \tag{17}$$

where FC denotes fully connected layers that map $\boldsymbol{e}_{t,1:d-2}$ to the same dimension as the $\boldsymbol{d}_j^{\text{inter}}$.

IV. Experimental Setup

A. Datasets

ASR models were trained on the LibriSpeech [19] data, a read audiobook corpus, and its dev/test sets (i.e. "test/dev-clean/other") were used for intra-domain evaluation. The source-domain text data included training set transcripts and LibriSpeech LM training text. To evaluate the domain adaptation capability of the LS-Transducer, two out-of-domain test corpora were used. The first corpus consisted of TED-LIUM2 [47] dev/test sets, comprising spontaneous lecture-style data. For target-domain adaptation text, the training set transcripts and TED-LIUM2 LM training text were used. The second corpus was AESRC2020 [48] dev/test sets, containing human-computer interaction speech commands, and the training set transcriptions were used as the target-domain text data.

The models and experimental evaluations were implemented based on the ESPnet2 [49] toolkit. Raw speech data was used as input, and 1000 ASR modelling units were used as text output, including 997 BPE [28] units and 3 non-verbal symbols (i.e. blank, unknown-character and start/end-of-sentence).

B. ASR Model Description

1) Standard Neural Transducer Models: Three standard Transformer transducer (T-T) [24] models, built with streaming wav2vec 2.0 encoders and different prediction networks, were compared to the proposed LS-Transducer. The T-T with an embedding layer as the prediction network is denoted the Stateless-Pred T-T (319 M parameters); the T-T with a 6-layer 1024-dimensional LSTM prediction network is referred to as the LSTM-Pred T-T (370 M parameters); and the T-T with a 6-layer unidirectional Transformer prediction network (1024 attention dimension, 2048 feed-forward dimension, and 8 heads) is called as the Transformer-Pred T-T (371 M parameters). Streaming wav2vec 2.0 encoders, based on a "w2v_large_lv_fsh_swbd_cv" [50], were built using a chunk-based mask [13] to enable streaming, with a 320 ms average latency. Inspired by [43], [51], three standard T-T models also used the CTC with 0.3 weight to help training. In addition, extra results with a conformer-based encoder are given in the Supplemental Materials which allow further detailed comparisons with previously published work.

2) LS-Transducer: The proposed LS-Transducer (373 M parameters) had the same streaming wav2vec2.0 encoder as the three standard T-T baseline models with a 320 ms theoretical average latency. A more detailed evaluation of the latency is given in the Supplemental Materials. The LS-Transducer had a unidirectional Transformer prediction network, which was the same as that of Transformer-Pred T-T. The intermediate output from the 3 rd sub-layer of the prediction network was employed as the query for the AIF mechanism. The linear layers in Fig. 2 mapped dimensions from 1024 to 1,000. In (5), the CTC loss was computed based on $\boldsymbol{e}_{t,1:d-2}$ where $d = 1024$, and γ and μ were respectively set to 0.5 and 0.05. In (14), β was set to 0.3 except for TED-LIUM2 which was 0.4.

3) Offline AED Model: An offline Transformer-based AED model (394 M parameters) was also built to compare with the online LS-Transducer. It used the same streaming wav2vec 2.0 encoder but underwent offline training and was decoded using offline CTC/attention joint decoding [44].

4) Related Variants of Standard Neural Transducer: Building upon the Transformer-Pred T-T structure, both factorised T-T [16] (372 M parameters) and HAT [34] (371 M parameters) were implemented with the same encoder and prediction network (called the vocabulary predictor in factorised T-T). For the factorised T-T, the embedding layer from the vocabulary predictor was directly used as the blank predictor.

C. LM Model and Text Adaptation

A source-domain 6-layer Transformer LM was trained on the source-domain text data for 25 epochs and fine-tuned on the target-domain text for an additional 15 epochs as a target-domain LM. The trained source-domain LM was used to initialise the prediction network of the LS-Transducer but not for the three standard T-T models because this didn't help enhance performance [15]. ASR training was for 40 epochs. When adapting the LS-Transducer prediction network as in (15), the first 3 layers were fixed, and the rest were fine-tuned on the target-domain text data with 50 epochs for AESRC2020 and 20 epochs for TED-LIUM2 data. Shallow fusion [30] was used with a 0.2 LM weight if using the target-domain LM for domain adaptation. A beam size of 10 was used during inference.

[4]LS-Transducer is not limited to this method of generating α_t. Other methods including convolutional or fully-connected layers could be used.

TABLE I
INITIAL EXPERIMENTS: INTRA-DOMAIN WER ON LIBRISPEECH DEV/TEST SETS FOR ONLINE TRANSDUCER MODELS TRAINED FROM LIBRISPEECH-100 H (TRAIN-CLEAN-100 SET)

Online ASR Models	Test		Dev	
	clean	other	clean	other
(Offline) W2v2 Transducer [52]	5.2	11.8	5.1	12.2
(Offline) Conformer Transducer [53]	5.9	16.9	–	–
Chunked Conformer Transducer [53]	6.8	20.4	–	–
(Offline) AED Model	4.4	11.3	4.2	11.3
Stateless-Pred T-T	5.6	12.6	5.5	12.6
LSTM-Pred T-T	5.3	12.5	5.1	12.5
HAT [34]	5.4	12.2	5.1	12.1
Factorised T-T [16]	5.4	12.4	5.3	12.4
Transformer-Pred T-T	5.1	12.0	4.9	12.0
Proposed LS-Transducer	**4.4**	**11.0**	**4.1**	**10.8**

Note that HAT and factorised T-T results were generated for this paper.

V. EXPERIMENTAL RESULTS

The LS-Transducer was compared to the standard T-T models for both intra and cross-domain scenarios. Ablation studies were conducted to evaluate the effectiveness of AIF, streaming joint decoding, and prediction network initialisation. Several related methods were also implemented and experimentally compared to the LS-Transducer.

A. Initial Experiments

Initial experiments were conducted on the LibriSpeech-100 h set and the ASR results are listed in Table I, in which our models showed good results on the LibriSpeech-100 h benchmark compared to various recent work. In addition, among the three standard T-T models, the Transformer-Pred T-T performed the best, showing that a strong Transformer-structured prediction network is very helpful for the neural transducer to achieve high ASR accuracy. Moreover, compared to the strong Transformer-Pred T-T, HAT [34] and factorised T-T [16] slightly degraded the intra-domain performance. However, the proposed LS-Transducer still clearly surpassed the strong Transformer-Pred T-T model with up to 16.3% relative WER reduction (WERR). Moreover, the online LS-Transducer even slightly outperformed the offline AED model, showing the advantages of the LS-Transducer, including that the prediction network performs as an explicit LM so that it can be easily initialised by a trained source-domain LM. This initialisation technique proved to be highly effective in improving ASR performance for some non-autoregressive E2E models [25], but remains a challenge for auto-regressive E2E models such as Transformer-based AED [29] and standard neural transducer [15] due to the lack of an explicit LM component.

Given the strong performance shown by the Transformer-Pred T-T, the main experiments were carried out on the LibriSpeech-960 h data focusing on the comparison between the Transformer-Pred T-T and LS-Transducer.

TABLE II
INTRA-DOMAIN WER ON LIBRISPEECH DEV/TEST SETS FOR ONLINE TRANSDUCER MODELS TRAINED FROM LIBRISPEECH-960 H

Online ASR Models	Test		Dev	
	clean	other	clean	other
ConvT-T [54]	3.5	8.3	–	–
Dynamic Encoder Transducer [55]	3.5	9.0	–	–
Parallel Encoder Transducer [56]	3.7	9.3	3.5	9.2
Transformer-Pred T-T	3.2	8.0	3.0	7.8
+LM Shallow Fusion	3.1	7.7	2.9	7.5
Proposed LS-Transducer	3.0	7.2	2.9	7.4
+LM Shallow Fusion	**2.7**	**6.8**	**2.6**	**6.7**

*Note that the results generated for this paper are not directly comparable to other published results since there are differences in both model encoder types and also in the streaming operation.

TABLE III
CROSS-DOMAIN WER RESULTS ON TED-LIUM 2 (TED2) AND AESRC2020 (AESRC) FOR ONLINE TRANSDUCER MODELS TRAINED FROM LIBRISPEECH-960 H (LS960)

Online ASR Models	LS960⇒Ted2		LS960⇒AESRC	
	Test	Dev	Dev	Test
Transformer-Pred T-T	12.7	13.1	19.0	18.7
+Target-domain LM SF	11.9	12.2	16.7	16.2
Proposed LS-Transducer	11.7	12.0	18.2	17.8
+Adapting Prediction Net	10.0	10.3	14.9	14.1
++Target-domain LM SF	**9.1**	**9.6**	**13.6**	**12.6**

SF denoted shallow fusion [30].

B. Main Experiments

Table II lists intra-domain ASR results, with E2E ASR models trained on the LibriSpeech-960 h corpus, our models yielded competitive results on the LibriSpeech-960 h benchmark compared to recent work. The LS-Transducer still outperformed the Transformer-Pred T-T in the high-resource LibriSpeech-960 h scenario, with 10% relative WERR. In addition, when the external source-domain LM was used for the E2E ASR via shallow fusion [30], the performance of both models was further improved, where the LS-Transducer still gave a 12.9% relative WERR compared to the Transformer-Pred T-T. It should be noted that the results in Table II are not directly comparable to other published results since there are many differences in both model encoder types and also in the streaming operation.

The TED-LIUM 2 and AESRC2020 dev/test sets were used to evaluate the cross-domain performance of the ASR models trained on the LibriSpeech-960 h data. As shown in Table III, the proposed LS-Transducer gave the best results on both cross-domain corpora, showing that LS-Transducer generalises well rather than overfitting to the source domain. With the prediction network adapted/fine-tuned using the target-domain text data, further improvements could be gained and surpass the

TABLE IV
ABLATION STUDIES ON THE LABEL-LEVEL ENCODER REPRESENTATION GENERATION MECHANISM: INTRA-DOMAIN WER FOR LS-TRANSDUCER TRAINED ON LIBRISPEECH-100 H OR LIBRISPEECH-960 H WITH AIF OR NORMAL CIF [18]

Online ASR Models	Test		Dev	
	clean	other	clean	other
LibriSpeech 100h				
Transformer-Pred T-T	5.1	12.0	4.9	12.0
Proposed LS-Transducer w/ AIF	**4.4**	**11.0**	**4.1**	**10.8**
Proposed LS-Transducer w/ CIF	7.0	13.3	6.5	13.2
LibriSpeech 960h				
Transformer-Pred T-T	3.2	8.0	3.0	7.8
Proposed LS-Transducer w/ AIF	**3.0**	**7.2**	**2.9**	**7.4**
Proposed LS-Transducer w/ CIF	4.6	9.0	4.3	8.7

Transformer-Pred T-T model with 21.4% and 24.6% relative WERR on TED-LIUM 2 and AESRC2020, respectively. Even when shallow fusion [30] was used for the Transformer-Pred T-T model to improve the cross-domain performance by incorporating an external target-domain LM, a performance gap of at least 10.8% relative WERR still existed compared to the LS-Transducer with adapted prediction network. In addition, the LS-Transducer could also use shallow fusion with the external target-domain LM to further boost cross-domain accuracy.

In summary, the proposed LS-Transducer not only outperforms the standard T-T models within the source domain but also exhibits greatly improved domain adaptation capabilities. This is primarily because the prediction network of the LS-Transducer works as an explicit LM, which brings advantages in utilising text-only data.

C. Ablation Studies on AIF

Ablation studies were conducted to evaluate the effectiveness of the proposed AIF mechanism. As shown in Table IV, when ASR models were trained on the LibriSpeech-100 h data, using CIF with LS-Transducer resulted in noticeably inferior performance compared to the strong Transformer-Pred T-T model, which is consistent with the conclusion about CIF in [26]. The proposed AIF gave much lower WER than CIF [18] and played an essential role in enabling the LS-Transducer to surpass the strong Transformer-Pred T-T model. This is consistent with the comparison in Section III-A that the proposed AIF has several advantages that improve performance over CIF, including no mismatch between training and decoding and enhanced robustness to inaccurate unit boundaries.

When ASR models were trained on the LibriSpeech-960 h corpus, as shown in Table IV, the LS-Transducer with CIF achieved obvious progress compared to when it was only trained on LibriSpeech-100 h data. However, it still failed to yield competitive performance compared to the Transformer-Pred T-T. Consistent with the LibriSpeech-100 h scenario, the LS-Transducer with AIF gave relative WERR between 14.9% to 34.7% compared to using CIF, thereby allowing the LS-Transducer to exceed the Transformer-Pred T-T.

Real speech examples are given in the Supplemental Materials to illustrate the operation of the AIF mechanism and compare it to CIF.

TABLE V
ABLATION STUDIES: INTRA-DOMAIN WER FOR LS-TRANSDUCER TRAINED ON LIBRISPEECH-960 H WITH OR WITHOUT THE STREAMING JOINT DECODING

Online ASR on LS960	Test		Dev	
	clean	other	clean	other
Transformer-Pred T-T	3.2	8.0	3.0	7.8
Proposed LS-Transducer				
w/ streaming joint decoding	**3.0**	**7.2**	**2.9**	**7.4**
w/o modification for [eos]	6.8	9.9	6.0	10.0
w/o streaming joint decoding	3.9	7.9	3.5	7.8

D. Ablation Studies on Streaming Joint Decoding

Ablation studies were also conducted to evaluate the proposed streaming joint decoding method. As shown in Table V, the LS-Transducer gave competitive results compared to the strong Transformer-Pred T-T model even without the streaming joint decoding. Moreover, streaming joint decoding could further yield up to 23.1% relative WERR for the LS-Transducer. This is because the online CTC prefix score can help refine the search space and eliminate irrelevant alignments. However, when the modification of the online CTC prefix score for [eos] proposed in (12) or line 2 of Algorithm 1 was not used, the performance was greatly degraded. This is consistent with what is mentioned in Section III-B. Therefore, the proposed streaming joint decoding method is simple and effective and can ensure strict synchronisation of the online CTC prefix score and the LS-Transducer predictions.

The performance of the LS-Transducer when the CTC branch is removed is explored in the Supplemental Materials.

E. Ablation Studies on Prediction Network Initialisation

In addition, considering a trained source-domain LM was used to initialise the prediction network of the LS-Transducer, ablation studies were conducted to evaluate its effectiveness for both the LS-Transducer and Transformer-Pred T-T. As shown in Table VI, pre-training the prediction network of the Transformer-Pred T-T cannot improve performance but rather harms it, consistent with the conclusion in [15]. In contrast, the prediction network initialisation is highly effective for the LS-Transducer because it performs as an explicit LM. Text-only data is normally easier to collect in large quantities, and the source-domain text data in this paper is much larger than the LibriSpeech-960h transcripts, which is why the prediction network initialisation is still effective for this high-resource LibriSpeech-960h data. Hence, the LS-Transducer provides a natural approach to utilise pre-trained LMs in ASR.

TABLE VI
INTRA-DOMAIN WER FOR TRANSFORMER-PRED T-T AND LS-TRANSDUCER TRAINED ON LIBRISPEECH-960 H WITH OR WITHOUT PREDICTION NETWORK PRE-TRAINED

Online ASR on LS960	Test		Dev	
	clean	other	clean	other
Transformer-Pred T-T	3.2	8.0	3.0	7.8
w/ pre-trained prediction network	4.5	9.6	4.2	9.5
Proposed LS-Transducer	**3.0**	**7.2**	**2.9**	**7.4**
w/o pre-trained prediction network	3.5	8.1	3.5	8.0

TABLE VII
WER ON INTRA (LS100) AND CROSS-DOMAIN (TED2 AND AESRC) TEST SETS FOR DIFFERENT MODELS TRAINED FROM LIBRISPEECH-100 H

Online ASR on LS100	LS100 Test		Ted2	AESRC
	clean	other	Test	Test
Transformer-Pred T-T Baseline	5.1	12.0	13.6	23.6
HAT [34]	5.4	12.2	13.6	23.0
Factorised T-T [16]	5.4	12.4	13.3	22.5
Proposed LS-Transducer	**4.4**	**11.0**	**11.1**	**20.7**

For cross-domain scenarios, the internal LM of HAT [34] was estimated, the vocabulary predictor of factorised T-T [16] was adapted on target-domain text, and shallow fusion was used. HAT and factorised T-T results were generated for this paper.

F. Comparison With Related Work

As a further point of comparison, the cross-domain performance of the factorised T-T [16] and HAT [34] models were compared to the LS-Transducer. The intra-domain and cross-domain results are listed in Table VII, as mentioned in Section V-A, HAT and factorised T-T performed slightly worse than the strong Transformer-Pred T-T in the intra-domain scenario. Nonetheless, leveraging their strengths in domain adaptation, such as internal LM estimation or adaptation, mitigates this gap and results in improved cross-domain performance compared to the Transformer-Pred T-T. However, the proposed LS-Transducer still significantly outperformed the HAT and factorised T-T in both intra and cross-domain scenarios with relative WERRs between 8.0% and 18.5%.

The WER improvement brought by the LS-Transducer over the HAT and factorised T-T is statistically significant at the 0.1% level according to a matched-pair sentence-segment word error statistical test [57].

VI. CONCLUSION

This paper proposes a label-synchronous neural transducer (LS-Transducer), which offers a natural solution to domain adaptation for online ASR. The LS-Transducer does not require the prediction of blank tokens and it is therefore easy to adapt the prediction network on text-only data. An Auto-regressive Integrate-and-Fire (AIF) mechanism is designed to generate a label-level encoder representation before being combined with the prediction network output while still allowing streaming. In addition, a streaming joint decoding method is proposed to refine the search space during beam search while maintaining synchronisation with the AIF. Experiments showed that the proposed LS-Transducer had superior ASR performance and effective domain adaptation capabilities, exceeding standard neural transducers with 12.9% and 24.6% relative WER reductions in intra-domain and cross-domain scenarios respectively.

REFERENCES

[1] G. E. Dahl, D. Yu, L. Deng, and A. Acero, "Context-dependent pre-trained deep neural networks for large-vocabulary speech recognition," *IEEE Trans. Audio Speech Lang. Process.*, vol. 20, no. 1, pp. 30–42, Jan. 2012.

[2] G. E. Hinton et al., "Deep neural networks for acoustic modeling in speech recognition," *IEEE Signal Process. Mag.*, vol. 29, no. 6, pp. 82–97, Nov. 2012.

[3] K. Veselý, A. Ghoshal, L. Burget, and D. Povey, "Sequence-discriminative training of deep neural networks," in *Proc. Interspeech Commun. Assoc.*, 2013, pp. 2345–2349.

[4] S. J. Young, J. J. Odell, and P. C. Woodland, "Tree-based state tying for high accuracy modelling," in *Proc. Workshop Hum. Lang. Technol.*, 1994, pp. 307–312.

[5] D. Wang, X. Wang, and S. Lv, "An overview of end-to-end automatic speech recognition," *Symmetry*, vol. 11, 2019, Art. no. 1018.

[6] A. Graves, A. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, 2013, pp. 4560–4564.

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

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

[9] W. Chan, N. Jaitly, Q. V. 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.

[10] A. Vaswani et al., "Attention is all you need," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2017, pp. 6000–6010.

[11] J. Li, "Recent advances in end-to-end automatic speech recognition," *APSIPA Trans. Signal Inf. Process.*, vol. 11, no. 1, 2022.

[12] 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.

[13] J. Li, Y. Wu, Y. Gaur, C. Wang, R. Zhao, and S. Liu, "On the comparison of popular end-to-end models for large scale speech recognition," in *Proc. Interspeech Commun. Assoc.*, 2020, pp. 1–5.

[14] Q. Li, C. Zhang, and P. C. Woodland, "Combining hybrid DNN-HMM ASR systems with attention-based models using lattice rescoring," *Speech Commun.*, vol. 147, pp. 12–21, 2023.

[15] 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.

[16] 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.

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

[18] 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.

[19] 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.

[20] Y. Higuchi, B. Yan, S. Arora, T. Ogawa, T. Kobayashi, and S. Watanabe, "BERT meets CTC: New formulation of end-to-end speech recognition with pre-trained masked language model," in *Proc. Conf. Empir. Methods Natural Lang. Process.*, 2022, pp. 5486–5503.

[21] C. Wang et al., "Low latency end-to-end streaming speech recognition with a scout network," in *Proc. Interspeech Commun. Assoc.*, 2020, pp. 2112–2116.
[22] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.
[23] A. Gulati et al., "Conformer: Convolution-augmented transformer for speech recognition," in *Proc. Interspeech Commun. Assoc.*, 2020, pp. 5036–5040.
[24] 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.
[25] C. Yi, S. Zhou, and B. Xu, "Efficiently fusing pretrained acoustic and linguistic encoders for low-resource speech recognition," *IEEE Signal Process. Lett.*, vol. 28, pp. 788–792, 2021.
[26] Y. Higuchi et al., "A comparative study on non-autoregressive modelings for speech-to-text generation," in *Proc. IEEE Autom. Speech Recognit. Understanding Workshop*, 2021, pp. 47–54.
[27] L. Yao, J. Song, R. Xu, Y. Yang, Z. Chen, and Y. Deng, "WaBERT: A low-resource end-to-end model for spoken language understanding and speech-to-bert alignment," 2022, *arXiv:2204.10461*.
[28] P. Gage, "A new algorithm for data compression," *C Users J.*, vol. 12, no. 02, pp. 23–38, 1994.
[29] K. Deng et al., "Improving CTC-based speech recognition via knowledge transferring from pre-trained language models," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, 2022, pp. 8517–8521.
[30] J. K. Chorowski, D. Bahdanau, D. Serdyuk, K. Cho, and Y. Bengio, "Attention-based models for speech recognition," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2015, pp. 577–585.
[31] A. Sriram, H. Jun, S. Satheesh, and A. Coates, "Cold fusion: Training Seq2Seq models together with language models," in *Proc. Interspeech Commun. Assoc.*, 2018, pp. 387–391.
[32] Z. Meng et al., "Internal language model training for domain-adaptive end-to-end speech recognition," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, 2021, pp. 7338–7342.
[33] C. Choudhury, A. Gandhe, X. Ding, and I. Bulyko, "A likelihood ratio based domain adaptation method for E2E models," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, 2022, pp. 6762–6766.
[34] 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.
[35] 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 Commun. Assoc.*, 2021, pp. 2856–2860.
[36] Z. Meng et al., "Internal language model estimation for domain-adaptive end-to-end speech recognition," in *Proc. IEEE Spoken Lang. Technol. Workshop*, 2021, pp. 243–250.
[37] 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.
[38] E. Tsunoo, Y. Kashiwagi, C. P. Narisetty, and S. Watanabe, "Residual language model for end-to-end speech recognition," in *Proc. Interspeech Commun. Assoc.*, 2022, pp. 3899–3903.
[39] Z. Meng et al., "Internal language model adaptation with text-only data for end-to-end speech recognition," in *Proc. Interspeech Commun. Assoc.*, 2022, pp. 2608–2612.
[40] Z. Meng et al., "Modular hybrid autoregressive transducer," in *Proc. IEEE Spoken Lang. Technol. Workshop*, 2023, pp. 197–204.
[41] Y. Deng et al., "Improving RNN-T for domain scaling using semi-supervised training with neural TTS," in *Proc. Interspeech Commun. Assoc.*, 2021, pp. 751–755.
[42] X. Zheng, Y. Liu, D. Gunceler, and D. Willett, "Using synthetic audio to improve the recognition of out-of-vocabulary words in end-to-end ASR systems," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, 2021, pp. 5674–5678.
[43] F. Boyer, Y. Shinohara, T. Ishii, H. Inaguma, and S. Watanabe, "A study of transducer based end-to-end ASR with ESPnet: Architecture, auxiliary loss and decoding strategies," in *Proc. IEEE Autom. Speech Recognit. Understanding Workshop*, 2021, pp. 16–23.
[44] 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.
[45] 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.
[46] 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 Commun. Assoc.*, 2019, pp. 2623–2627.
[47] A. Rousseau, P. Deléglise, and Y. Estève, "Enhancing the TED-LIUM corpus with selected data for language modeling and more TED talks," in *Proc. 9th Int. Conf. Lang. Resour. Eval.*, 2014, pp. 3935–3939.
[48] X. Shi et al., "The accented English speech recognition challenge 2020: Open datasets, tracks, baselines, results and methods," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, 2021, pp. 6918–6922.
[49] S. Watanabe et al., "ESPnet: End-to-end speech processing toolkit," in *Proc. Interspeech Commun. Assoc.*, 2018, pp. 2207–2211.
[50] W.-N. Hsu et al., "Robust wav2vec 2.0: Analyzing domain shift in self-supervised pre-training," in *Proc. Interspeech Commun. Assoc.*, 2021, pp. 721–725.
[51] R. Zhao, J. Xue, P. Parthasarathy, V. Miljanic, and J. Li, "Fast and accurate factorized neural transducer for text adaption of end-to-end speech recognition models," in *Proc. Int. Conf. Acoust., Speech, Signal Process. (ICASSP)*, 2023, pp. 1–5.
[52] X. Yang, Q. Li, and P. C. Woodland, "Knowledge distillation for neural transducers from large self-supervised pre-trained models," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, 2022, pp. 8527–8531.
[53] D. Albesano, J. Andrés-Ferrer, N. Ferri, and P. Zhan, "On the prediction network architecture in RNN-T for ASR," in *Proc. Interspeech Commun. Assoc.*, 2022, pp. 2093–2097.
[54] W. Huang, W. Hu, Y. T. Yeung, and X. Chen, "Conv-transformer transducer: Low latency, low frame rate, streamable end-to-end speech recognition," in *Proc. Interspeech Commun. Assoc.*, 2020, pp. 5001–5005.
[55] Y. Shi et al., "Dynamic encoder transducer: A flexible solution for trading off accuracy for latency," in *Proc. Interspeech Commun. Assoc.*, 2021, pp. 2042–2046.
[56] Y. Sudo, S. Muhammad, Y. Peng, and S. Watanabe, "Time-synchronous one-pass beam search for parallel online and offline transducers with dynamic block training," in *Proc. Interspeech Commun. Assoc.*, 2023, pp. 4479–4483.
[57] D. S. Pallet, W. M. Fisher, and J. G. Fiscus, "Tools for the analysis of benchmark speech recognition tests," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, 1990, pp. 97–100.

Keqi Deng (Graduate Student Member, IEEE) received the B.S. degree from Nanjing University, Nanjing, China. He is currently working toward the Ph.D. degree with the University of Cambridge, Cambridge, U.K., supervised by Prof. Phil Woodland. His research interests include speech recognition, speech translation, and large language models. He has authored or coauthored more than ten first-author speech and language processing papers and serves as a regular reviewer for IEEE/ACM Transactions on Audio Speech and Language Processing.

Philip C. Woodland (Fellow, IEEE) is currently a Professor of information engineering with the Department of Engineering, University of Cambridge, Cambridge, U.K., where he is the Head of the Machine Intelligence Laboratory and a Professorial Fellow of Peterhouse. After working with British Telecom Research Labs for three years, he returned to a Lectureship at Cambridge, in 1989, and became a Full Professor in 2002. He has authored or coauthored more than 300 papers in the area of speech and language technology with a focus on speech recognition systems and related areas. He was the recipient of number of best paper awards including for work on speaker adaptation and discriminative training. He was one of the original coauthors of the HTK toolkit and then continued to play a major role in its development. He was a Member of the Editorial Board of Computer Speech and Language durng 1994–2009 and is currently a Member of the Editorial Board Member of Speech Communication. He was a Member of the Speech Technical Committee of the IEEE Signal Processing Society from 1999 to 2003. He is a Fellow of the International Speech Communication Association and the Royal Academy of Engineering.