INTERSPEECH 2023
20-24 August 2023, Dublin, Ireland

Bypass Temporal Classification: Weakly Supervised Automatic Speech Recognition with Imperfect Transcripts

Dongji Gao[1], *Matthew Wiesner*[2], *Hainan Xu*[3], *Leibny Paola Garcia*[1,2], *Daniel Povey*[4], *Sanjeev Khudanpur*[1,2]

[1]CLSP & [2]HLTCOE, Johns Hopkins University, USA
[3]NVIDIA, USA
[4]Xiaomi Corp., China

{dgao5,wiesner,lgarci27,khudanpur}@jhu.edu, hainanx@nvidia.com, dpovey@gmail.com

Abstract

This paper presents a novel algorithm for building an automatic speech recognition (ASR) model with imperfect training data. Imperfectly transcribed speech is a prevalent issue in human-annotated speech corpora, which degrades the performance of ASR models. To address this problem, we propose Bypass Temporal Classification (BTC) as an expansion of the Connectionist Temporal Classification (CTC) criterion. BTC explicitly encodes the uncertainties associated with transcripts during training. This is accomplished by enhancing the flexibility of the training graph, which is implemented as a weighted finite-state transducer (WFST) composition. The proposed algorithm improves the robustness and accuracy of ASR systems, particularly when working with imprecisely transcribed speech corpora. Our implementation will be open-sourced.

Index Terms: weakly supervised learning, automatic speech recognition, weighted finite-state transducer, CTC

1. Introduction

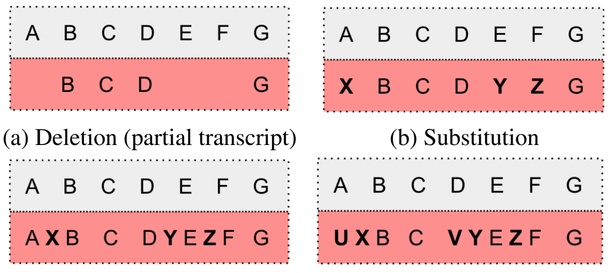
The quality and quantity of data are critical for training a successful ASR system. State-of-the-art end-to-end (E2E) ASR models rely heavily on large quantities of accurately annotated speech data. However, transcription of speech corpora by human annotators, unlike read speech, is prone to errors. Removing low-quality data can significantly reduce the amount of available data. This is especially problematic for low-resource languages and dialects, where data is scarce. Therefore, it is imperative to develop methods that address imperfectly labeled data for speech processing and other sequence classification tasks, to maximize the utilization of existing data.

Previous studies have focused on extracting reliable parallel speech and text from noisy data sources, such as television news broadcasts and their closed captions. While closed captions provide useful information, they are often not completely accurate, usually with a word error rate (WER) of between 10% and 20% compared to a careful verbatim transcript [1]. To address this issue, two-step solutions have been developed:

1. Use an extra ASR model to generate hypotheses for each utterance, which is then aligned with the corresponding closed caption to identify the matching speech fragment, either at word level [2, 3, 4] or segment level [5, 6, 7, 8, 9].
2. Add the resulting speech and caption pairs to the training data to improve the ASR model.

This process can be repeated iteratively until adding data does not improve the ASR performance. Although these methods have provided a partial solution to the problem, their main limitation is that they are not self-contained, meaning that external ASR models are required for alignment. Furthermore, the iterative training approach can be time-consuming, which may limit its practicality in some scenarios.

This work was partially supported by the National Science Foundation CCRI program via Grant No 2120435.



(a) Deletion (partial transcript) (b) Substitution (c) Insertion (d) Substitution and insertion

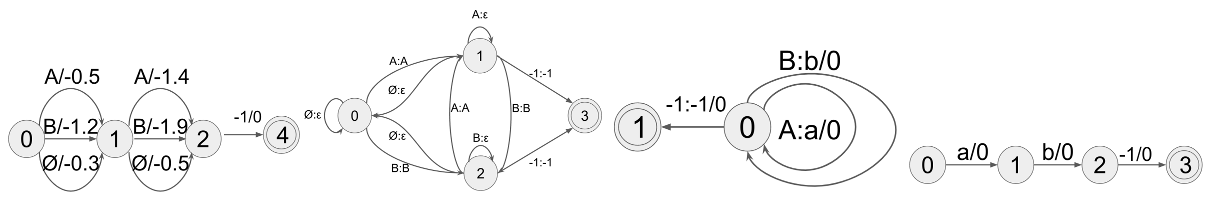
Figure 1: *Examples of error in the transcript. The grey box is the exact text and the red box is the imperfect text. Inaccurate words are marked in bold.*

Alternative efforts have been proposed to directly train a model with inaccurate labels [10], which is commonly referred to as *weakly supervised learning*. Prior research in this realm of methods has primarily concentrated on tasks involving partially labeled classification [11, 12]. Specifically, in ASR tasks, each spoken utterance only has partial transcriptions available (The term "deletion" is used in this case, as shown in Fig. 1(a)). To address this challenge, [13] proposes a novel variant of the CTC [14] criterion known as W-CTC. This approach enables the handling of sequences that are missing both beginning and ending labels. This work is extended by the Star Temporal Classification (STC) [15] to tackle the more general problem of partial labeling. The proposed STC algorithm incorporates WFSTs to explicitly manage an arbitrary number of missing labels, regardless of their location in the sequence. This study suggests that in situations where up to 70% of labels are missing, the performance of STC can approach that of a supervised baseline.

Alongside this direction, we focus on the other two types of errors, i.e., substitution and insertion in transcripts, as illustrated in Fig. 1(b), (c), and (d). We propose an extended CTC criterion, termed Bypass Temporal Classification (BTC), to handle substitution and insertion errors during training explicitly. Our proposed approach can be effectively implemented under WFSTs' framework. We implemented WFST in the k2 toolkit[1], which supports automatic differentiation for WFST to enable seamless integration between WFST and neural models. Notably, all our WFST operations can be executed on GPU, thereby significantly accelerating the training process (STC re-

[1]`https://github.com/k2-fsa/k2`

10.21437/Interspeech.2023-2258



(a) Emission WFST (E) (b) Alignment WFST (H) (c) Lexicon WFST (L) (d) Grammar WFST (G)

Figure 2: *WFST topology of transcript "a b" with lexicon* $\{$*a:A, b:B*$\}$. *A state with* 0 *is the start state. The arc with* -1 *is a special arc in k2 pointing to the final state (double circle). Each arc is associated with an input symbol, an output symbol (delimited by a colon), and weight after the slash. Arc with a single symbol indicates identical input and output. Arc without slash indicates zero weight.*

quires switching between GPU and CPU[16]). We illustrate that an ASR model can be trained from scratch using BTC with imperfect transcripts, containing up to $50-70\%$ substitution or insertion errors. The model achieves acceptable ASR performance as measured by either phone error rate (PER) or word error rate (WER).

2. Preliminaries

2.1. CTC

The CTC criterion was proposed for sequence labeling and is commonly used in ASR. Given an acoustic feature sequence $\mathbf{x} = [x_1, \dots, x_T]$ of length T, and its corresponding transcript $\mathbf{l} = [l_1, \dots, l_U] \in \mathcal{V}^{1\times U}$ of length U, where $\mathcal{V}$ is the vocabulary, and with the constraint $U \leq T$, CTC loss is defined as

$$L_{\text{ctc}} = -\log P(\mathbf{l}|\mathbf{x}). \tag{1}$$

By introducing a blank token $\oslash$ and the framewise alignment sequence $\boldsymbol{\pi} = [\pi_1, \dots, \pi_T]$, where $\pi_t \in \mathcal{V} \cup \oslash$, the posterior distribution $P(\mathbf{l}|\mathbf{x})$ can be factorized as

$$\begin{aligned} P(\mathbf{l}|\mathbf{x}) &= \sum_{\boldsymbol{\pi}\in\mathcal{B}^{-1}(\mathbf{l})} P(\mathbf{l},\boldsymbol{\pi}|\mathbf{x}) && (2) \\ &= \sum_{\boldsymbol{\pi}\in\mathcal{B}^{-1}(\mathbf{l})} P(\mathbf{l}|\boldsymbol{\pi},\mathbf{x})P(\boldsymbol{\pi}|\mathbf{x}) && (3) \\ &= \sum_{\boldsymbol{\pi}\in\mathcal{B}^{-1}(\mathbf{l})} P(\boldsymbol{\pi}|\mathbf{x}), && (4) \end{aligned}$$

where $\mathcal{B}$ maps sequences from $\boldsymbol{\pi}$ to $\mathbf{l}$ by removing $\oslash$ and adjacent repetitions, which is why $P(\mathbf{l}|\boldsymbol{\pi},\mathbf{x}) \equiv 1$ for every $\boldsymbol{\pi} \in \mathcal{B}^{-1}(\mathbf{l})$. $P(\boldsymbol{\pi}|\mathbf{x})$ may be further factorized as

$$\begin{aligned} P(\boldsymbol{\pi}|\mathbf{x}) &= \prod_{t=1}^{T} P(\pi_t|\pi_1,\dots,\pi_{t-1},\mathbf{x}) \\ &= \prod_{t=1}^{T} P(\pi_t|\mathbf{x}), \end{aligned} \tag{5}$$

by *assuming* that $\boldsymbol{\pi}$ is *conditionally i.i.d.* given the sequence $\mathbf{x}$.

2.2. WFST

The marginalization over token sequences represented by $\mathcal{B}^{-1}(\mathbf{l})$ can be efficiently computed using WFSTs. A WFST maps input symbol sequences to output symbol sequences and assigns a weight to each transition in the transducer. The weight of the mapping is often set to be the conditional probability of the output sequence given the input. WFSTs have found extensive application in the field of ASR to model the probability of decoding unit sequences [17, 18, 19]. This can be attributed to the compactness of its representation of the model architecture [20, 21].

Two WFSTs can be composed to cascade mapping operations. Given a WFST F_1 that maps $\mathbf{a}$ to $\mathbf{b}$ with weight w_1, and a WFST F_2 that maps $\mathbf{b}$ to $\mathbf{c}$ with weight w_2. The composed WFST, denoted $F_1 \circ F_2$, maps $\mathbf{a}$ to $\mathbf{c}$. Its weight can be either $\max(w_1, w_2)$ (tropical semiring) or log-sum-exp(w_1, w_2) (log semiring).

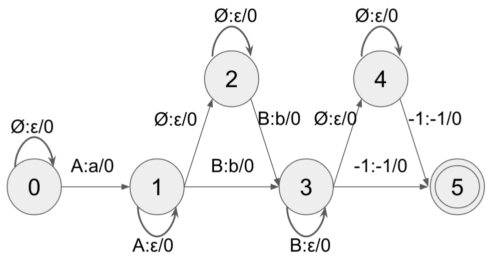


Figure 3: *CTC graph of "a b".* $\oslash$ *is the blank token and* ϵ *represents no output symbol in WFST transitions. The reader can refer to Fig. 2 for details of WFST representation.*

The composition of the alignment WFST (H), with a WFST representation of a lexicon (L) and transcript ($G(\mathbf{l})$) and inverting the result, returns a transducer that maps from transcripts to a graph that represents the space of all possible CTC paths associated with the transcript. Marginalization is enabled via the application of the forward algorithm on the resulting lattice of paths. We show that Eq. 4 may be represented using a composition of WFSTs:

$$P(\mathbf{l}|\boldsymbol{\pi}) \quad = \sum_{\boldsymbol{\pi}\in(H\circ L\circ G(\mathbf{l}))^{-1}} \underbrace{P(\boldsymbol{\pi}|\mathbf{x})}_{E}, \tag{6}$$

where the superscript -1 denotes WFST inversion;

- E is the emission WFSA representing $P(\boldsymbol{\pi}|\mathbf{x})$, as shown in Fig. 2(a). For state t, the weights on its arcs are the log-probabilities on the set of *decoding units* (e.g., characters or phones or sub-words or words, and $\oslash$) at time frame t;
- H is the CTC alignment WFST. It removes $\oslash$ and adjacent repeated decoding units;
- L is the lexicon. It maps sequences of decoding units to words. It is discarded if the decoding units are words;
- $G(\mathbf{l})$ is the grammar or language model of $\mathbf{l}$, assumed represent-able as a WFST, as shown in Fig. 2(d).

Therefore, $\log P(\mathbf{l}|\mathbf{x})$ equals to the total weight of $E \circ H \circ L \circ G(\mathbf{l})$ under the log-semiring. We refer to $H \circ L \circ G(\mathbf{l})$ as *the graph* of $\mathbf{l}$, and illustrate it with an example in Fig. 3.

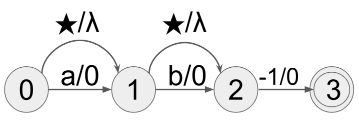


Figure 4: *BTC Grammar WFST (G) of transcript “a b”. Each label has parallel arcs added with input symbol ⋆ and weight λ.*

3. Method

In the training process of CTC models, for each training example $(\mathbf{x}, \mathbf{l})$, a graph is constructed based on $\mathbf{l}$. The presence of substitution and insertion errors in $\mathbf{l}$ can lead to erroneous graph construction, causing misalignment between certain segments of $\mathbf{x}$ and incorrect tokens. The consequential misalignment can adversely affect the ASR model through back-propagation.

In the CTC framework, $P(\pi_t|\boldsymbol{x})$ represents the underlying model's estimate of “which decoding unit best represents this frame.” Setting $\pi_t = \oslash$ represents a continuation-without-explicit-repetition of the most recent non-blank symbol in $\boldsymbol{\pi}$, allowing for *uncertainty in the temporal span* of the words in $\mathbf{l}$. But there is no uncertainty in $\mathbf{l} = \mathcal{B}(\boldsymbol{\pi})$. In BTC, we introduce a special “wildcard” token ⋆ to model *uncertainty in the transcript* $\mathbf{l}$. The ⋆ arcs are added to $G(\mathbf{l})$ in parallel to words in the transcript, as shown in Fig. 4, and to L as an identity mapping (i.e. with no subword decomposition). The alignment transducer H treats ⋆ as a standard decoding unit, resulting in a graph of the kind shown in Fig. 5. This topology is capable of modeling insertions and substitution errors in $\mathbf{l}$:

1. Substitution: since there is a ⋆ arc parallel to arc of an incorrect word l, $P(\pi_t|\boldsymbol{x})$ can bypass l and assign the acoustics of the (unknowable) correct word to ⋆ with high probability, thereby avoid corruption of its internal representation of l: ⋆ acts like a garbage collector.
2. Insertion: when $\mathbf{l}$ contains more words than were spoken, $P(\pi_t|\boldsymbol{x})$ can again bypass an inserted word l, assigning a *minimally necessary* number of frames to the ⋆ arc in parallel to l, and continue. This again avoids corrupting the internal representation of l, albeit by “stealing” a minimal number of frames from the adjacent words to assign to ⋆.

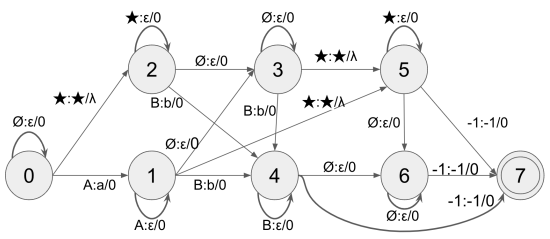


Figure 5: *Topology of the BTC graph for “a b” after composition with the CTC alignment transducer H.*

Training naively on the BTC grammar with random initialization, however, leads to a degenerate solution: map all acoustic features to ⋆ with high probability. In order to avoid this solution, we introduce a penalty, i.e. a negative weight denoted by λ, to all ⋆ arcs. We then investigate a scheduling algorithm that adjusts this penalty during training: it is initially set to a large value to encourage the model to start learning the correspondence between acoustics and the imperfect transcripts. As the training progresses, and the models understanding of this relationship begins to take form, we gradually decrease the penalty. Specifically, for the i-th epoch, the penalty is set as

$$\lambda_i = \beta * \tau^i, \tag{7}$$

where β is the initial (high) penalty that decays geometrically by a factor $\tau \in (0, 1)$. We determine β and τ empirically.

4. Experimental Setup

4.1. Datasets

We use two standard datasets for our experiments.

TIMIT [22] is comprised of 6300 sentences, totaling 5.4 hours of recorded read speech. Each sentence has been phonetically transcribed. We adhere to the standard training, development, and test partitions, consisting of 3696, 400, and 192 sentences, respectively.

LibriSpeech [23] is comprised of 960 hours of read speech for training, further divided into three subsets of 100, 360 and 500 hours of clean speech and “other” speech. Two development subsets and two test subsets, each consisting of 5 hours of speech, are also included in the data set.

4.2. Imperfect transcripts generation

In our study, we generate synthetically erroneous transcripts from the provided transcripts, which we assume to be perfect, for three scenarios.

- For substitution, each token in the transcript is replaced by a random token with $p_{\text{sub}} \in [0.1, 0.3, 0.5, 0.7]$.
- For insertion, a random token is inserted with $p_{\text{ins}} \in [0.1, 0.3, 0.5, 0.7]$ between any two tokens in the transcript.
- For substitution+insertion, insertion is followed by substitution with $p_{\text{sub}} \in [0.05, 0.15, 0.25, 0.35]$ and $p_{\text{ins}} \in [0.05, 0.15, 0.25, 0.35]$.

Different choices of p_{sub} and p_{ins} are studied and compared.

4.3. Acoustic feature

We utilize a pre-trained wav2vec 2.0 model [24] (wav2vec2-base) to extract 768-dimensional acoustic features. This self-supervised model contains 12 transformer blocks with 8 attention heads [25]. The acoustic features are extracted using the S3PRL toolkit.

4.4. Model architecture

We investigate two distinct model architectures utilizing the BTC criterion:

TDNN-LSTM [26, 27]: It combines time-delay neural networks (TDNN) and long short-term memory(LSTM) [28] networks. The model consists of 3 TDNN layers followed by the LSTM layer, and a linear decoder layer. This hybrid architecture can effectively capture both local and global dependencies in speech signals.

Conformer [29]: It employs a combination of convolutional neural networks (CNNs) [30] and self-attention mechanisms to model speech signals effectively. The Conformer encoder in our study consists of 12 layers, with each layer comprising a conformer block that includes a convolution module stacked after the self-attention module. The decoder is simply a linear layer.

5. Results

5.1. TIMIT

We first evaluate BTC on the TIMIT dataset to gain insight into BTC as well as tuning hyper-parameters, specifically the initial

Table 1: *WER (%) on LibriSpeech test-clean and test-other dataset. We compared CTC (highlighted in grey) and BTC in three scenarios: substitution-only, insertion-only, and substitution and insertion. We measure WER with and without LM (delimited by a slash). "-" indicates that the model does not converge.*

Error	Criterion	$p_{\text{sub}}, p_{\text{ins}}, p_{\text{sub+ins}}$							
		0.1		0.3		0.5		0.7	
		clean	other	clean	other	clean	other	clean	other
sub	CTC	6.9/15.4	15.7/29.2	12.9/20.4	24.5/36.5	36.1/43.2	54.9/60.5	-/-	-/-
	BTC	6.1/14.7	14.5/29.0	6.6/17.5	15.6/33.3	7.4/19.8	18.0/36.5	-/-	-/-
ins	CTC	6.9/16.6	15.8/31.0	19.5/29.3	30.0/44.3	-/-	-/-	-/-	-/-
	BTC	5.5/12.0	14.1/24.0	5.6/12.1	14.1/24.1	5.6/12.1	14.2/24.4	6.0/12.7	14.8/24.6
sub+ins	CTC	7.3/17.3	16.2/31.9	19.5/25.1	31.9/41.2	45.6/45.4	59.94/61.2	-/-	-/-
	BTC	5.8/13.9	13.8/27.8	6.2/15.6	14.1/29.8	6.6/16.2	14.8/30.8	6.9/16.9	15.6/32.3

penalty and penalty decay factor. The TDNN-LSTM model is utilized, with phone as the decoding unit.

Table 2: *PER (%) on TIMIT test set with substitution error. CTC is highlighted in grey.*

Error	Criterion	$p_{\text{sub}}, p_{\text{ins}}, p_{\text{sub+ins}}$			
		0.1	0.3	0.5	0.7
sub	CTC	28.3	40.9	49.2	63.1
	BTC	13.1	16.8	17.2	21.4
ins	CTC	16.1	18.1	28.8	32.8
	BTC	13.6	14.2	14.3	14.7
sub+ins	CTC	20.3	27.7	31.2	42.6
	BTC	13.4	14.0	15.9	21.4

We establish a CTC baseline and compare the phone error rate (PER) between BTC and CTC. Our findings, presented in Table 2, demonstrate that CTC's performance degrades as the error rate increases. Even with a relatively low mistranscription rate of 10%, the PER increases from 13.51 to 28.33, indicating a near-doubling of the error rate. The PER can reach as high as 63.13 under more severe mistranscription conditions (70% substitution error rate). In contrast, our evaluation reveals that the BTC shows high robustness across varying error scenarios. Even when faced with a challenging scenario involving both substitution and insertion errors, BTC achieved a phone error rate (PER) of approximately 20, despite 70% of transcripts being incorrect.

5.2. LibriSpeech

For LibriSpeech, we use clean 100 hours set as the training set. As mentioned in Section 2.2, the use of a lexicon WFST L converts decoding units to words, thereby enabling the measurement of WER.

5.2.1. Model and decoding unit selection

Given the similarity in trend between substitution and insertion, we focus on comparing TDNN-LSTM and Conformer models in the substitution case. Phones are utilized as the decoding unit, and the findings presented in Table 3 demonstrate the consistent superiority of the Conformer model over TDNN-LSTM in terms of word error rate (WER) across different substitution error rates. So we choose the Conformer model for the following experiments.

We then compare the effectiveness of BPE subwords and phones for decoding using Conformer architecture. We conduct the analysis of phones against BPE with vocabulary sizes of 100 and 500. Our results in Table 4 indicate that phones outperform BPE in terms of performance and robustness. Notably, BPE fails to converge at a substitution error rate of 0.5, even with a comparable vocabulary size of phones (100 versus 71). This is attributed to the ability of phones to capture acoustic feature patterns.

Table 3: *WER (%) of TDNN-LSTM and Conformer with substitution error. "-" indicates that the model does not converge.*

Architecture	p_{sub}			
	0.1	0.3	0.5	0.7
TDNN-LSTM	6.8	7.1	8.6	-
Conformer	6.1	6.6	7.4	-

Table 4: *WER (%) of phone and BPE with substitution error. "-" indicates that the model does not converge.*

Unit	Vocab size	p_{sub}			
		0.1	0.3	0.5	0.7
BPE	500	6.5	7.1	-	-
	100	6.3	6.7	-	-
phone	71	6.1	6.6	7.4	-

5.2.2. Results

We show the result with and without integrating a tri-gram language model (LM) in Table 1. The performance of BTC is consistent with that observed in the TIMIT dataset. The ASR model failed to converge when subject to a substitution error rate of 70%. Other than that, the system trained using BTC still remains satisfactory, with minor degradation. Moreover, for error-free transcripts, BTC does not hurt the performance compared with CTC: 5.5 vs. 5.6 on test-clean and 14.4 vs. 14.5 on test-other.

6. Conclusion and future work

This research introduces BTC as a promising approach for weakly supervised ASR. BTC, a variant of the CTC algorithm, enables ASR model training from scratch using an imperfect transcript that contains substitution and insertion errors. The implementation of the algorithm is efficient within the WFST framework, and all operations can be fully performed on GPU to expedite the training process. Our experiments on the TIMIT and LibriSpeech datasets demonstrate that the BTC approach can effectively train an ASR model without much degradation occurring when 50% to 70% of the transcripts contain errors.

Moving forward, our goal is to integrate BTC with STC to address all three types of errors (deletion, substitution, and insertion) within a unified framework for weakly supervised tasks.

7. References

[1] P. Placeway and J. D. Lafferty, "Cheating with imperfect transcripts," in *ICSLP*, 1996.

[2] M. Witbrock and A. Hauptmann, "Improving acoustic models by watching television," *Tech. Rep. CMUCS-98-110*, 1998.

[3] L. Nguyen and B. Xiang, "Light supervision in acoustic model training," in *ICASSP*, 2004.

[4] J. Driesen and S. Renals, "Lightly supervised automatic subtitling of weather forecasts," in *ASRU*, 2013.

[5] N. Braunschweiler, M. J. Gales, and S. Buchholz, "Lightly supervised recognition for automatic alignment of large coherent speech recordings," in *INTERSPEECH*, 2010.

[6] Y. Long, M. J. Gales, P. Lanchantin, X. Liu, M. S. Seigel, and P. C. Woodland, "Improving lightly supervised training for broadcast transcription," in *INTERSPEECH*, 2013.

[7] P. Lanchantin, M. J. Gales, P. Karanasou, X. Liu, Y. Qian, L. Wang, P. C. Woodland, and C. Zhang, "Selection of multi-genre broadcast data for the training of automatic speech recognition systems." in *INTERSPEECH*, 2016.

[8] M. Wiesner, M. Sarma, A. Arora, D. Raj, D. Gao, R. Huang, S. Preet, M. Johnson, Z. Iqbal, N. Goel *et al.*, "Training hybrid models on noisy transliterated transcripts for code-switched speech recognition." in *INTERSPEECH*, 2021.

[9] A. Radford, J. W. Kim, T. Xu, G. Brockman, C. McLeavey, and I. Sutskever, "Robust speech recognition via large-scale weak supervision," *ArXiv*, 2022.

[10] D. Paul, M. Singh, M. A. Hedderich, and D. Klakow, "Handling noisy labels for robustly learning from self-training data for low-resource sequence labeling," in *NAACL*, 2019.

[11] T. Cour, B. Sapp, and B. Taskar, "Learning from partial labels," *The Journal of Machine Learning Research*, 2011.

[12] L. Liu and T. Dietterich, "Learnability of the superset label learning problem," in *ICML*, 2014.

[13] X. Cai, J. Yuan, Y. Bian *et al.*, "W-ctc: a connectionist temporal classification loss with wild cards," in *ICLR*, 2022.

[14] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, "Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks," in *ICML*, 2006.

[15] V. Pratap, A. Hannun, G. Synnaeve *et al.*, "Star temporal classification: Sequence classification with partially labeled data," *NeurIPS*, 2022.

[16] A. Hannun, V. Pratap, J. Kahn, and W.-N. Hsu, "Differentiable weighted finite-state transducers," *arXiv preprint arXiv:2010.01003*, 2020.

[17] M. Mohri, F. Pereira, and M. Riley, "Weighted finite-state transducers in speech recognition," *CSL*, 2002.

[18] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. K. Goel, M. Hannemann, P. Motlícek, Y. Qian, P. Schwarz, J. Silovský, G. Stemmer, and K. Veselý, "The kaldi speech recognition toolkit," in *ASRU*, 2011.

[19] D. Povey, M. Hannemann, G. Boulianne, L. Burget, A. Ghoshal, M. Janda, M. Karafiát, S. Kombrink, P. Motlícek, Y. Qian, K. Riedhammer, K. Veselý, and N. T. Vu, "Generating exact lattices in the wfst framework," in *ICASSP*, 2012.

[20] Y. Miao, M. Gowayyed, and F. Metze, "Eesen: End-to-end speech recognition using deep rnn models and wfst-based decoding," in *ASRU*, 2015.

[21] A. Laptev, S. Majumdar, and B. Ginsburg, "Ctc variations through new wfst topologies," *INTERSPEECH*, 2022.

[22] J. Garofolo, L. Lamel, W. Fisher *et al.*, "Timit acoustic-phonetic continuous speech corpus ldc93s1," *Linguistic Data Consortium*, 1993.

[23] V. Panayotov, G. Chen, D. Povey *et al.*, "Librispeech: an ASR corpus based on public domain audio books," in *ICASSP*, 2015.

[24] A. Baevski, Y. Zhou, A. Mohamed *et al.*, "wav2vec 2.0: A framework for self-supervised learning of speech representations," in *NeurIPS*, 2020.

[25] A. Vaswani, N. Shazeer, N. Parmar *et al.*, "Attention is all you need," *NeurIPS*, 2017.

[26] H. Sak, A. W. Senior, and F. Beaufays, "Long short-term memory recurrent neural network architectures for large scale acoustic modeling," in *INTERSPEECH*, 2014.

[27] V. Peddinti, Y. Wang, D. Povey, and S. Khudanpur, "Low latency acoustic modeling using temporal convolution and lstms," *IEEE Signal Processing Letters*, 2018.

[28] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, 1997.

[29] A. Gulati, J. Qin, C. Chiu *et al.*, "Conformer: Convolution-augmented transformer for speech recognition," *INTERSPEECH*, 2020.

[30] Y. LeCun and Y. Bengio, "Convolutional networks for images, speech, and time series," 1998.