EXPLORING DOMINANT PATHS IN CTC-LIKE ASR MODELS: UNRAVELING THE EFFECTIVENESS OF VITERBI DECODING

Zeyu Zhao, Peter Bell, Ondřej Klejch

Centre for Speech Technology Research, University of Edinburgh, UK

ABSTRACT

Connectionist Temporal Classification (CTC) has emerged as a fundamental technique in Automatic Speech Recognition (ASR), renowned for its ability to marginalise all possible alignments between input and target sequences. This study reevaluates the traditional dependency on prefix beam search, which typically considers multiple alignments for each hypothesis, in CTC-based models. Our findings initially indicate the absence of a single dominant path in the lattices. However, we discover that the Viterbi decoder can effectively emulate the performance of a prefix beam search, as the hypotheses it identifies tend to be dominant in the lattices. This leads us to propose that an effective CTC-like model should not only aim for high accuracy but also align the optimal hypothesis with the most probable path, thereby enhancing the robustness of Viterbi decoding. Furthermore, our insights extend to a variety of topologies, demonstrating their applicability within a more comprehensive ASR framework.

***Index Terms*—** ASR, E2E ASR, Differentiable WFST, Topology, Viterbi Decoding

1. INTRODUCTION

In recent years, End-to-End (E2E) Automatic Speech Recognition (ASR) has simplified the development process. This field has seen a thorough exploration of methods such as Connectionist Temporal Classification (CTC) [1], Attention-based Encoder-Decoder (AED) [2], E2E Lattice-Free Maximum Mutual Information (LF-MMI) [3, 4], and Recurrent Neural Network Transducer (RNNT) [5]. Progress has been made in E2E ASR performance, with the introduction of new neural network architectures like Transformer [6], Squeezeformer [7], and Conformer [8], and the use of unlabelled data through self-supervised [9] or semi-supervised learning [10].

One challenge of training ASR models is aligning a sequence of acoustic observations with its corresponding transcription. The Hidden Markov Model (HMM) [11, 12], CTC [1] both address this by marginalising probabilities along all possible alignments. This is sometimes known as "full-sum" training [13]. As for decoding methods, we can classify them into two categories based on how they handle paths corresponding to the same hypothesis. The first style, *Viterbi-style*, finds the single best path in the searching space and considers its corresponding word sequence as the best hypothesis [11, 12]. The second one, denoted as *full-sum* in this paper, considers all or a subset of the paths associated with each hypothesis, sums the paths' probabilities to get the hypothesis' probability, and then decides the best hypothesis [14, 15]. While full-sum training is common in ASR models [16], traditional HMM-based ASR systems typically apply a Viterbi decoder to achieve optimal performance [11, 12]. Unlike HMM-based ASR, CTC-based E2E ASR usually relies on a full-sum style decoder [1, 17]. Since in most circumstances, the Viterbi decoder is more efficient [15], then a question comes into our mind, i.e., how reliable the Viterbi decoder is with a CTC-based E2E ASR? Traditionally HMM-based ASR usually applies the Bakis topology, a 3-state topology, while the CTC topology has a special 1-state topology with a special shared blank state. Thus, We also wonder whether the topology plays a significant role in the above-mentioned decoding behaviour.

In this paper, we analyze the lattices from different topologies using two self-defined metrics. To flexibly implement different topologies including CTC and calculate the loss function during training, we apply Differentiable Weighted Finite-State Transducer (DWFST) [18]. Our goal is to enhance understanding of the Viterbi decoding process with CTC-like E2E ASR.

2. METHOD

2.1. Training

Given input speech X in the form of a sequence of feature vectors or raw waveform, by applying an acoustic encoder, we obtain a sequence of hidden features or high-level representations. Next, we apply a (set of) linear layer(s) with a final softmax activation to obtain a probability distribution over each token π_t at time step t, $p_t(\pi_t|X)$, $t = 1, 2, \ldots, T$.

The posterior probability $p(Y|X)$ is modelled as

$$p(Y|X) = \frac{\sum_{\pi \in \Pi(Y; \boldsymbol{T} \circ \boldsymbol{L})} p(\pi|X)}{\sum_{Y'} \sum_{\pi \in \Pi(Y'; \boldsymbol{T} \circ \boldsymbol{L})} p(\pi|X)}, \quad (1)$$

This work was supported by a Consolidated Studentships Award funded by Huawei Technologies Co., Ltd.

where in the denominator, Y' represents any arbitrary word sequence, and $\Pi(Y; \boldsymbol{F})$ denotes the set of all the token sequences ("paths") corresponding to the output label sequence Y in the FST $\boldsymbol{F}$. $\boldsymbol{T}$ and $\boldsymbol{L}$ denote the Token FST and the Lexicon FST, respectively, and $\circ$ denotes the composition of WFSTs [19, 20, 18]. We apply the conditional independence assumption and thus

$$p(\pi|X) = \prod_{t=1}^{T} p_t(\pi_t|X). \tag{2}$$

We compute the numerator and the denominator in eq. (1) with DWFST. Specifically, for the numerator,

$$\sum_{\pi \in \Pi(Y; \boldsymbol{T} \circ \boldsymbol{L})} p(\pi|X) = \mathrm{TotalScore}(\boldsymbol{E} \circ \boldsymbol{S}^{\mathrm{trn}}), \tag{3}$$

where $\mathrm{TotalScore}$ denotes the total score operation, and $\boldsymbol{E}$ is the Emission FST constructed from the neural network model's output. $\boldsymbol{S}^{\mathrm{trn}}$ is the training graph derived from the target sequence Y,

$$\boldsymbol{S}^{\mathrm{trn}} = \boldsymbol{T} \circ (\boldsymbol{L} \circ \boldsymbol{W}), \tag{4}$$

where $\boldsymbol{W}$ is a linear Word FST with the word sequence Y as input and output labels [18]. We should emphasise that the term *topology* discussed in this paper is the topology of $\boldsymbol{T}$ which describes how to form a modelling unit (phoneme, BPE, or character) from NN's outputs.

Our previous work [21] has underscored the importance of normalisation (the denominator term in eq. (1)) for achieving training convergence. However, it is unrealistic to create a training graph as eq. (4) for every possible word sequence. Thus, to compute the denominator, given $\boldsymbol{E}$,we solely take into account the paths that are acceptable for $\boldsymbol{T}$, so we have

$$\sum_{Y'} \sum_{\pi \in \Pi(Y'; \boldsymbol{T} \circ \boldsymbol{L})} p(\pi|X) \approx \mathrm{TotalScore}(\boldsymbol{E} \circ \boldsymbol{T}). \tag{5}$$

We note that when $\boldsymbol{T}$ is set as CTC topology, as shown in fig. 1a, any arbitrary token sequence can be accepted by $\boldsymbol{T}$, so the denominator is always one, resulting in

$$p(Y|X) = \sum_{\pi \in \Pi(Y; \boldsymbol{T}_{\mathrm{CTC}} \circ \boldsymbol{L})} p(\pi|X), \tag{6}$$

which is equivalent to the CTC loss function [1]. Except for a few topologies, e.g. CTC, not all paths in $\boldsymbol{E}$ can be accepted by $\boldsymbol{T}$, which makes the denominator not equal to one.

2.2. Decoding

We construct the decoding graph as

$$\boldsymbol{S}^{\mathrm{dec}} = \boldsymbol{T} \circ (\boldsymbol{L} \circ \boldsymbol{G}), \tag{7}$$

where $\boldsymbol{G}$ is a grammar FST, often obtained from an n-gram language model [20]. For simplicity, operations such as determinisation and minimisation [19] have been omitted in the equation. With the output of the neural network model and the decoding graph, we apply the Viterbi decoder [22, 12] to get the best hypothesis

$$W^* = \arg\max_W [\log p(W) + \alpha \max_{\pi \in \Pi(W; \boldsymbol{S}^{\mathrm{dec}})} \log p(\pi|X)], \tag{8}$$

where $p(W)$ is determined by the language model, which is encoded in $\boldsymbol{S}^{\mathrm{dec}}$ as transition weights, and α is the acoustic weight.

To highlight the difference between the two decoding methods, the *full-sum* decoding strategy can be expressed as:

$$W^* = \arg\max_W [\log p(W) + \alpha \sum_{\pi \in \Pi(W; \boldsymbol{S}^{\mathrm{dec}})} \log p(\pi|X)]. \tag{9}$$

The key difference lies in whether we sum multiple (not necessarily all as some pruning may be applied) alignments' probabilities together. The latter style of decoders is commonly employed in CTC and CTC-related ASR systems [14, 1, 23, 24, 17]. Note that we only utilise the Viterbi decoder and assess its reliability with the following metrics.

2.3. Lattice and Metrics

As it is often computationally impractical to compose the Emission FST with the decoding graph and analyze the resulting FST in its entirety [22], we adopt an alternative approach to assess the reliability of results produced by the Viterbi decoder. First, for each evaluation utterance, we generate a lattice L using the method described in [22]. Then, we apply the Viterbi decoder and get the best hypothesis W^*. It is worth noting that the lattice inherently includes the best path identified by the Viterbi decoder.

Given the lattice L and the best hypothesis W^*, our first metric, *Best Path Proportion (BPP)* P_{path}, is the probability proportion of the best path among all the paths corresponding to W^*, i.e.,

$$P_{\mathrm{path}} = \frac{\max_{\pi \in \Pi(W^*; L)} p(\pi|X)}{\sum_{\pi \in \Pi(W^*; L)} p(\pi|X)}. \tag{10}$$

A higher value of P_{path} implies a more concentrated ASR model. Some people may believe that there is a dominant path in the lattice which eats up a large probability proportion, but we will see whether this is true or not in the next section.

Another metric we propose is the probability proportion of W^* within the entire lattice L, *Best Hypothesis Proportion (BHP)*, i.e.,

$$P_{\mathrm{hypo}} = \frac{\sum_{\pi \in \Pi(W^*; L)} p(\pi|X)}{\sum_{W} \sum_{\pi \in \Pi(W; L)} p(\pi|X)}, \tag{11}$$

where $\sum_W$ is the summation over all the hypotheses in L. It is important to note that, $P_{\text{hypo}} > 0.5$ indicates that the best hypothesis identified by the Viterbi decoder in eq. (8) aligns with the full-sum decoder in eq. (9), where the summation is limited to the lattice.

3. EXPERIMENTS

3.1. Settings

We conduct experiments on WSJ and Librispeech by fine-tuning the wav2vec 2.0 model[1], which was pre-trained on 960 hours of unlabeled speech data from the Librispeech dataset [9]. We fine-tune the encoder part of the model plus a linear output layer with a log-softmax activation while keeping the feature extractor fixed. We employ phonemes as modelling tokens, utilising CMUDict on WSJ and the official lexicon on Librispeech.

To enhance the generalisation capabilities of the models and mitigate the risk of overfitting, we incorporate speed perturbation and SpecAugmentation techniques [25]. We utilise the Adam optimiser with warm-up steps of 2500 and 5000 for WSJ and Librispeech, respectively. The learning rate linearly increases during the warm-up phase, reaching its maximum value (2e-5 for WSJ and 3e-5 for Librispeech), and gradually decreases afterwards.

Throughout our experiments, we employ Kaldi [26] for data preparation, PyTorch [27] for neural network training, and k2[2] as the DWFST backend. As for decoding, we utilise the Viterbi decoder, `decode-faster` in Kaldi, k2 for lattice generation. To enhance the reproducibility of our experiment results presented in this paper, we have made our code open-source.[3]

As mentioned in section 1, the topology may influence the decoding behaviour discussed in this paper. Thus, we explore eight different topologies, as summarised in fig. 1. For all the topologies, there is an optional shared blank label to ensure a fair comparison with CTC (S1-T1).

3.2. WER Performance

We apply the Viterbi decoder with a beam size of 32 and a maximum active state of 2000 on both WSJ and Librispeech. As for acoustic weight, we tune it on each development set to achieve optimal performance. On WSJ, we apply the official 4-gram language model (LM) and for Librispeech, the 3-gram LM (tglarge) is employed. The WER performance with different topologies on WSJ and Librispeech is presented in table 1. Notably, we observe that our models achieve superior WER performance on WSJ compared to the state-of-the-art results reported in [3]. We find that S3-T2★ and S3-T2★★ cannot converge very well on Librispeech for some unknown reason. However, this is useful for the following analysis with the metrics aforementioned. We would like to emphasise that the main purpose of showing the WER results is to make sure our models are working properly.

[1] `WAV2VEC2_BASE` in `torchaudio.pipelines`

[2] https://github.com/k2-fsa/k2

[3] https://github.com/ZhaoZeyu1995/Waterfall

Table 1: The WER(%) performance with different topologies on WSJ and Librispeech

Topology	WSJ		Librispeech	
	dev93	eval92	test-clean	test-other
S1-T1	3.9	2.6	3.9	8.5
S2-T1	3.9	2.7	3.9	8.2
S2-T1★	3.9	2.6	3.9	8.3
S2-T2★	3.9	2.6	4.2	9.0
S2-T2	4.2	2.8	4.0	8.3
S3-T2	4.1	2.7	4.2	8.9
S3-T2★	4.3	3.0	6.6	11.6
S3-T2★★	4.1	2.9	5.9	11.4

3.3. Lattice Analysis

For lattice generation, we utilize the same hyperparameters as the decoding phase and set the lattice beam size to match the search beam (32), ensuring that most paths from the Viterbi decoding are retained in the lattice.

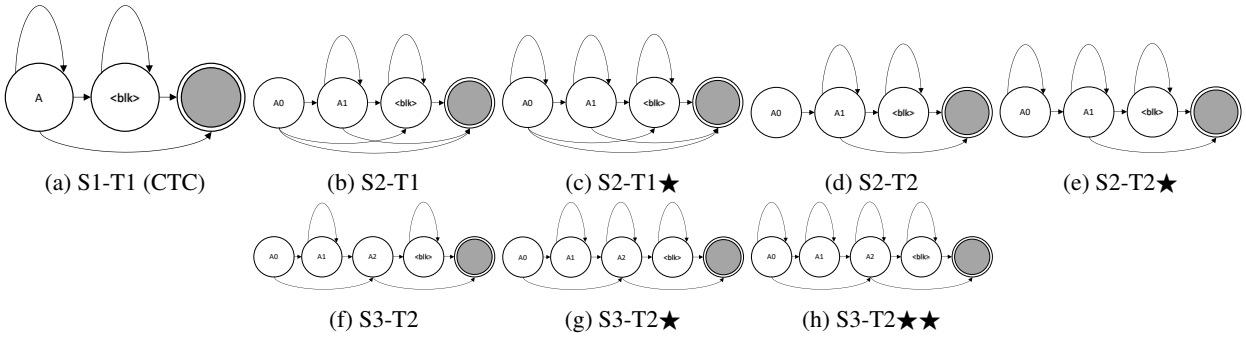
3.3.1. Best Path Proportion

We first examine the BPP, P_{path}, as defined in eq. (10). We compute the metric for all utterances in the evaluation sets (eval92 for WSJ and test-other for Librispeech), and mean values are shown in table 2.

Regardless of the applied topology, we consistently observe that P_{path} remains significantly small, with none of the values surpassing one per cent, which suggests that there is no dominant path in the searching space and expecting the model to learn a highly confident alignment is unrealistic. We hypothesise that this limitation stems from the training objective, where probabilities are summed across all possible alignments based on the transcription. It is not feasible to ask the models to learn the best alignment and assign it a very high probability as we did not give such information to the model during training. However, the question is whether the Viterbi decoder is reliable in the case where there is no dominant path, as we have seen.

Table 2: BPP (average) on WSJ and Librispeech

	WSJ (eval92)	Librispeech (test-other)
S1-T1	0.285%	0.633%
S2-T1	0.119%	0.576%
S2-T1★	0.0970%	0.621%
S2-T2★	0.0113%	0.148%
S2-T2	0.0373%	0.311%
S3-T2	0.0844%	0.364%
S3-T2★	0.00643%	0.294%
S3-T2★★	0.00390%	0.473%



(a) S1-T1 (CTC) (b) S2-T1 (c) S2-T1★ (d) S2-T2 (e) S2-T2★

(f) S3-T2 (g) S3-T2★ (h) S3-T2★★

Fig. 1: Topologies investigated in this paper, where '<blk>' denotes the shared optional blank label. Note that in some cases the blank label is unskippable in CTC but we omit it for simplicity. Sx-Ty means there are x states for each phone, the minimum traversal frame is y, and one ★ means one more self-loop is added.

Interestingly, even though the best alignment identified by the Viterbi decoder occupies a minuscule proportion, the WER performance remains acceptable, as demonstrated in table 1. We believe this can be explained by our second metric, BHP P_{hypo}.

3.3.2. Best Hypothesis Proportion

According to BHP's definition in eq. (11), when $P_{\text{hypo}} > 0.5$, the Viterbi decoding result in eq. (8) is the same as eq. (9), where multiple paths within the lattice are considered. Similarly to P_{path}, we also compute P_{hypo} for each utterance in the evaluation sets and then calculate the frequency of $P_{\text{hypo}} > 0.5$, which is presented in table 3.

Table 3: The ratio of $P_{\text{hypo}} > 0.5$ on WSJ and Librispeech

	WSJ (eval92)	Librispeech (test-other)
S1-T1	87.0%	59.5%
S2-T1	92.7%	62.1%
S2-T1★	90.9%	61.1%
S2-T2★	91.5%	62.5%
S2-T2	89.7%	60.3%
S3-T2	93.6%	58.8%
S3-T2★	91.8%	45.4%
S3-T2★★	92.1%	39.8%

We find that WER is correlated with the ratio of $P_{\text{hypo}} > 0.5$. In table 1, on WSJ, all the topologies achieve roughly the same WER performance, and their ratios are also close. Comparing WER on WSJ and Librispeech, we note that a lower ratio tends to give a higher WER. Particularly, the S3-T2★ and S3-T2★★ topologies do not perform well on Librispeech due to convergence issues, and they have very low ratios of $P_{\text{hypo}} > 0.5$. As we mentioned, a higher frequency of $P_{\text{hypo}} > 0.5$ means a higher chance of getting the Viterbi decoder equivalent to a full-sum decoder on the lattice, indicating the best hypothesis found by the Viterbi decoder more reliable.

To further investigate the relationship between the ratio and WER, we partition the utterances in eval92 into two subsets: one with $P_{\text{hypo}} > 0.5$ and the other with $P_{\text{hypo}} < 0.5$. We then evaluate the WER on these subsets separately. The results are presented in table 4.

Table 4: WER% in two subsets of eval92

	$P_{\text{hypo}} > 0.5$	$P_{\text{hypo}} < 0.5$
S1-T1	1.89	8.12
S2-T1	2.48	6.83
S2-T1★	2.33	4.72
S2-T2★	2.24	5.71
S2-T2	2.28	5.16
S3-T2	2.35	4.75
S3-T2★	2.40	7.43
S3-T2★★	2.32	5.36

The results show that the models perform better on the subset whose utterances have $P_{\text{hypo}} > 0.5$ than the other subset. This is because, again, when $P_{\text{hypo}} > 0.5$, the Viterbi process in eq. (8) is equivalent to eq. (9), where all the paths in the lattice are considered. Therefore, $P_{\text{hypo}} > 0.5$ implies that we are virtually decoding with a full-sum decoder even though the Viterbi decoder is applied. Thus, even though there is no dominant path, as we have only seen small P_{path} values, a high value of P_{hypo} can ensure that the Viterbi decoder can effectively and reliably operate. Comparing CTC (S1-T1) topology with others, it is less robust as it achieves a much better WER on the subset $P_{\text{hypo}} > 0.5$ but a relatively much worse WER on the other subset. People may prefer a model with which the Viterbi decoder is still reliable even for utterances whose $P_{\text{hypo}} < 0.5$.

4. CONCLUSION

In this study, we examined the reliability of Viterbi decoding in CTC-like E2E ASR across various topologies. Our lattice analysis revealed that a dominant path for the Viterbi-determined hypothesis is absent. However, decoding results remain reliable if the hypothesis occupies over half of the lattice. Compared with other topologies, CTC is not optimal concerning WER and robustness with the Viterbi decoder.

5. REFERENCES

[1] Alex Graves, Santiago Fernández, et al., "Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks," in *Proceedings of the 23rd International Conference on Machine Learning*, 2006, pp. 369–376.

[2] William Chan, Navdeep Jaitly, et al., "Listen, Attend and Spell," *arXiv:1508.01211 [cs, stat]*, Aug. 2015.

[3] Hossein Hadian, Hossein Sameti, et al., "End-to-end Speech Recognition Using Lattice-free MMI," in *Interspeech 2018*. Sept. 2018, pp. 12–16, ISCA.

[4] Daniel Povey, Vijayaditya Peddinti, et al., "Purely Sequence-Trained Neural Networks for ASR Based on Lattice-Free MMI," in *Interspeech 2016*. Sept. 2016, pp. 2751–2755, ISCA.

[5] Alex Graves, "Sequence Transduction with Recurrent Neural Networks," *arXiv:1211.3711 [cs, stat]*, Nov. 2012.

[6] Ashish Vaswani, Noam Shazeer, et al., "Attention is All you Need," in *Advances in Neural Information Processing Systems*. 2017, vol. 30, Curran Associates, Inc.

[7] Sehoon Kim, Amir Gholami, et al., "Squeezeformer: An Efficient Transformer for Automatic Speech Recognition," Oct. 2022.

[8] Anmol Gulati, James Qin, et al., "Conformer: Convolution-augmented Transformer for Speech Recognition," in *Interspeech 2020*. Oct. 2020, pp. 5036–5040, ISCA.

[9] Alexei Baevski, Yuhao Zhou, et al., "Wav2vec 2.0: A Framework for Self-Supervised Learning of Speech Representations," in *Advances in Neural Information Processing Systems*. 2020, vol. 33, pp. 12449–12460, Curran Associates, Inc.

[10] Yosuke Higuchi, Niko Moritz, et al., "Advancing Momentum Pseudo-Labeling with Conformer and Initialization Strategy," in *ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2022, pp. 7672–7676.

[11] Mark Gales and Steve Young, "The Application of Hidden Markov Models in Speech Recognition," *Foundations and Trends® in Signal Processing*, vol. 1, no. 3, pp. 195–304, 2007.

[12] Lawrence R Rabiner, "A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition," *PROCEEDINGS OF THE IEEE*, vol. 77, no. 2, pp. 30, 1989.

[13] Tina Raissi, Wei Zhou, et al., "HMM vs. CTC for Automatic Speech Recognition: Comparison Based on Full-Sum Training from Scratch," in *2022 IEEE Spoken Language Technology Workshop (SLT)*, Jan. 2023, pp. 287–294.

[14] Awni Y. Hannun, Andrew L. Maas, et al., "First-Pass Large Vocabulary Continuous Speech Recognition using Bi-Directional Recurrent DNNs," *arXiv:1408.2873 [cs]*, Aug. 2014.

[15] Wei Zhou, Ralf Schlüter, and Hermann Ney, "Full-Sum Decoding for Hybrid Hmm Based Speech Recognition Using LSTM Language Model," in *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, May 2020, pp. 7834–7838.

[16] Albert Zeyer, Eugen Beck, et al., "CTC in the Context of Generalized Full-Sum HMM Training," in *Interspeech 2017*. Aug. 2017, pp. 944–948, ISCA.

[17] Ronan Collobert, Christian Puhrsch, and Gabriel Synnaeve, "Wav2Letter: An End-to-End ConvNet-based Speech Recognition System," Sept. 2016.

[18] Awni Hannun, Vineel Pratap, et al., "Differentiable Weighted Finite-State Transducers," *arXiv:2010.01003 [cs, stat]*, Oct. 2020.

[19] Mehryar Mohri, Fernando Pereira, and Michael Riley, "Speech Recognition with Weighted Finite-State Transducers," in *Springer Handbook of Speech Processing*, Jacob Benesty, M. Mohan Sondhi, and Yiteng Arden Huang, Eds., pp. 559–584. Springer Berlin Heidelberg, Berlin, Heidelberg, 2008.

[20] Y. Miao, M. Gowayyed, and F. Metze, "EESEN: End-to-end speech recognition using deep RNN models and WFST-based decoding," in *2015 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, Dec. 2015, pp. 167–174.

[21] Zeyu Zhao and Peter Bell, "Investigating Sequence-Level Normalisation For CTC-Like End-to-End ASR," in *ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2022, pp. 7792–7796.

[22] Daniel Povey, Mirko Hannemann, et al., "Generating exact lattices in the WFST framework," in *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Mar. 2012, pp. 4213–4216.

[23] Shinji Watanabe, Takaaki Hori, et al., "Hybrid CTC/Attention Architecture for End-to-End Speech Recognition," *IEEE Journal of Selected Topics in Signal Processing*, vol. 11, no. 8, pp. 1240–1253, Dec. 2017.

[24] Vineel Pratap, Awni Hannun, et al., "Wav2Letter++: A Fast Open-source Speech Recognition System," in *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019, pp. 6460–6464.

[25] Daniel S. Park, William Chan, et al., "SpecAugment: A Simple Data Augmentation Method for Automatic Speech Recognition," *Interspeech 2019*, pp. 2613–2617, Sept. 2019.

[26] Daniel Povey, Arnab Ghoshal, et al., "The Kaldi speech recognition toolkit," in *IEEE 2011 Workshop on Automatic Speech Recognition and Understanding*. 2011, number CONF, IEEE Signal Processing Society.

[27] Adam Paszke, Sam Gross, et al., "Pytorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems 32*, pp. 8024–8035. Curran Associates, Inc., 2019.