INTERSPEECH 2016
September 8–12, 2016, San Francisco, USA

Phone Synchronous Decoding with CTC Lattice

Zhehuai Chen[1], *Wei Deng*[2], *Tao Xu*[2], *Kai Yu*[1]

[1]Key Lab. of Shanghai Education Commission for Intelligent Interaction and Cognitive Engineering
Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai, China
[2]AISpeech Ltd.

{chenzhehuai,kai.yu}@sjtu.edu.cn, {wei.deng,tao.xu}@aispeech.com

Abstract

Connectionist Temporal Classification (CTC) has recently shown improved efficiency in LVCSR decoding. One popular implementation is to use a CTC model to predict the phone posteriors at each frame which are then used for Viterbi beam search on a modified WFST network. This is still within the traditional frame synchronous decoding framework. In this paper, the peaky posterior property of a CTC model is carefully investigated and it is found that ignoring blank frames will not introduce additional search errors. Based on this phenomenon, a novel *phone synchronous* decoding framework is proposed. Here, a phone-level CTC lattice is constructed purely using the CTC acoustic model. The resultant CTC lattice is highly compact and removes tremendous search redundancy due to blank frames. Then, the CTC lattice can be composed with the standard WFST to yield the final decoding result. The proposed approach effectively separates the acoustic evidence calculation and the search operation. This not only significantly improves online search efficiency, but also allows flexible acoustic/linguistic resources to be used. Experiments on LVCSR tasks show that phone synchronous decoding can yield an extra 2-3 times speed up compared to the traditional frame synchronous CTC decoding implementation.

Index Terms: LVCSR, Decoder, CTC, DLSS, WFST, Lattice

1. Introduction

Large vocabulary continuous speech recognition (LVCSR) is both a pattern recognition and search problem[1], while speech recognition errors come from both sides, called modeling error and search error respectively. The search process of a speech recognizer is to find a sequence of labels whose corresponding acoustic and language models best match the input feature so as to minimize the search error. Such kind of search algorithms can mainly be divided into two types, breadth-first style *time synchronous beam search* and depth-first style *stack decoding (A* search)*. As the heuristic function for A* search is very hard to be obtained in LVCSR[2], the beam search method has dominated for decades. However, beam search is also of its weakness in wasting efforts on unpromising paths in each time frame (frame synchronous decoding). To solve this problem, WFST[3] is proposed to offline combine different knowledge sources(acoustic and language models, etc) and perform search space optimization to achieve best searching efficiency in advance[4][5]. Runtime beam pruning is conducted on all decoding paths [6], which is the main source of search error in this framework. Especially in context-dependent state-tying acoustic models[7], which leads to larger search space (compiled as HCLG with WFSTs), the search errors becomes even more critical because of applying harder pruning to reduce exhaustive search in real time decoder.

Connectionist temporal classification (CTC) [8] has been proposed as a new type of acoustic model in LVCSR and achieves state-of-the-art performance[9][10][11]. Besides, context independent phonemic CTC (CI-phone-CTC) model also shows competing performance compared with context dependent state clustering neural network (NN) HMM (CD-state-HMM) model[11][12][13][14]. With blank symbols inserted between labels, CTC constructs frame-level paths as intermediate representations to connect frame-level network outputs with label sequences[13]. The outputs from a CTC-trained model display a highly peaky distribution, while the majority of frames have the *blank* as their labels[12]. Due to this characteristics, when speech waveform is in the span of blank, any network traverse becomes redundancy because linguistic search space cannot change during that span. After the blank span, linguistic search space changes with the acoustic information from next phonemic span. At that time, network traverse should be continued. Such characteristics is named as *discontinuous linguistic search space* (DLSS) phenomenon of CTC-trained model and is further analyzed in Section 2.1.

In this paper, the potential of DLSS phenomenon in CTC-trained model can be utilized by removing tremendous search redundancy due to blank frames. Such process can also be viewed as a variant of *variable frame rate analysis*[15], *frame skipping* [16] or *frame selection method*[17]. However, the key difference is that all technologies listed are based on raw feature selection and result in limited improvement in decoding efficiency, while the proposed method applies skipping on a higher level. In this way, frame synchronous decoding is transformed into *phone synchronous decoding*. Section 2 formulates the phone synchronous decoding derived from former frame synchronous decoding and describes the empirical method to apply phone synchronization into decoding framework using the CTC lattice, a compact preserver for acoustic search space. The experiments in Section 3 show a great speedup effect of phone synchronous decoding and its extensibility in applying larger and more complex language model, with competing performance compared to other context dependent systems. The paper concludes with Section 4.

2. Phone Synchronous Decoding

2.1. From Frame Synchronization to Phone Synchronization

In LVCSR, decoding is referred to the task of finding the best word sequence. By applying dictionary and language model to

This work was supported by the Program for Professor of Special Appointment (Eastern Scholar) at Shanghai Institutions of Higher Learning, the China NSFC project No. 61573241 and the Interdisciplinary Program (14JCZ03) of Shanghai Jiao Tong University in China.

http://dx.doi.org/10.21437/Interspeech.2016-831

transform word sequence to CTC labeling, the decoding formula in CTC-trained model is derived as (3)

$$\begin{aligned}\mathbf{w}^* &= \underset{\mathbf{w}}{\operatorname{argmax}}\{P(\mathbf{w})p(\mathbf{x}|\mathbf{w})\} \\ &= \underset{\mathbf{w}}{\operatorname{argmax}}\{P(\mathbf{w})p(\mathbf{x}|\mathbf{l}_\mathbf{w})\} \quad (1) \\ &= \underset{\mathbf{w}}{\operatorname{argmax}}\{\frac{p(\mathbf{l}_\mathbf{w}|\mathbf{x})P(\mathbf{w})}{P(\mathbf{l}_\mathbf{w})}\} \quad (2) \\ &= \underset{\mathbf{w}}{\operatorname{argmax}}\{\frac{P(\mathbf{w})}{P(\mathbf{l}_\mathbf{w})}\max_{\mathbf{l}_\mathbf{w}} p(\mathbf{l}_\mathbf{w}|\mathbf{x})\} \quad (3)\end{aligned}$$

The notaions are all listed in Table 1. Here, mono-phone CTC is taken as an example (the CTC label becomes phone label). Besides, $p(\mathbf{l}_\mathbf{w})$ in (2) is the prior probability of phone labels, which can be ignored for not being related to latter derivation.

Table 1: *Notations of CTC Decoding*

Notation	Meaning
$\mathbf{w}$	word sequence
$\mathbf{w}^*$	best word sequence
$\mathbf{x}$	feature sequence
t, t'	time frame
j, j'	the j-th phone label in sequence
L	phone set
L'	$L \cup \{blank\}$
$\mathcal{B}: L' \mapsto L$	map from decoding path to CTC label sequence
y_k^t	the activation of NN output unit k at time t
$\pi_{1:t}$	frame-wise decoding path, from frame 1 to t
$\pi'_{1:j}$	phone-wise decoding path, from phone 1 to j
$\mathbf{l}$	phone label sequence
$\mathbf{l}_\mathbf{w}$	phone sequence corresponding to $\mathbf{w}$ in dictionary

For a certain CTC labeling, forward probability is defined and approximated as in [8]

$$\begin{aligned}p(\mathbf{l}|\mathbf{x}) &= \sum_{\pi:\pi\in L',\mathcal{B}(\pi_{1:t})=\mathbf{l}} \prod_{t'=1}^{t} y_{\pi_{t'}}^{t'} \\ &\cong \max_{\pi:\pi\in L',\mathcal{B}(\pi_{1:t})=\mathbf{l}} \prod_{t'=1}^{t} y_{\pi_{t'}}^{t'} \quad (4)\end{aligned}$$

Therefore, (3) can be transformed to *frame synchronous viterbi beam search* as below,

$$\mathbf{w}^* \cong \underset{\mathbf{w}}{\operatorname{argmax}}\{\frac{P(\mathbf{w})}{P(\mathbf{l}_\mathbf{w})} \max_{\pi:\pi\in L',\mathcal{B}(\pi_{1:t})=\mathbf{l}_\mathbf{w}} \{\prod_{t'=1}^{t} y_{\pi_{t'}}^{t'}\}\} \quad (5)$$

By this way, network traverse is done for each frame in an overall optimized search space [5].

(5) reveals the DLSS phenomenon discussed in Section 1, that $\pi_{t'} = blank$ doesn't change the output of $\mathcal{B}(\pi_{1:t})$, therefore, the linguistic search space is unchanged without it. And for acoustic information, if all competing paths share the same span of blank frames, ignoring those parts of blank score doesn't change(with softmax layer in model, the score is around 1). In this way, common blank time index u is defined as (6),

$$U = \{u : y_{blank}^u \simeq 1\} \quad (6)$$

Therefore, *blank* terms from (7) can be removed and changed into (8)

$$\begin{aligned}\mathbf{w}^* &\cong \underset{\mathbf{w}}{\operatorname{argmax}}\{\frac{P(\mathbf{w})}{P(\mathbf{l}_\mathbf{w})} \max_{\pi:\pi\in L',\mathcal{B}(\pi_{1:t})=\mathbf{l}_\mathbf{w}} \{\prod_{t'\notin U} y_{\pi_{t'}}^{t'} \cdot \prod_{t'\in U} y_{blank}^{t'}\}\} \quad (7) \\ &= \underset{\mathbf{w}}{\operatorname{argmax}}\{\frac{P(\mathbf{w})}{P(\mathbf{l}_\mathbf{w})} \max_{\pi:\pi\in L',\mathcal{B}(\pi_{1:t})=\mathbf{l}_\mathbf{w}} \{\prod_{t'\notin U} y_{\pi_{t'}}^{t'}\}\} \\ &= \underset{\mathbf{w}}{\operatorname{argmax}}\{\frac{P(\mathbf{w})}{P(\mathbf{l}_\mathbf{w})} \max_{\pi':\pi'\in L,\mathcal{B}(\pi'_{1:j})=\mathbf{l}_\mathbf{w}} \{\prod_{j'=1}^{j} y_{\pi'_{j'}}^{t_{j'}}\}\} \quad (8)\end{aligned}$$

As search iteration in (8) is regarding to j, which is the number of output phones ($j = t - |U|$), therefore, based on DLSS phenomenon, frame synchronous viterbi beam search has been transformed to *phone synchronous viterbi beam search*.

Algorithm 1 for phone synchronous decoding is summarized as below. Here, S and E are start and end node for the search space. Q preserves the active tokens. T is the time frame number and $\hat{B}$ preserves decoding path. The effect of called functions are as comment.

Algorithm 1 Phone Synchronous Viterbi Beam Search(S, E, Q, T)

```
Q ← S                                   ▷ initialization with start node
for each t ∈ [1, T] do                  ▷ frame-wise NN Propagation
    F ← NNPropagate(t)
    if !isBlankFrame(F) then            ▷ phone-wise WFST search
        Q=frameSynchronousViterbiBeamSearch(F, Q)
    end if
end for
B̂ ← finalTransition(E, S, Q)            ▷ to reach end node
backtrace(B̂)
```

The main differences between frame synchronous decoding (FSD) and phone synchronous decoding (PSD) is the *Decoding Interval*. In FSD, WFST network traverse is done in an equal interval (even *multiframe deep neural networks decoding* [18] traverses linguistic search space by longer but still equal interval) However, in PSD, the decoding interval is self-adjusting (smart and without performance deterioration) to remove tremendous search redundancy due to $blank$ frames, which brings about efficiency in both decoding and lattice generation.

2.2. Search Space and Decoding Error Analysis

In phone synchronous viterbi beam search, there's no loss in search space compared to frame synchronization, which has been proved in Section 2.1. On the other hand, compared to frame synchronous decoding, the number of search iterations in linguistic search space is greatly reduced, because of a large quantities of $blank$ labels (frame with the output of $blank$) in the decoding path of CTC-trained model . Suppose $blank$ rate λ for a certain utterance is defined as,

$$\lambda = \frac{\#\{U\}}{t} \quad (9)$$

Here function $\#\{\cdot\}$ is to count the number of the input. U is defined in (6) and t is the number of frames in the utterance. The average percentage of remaining active acoustic candidates in all model units is defined as β. Then the total theoretical compression rate defined as below,

$$R = 1 - (1 - \lambda) \times \beta \quad (10)$$

In general, λ is usually greater than 0.8, while β is around 0.1. Therefore, the compression in search space is effective. More quantitative statistics are given in Section 3.3.

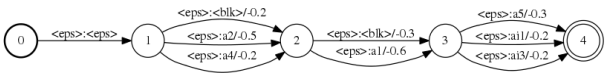

Figure 1: *CTC Lattice Example*

Thanks to the tremendous reduction in active tokens, a wider beam can be used to achieve a better performance. That's another superiority of phone synchronous decoding over frame synchronization.

2.3. the CTC Lattice - an Empirical Method to Apply Phone Synchronization into Decoding

2.3.1. Phone Lattice Generation - Extremely Compact Acoustic Information Preserver

Provided the discussion before, *CTC lattice* is proposed, as a preserver of phone level acoustic information. CTC lattice can fulfill the search optimization advantage in Section 2.2, while providing compact and flexible form for further utilization.

Figure 1 is an example of CTC lattice in WFST form. Suppose phone level acoustic information is as Table 2 (only remaining posteriors larger than 0.1 for instance) Then WFST can be built with "sausage" style that each span between two phonemic frames is built by arcs with each phone as WFST output label and its negative acoustic score as arc weight, while the input label is all set to $< eps >$ so that further WFST optimization operation can be done.

Table 2: *Acoustic Information of CTC Lattice Example*

Time	phone:score
0.4s	$< blk >: 0.2 \quad a2 : 0.5 \quad a4 : 0.2$
0.9s	$< blk >: 0.3 \quad a1 : 0.6$
1.5s	$a5 : 0.3 \quad ai1 : 0.2 \quad ai3 : 0.2$

CTC lattice is extremely compact in acoustic information compression compared to frame level posterior distribution and can be applied in both knowledge integrated decoding architecture[19][4] and multi-layerd modular decoding architecture[20][21].

2.3.2. From Phone Lattice to Word Level Lattice - Flexible Dictionary Composition

In order to integrate linguistic information into phone synchronous decoding, a dictionary is used to map the phone sequence to the word sequence. The pipeline of this procedure is

$$W \leftarrow epsremoval(P \circ T \circ L) \tag{11}$$

Here, P is CTC lattice and L is lexicon WFST built from dictionary. T is designed to filter out *blank* label and repeating phone label as in [12]. Because there are many *epsilon* labels in P, *epsilon removal* [4] is done on the composition result. Finally, P is transformed to word lattice W and ready for applying language model.

2.3.3. Language Model Composition - Changeable Linguistic Search Space

After transforming from phone level lattice to word level lattice, it's trivially to apply different types of language models by the lattice rescoring process. For example, n-gram composition[22] and RNN lattice rescoring[23].

3. Experiment

3.1. Experimental Setup

The experiments are conducted on both English and Mandarin. In training stage, the procedure and configuration is similar to [13]. The details of dataset are listed in Table 3.

Table 3: *Details of Training Setup*

Dataset	Language	Size (hours)	LM
Switchboard[24]	English	309	standard SWB
CN 300h	Mandarin	300	CN 1.7GB
CN 5000h	Mandarin	5000	CN 1.7GB

Here *standard SWB* refers to the tri-gram standard language model from Switchboard without interpolation with other sources, and *CN 1.7GB* refers to a tri-gram language model trained by [25] with *118K* words.

In the test stage, all experiments are conducted on *Intel(R) Xeon(R) CPU E5-2690 v2 @ 3.00GHz*. Besides the *hub5e00* testset from Switchboard and a Mandarin testset, *CN Cell-Phone*, is used, which is recorded in several speech scenarios and with more than 16,000 utterances (about 25 hours).

3.2. Error Rate & Decoding Time Comparison Between CD & CI Modeled by HMM & CTC

In the former discussion, it is assumed that CI-phone-CTC model is competing with CD-state-HMM model, which is one of the basis for phone synchronous decoding. In this part, a series of experiments, including different languages and training data sizes, are conducted to fully testify it. And models are all trained with similar number of parameters.

Table 4: *Performance Comparison between CI-phone-CTC & CD-state-HMM. CER / WER refers to character / word error rate and RTF refers to real-time factor in decoding*

Task	Context Dependency	Acoustic Model	CER / WER	RTF
Switchboard	CD	dnn-hmm	18.3	0.27
	CD	lstm-hmm	15.9	0.29
	CI	lstm-ctc	20.7	0.044
CN 300h	CD	dnn-hmm	16.88	0.31
	CD	lstm-hmm	13.76	0.34
	CI	lstm-ctc	14.60	0.046
CN 5000h	CD	dnn-hmm	13.30	0.32
	CI	lstm-ctc	10.20	0.044

As Table 4 displays, our result is compatible with [11][12][13][14], that with 300 hours data, performance is similar between CI-phone-CTC and CD-state-HMM, while wtih 5000 hours, CI-phone-CTC becomes more competing. Therefore, latter experiments are held on CI-phone-CTC models.

3.3. Search Space Comparison between Frame Synchronous and Phone Synchronous Decoding

To prove the analysis in Section 2.2 and fulfill the room of improvement between frame synchronous and phone synchronous decoding, statistics are conducted on both $blank$ rate λ (defined in Section 2.2) from force-aligned [26] CTC paths, and average acoustic active rate β by backtracing after Viterbi search. Finally, the theoretical compression rate R is gotten. All results are listed in Table 5

Table 5: *Search Space Statistics*

testset	$\lambda(\%)$	$\beta(\%)$	$R(\%)$
Switchboard	88	5	99.4
CellPhone	87	11	98.6

In the experiment, λ reveals the rate of network traverse reduction which is directly related to decoding speed, while R reveals the rate of acoustic information compression which is related to compactness of CTC lattice in preserving acoustic search space. For the latter, it means CTC lattice only keeps around 1% of acoustic information from LSTM propagation by each frame. Results of both indicators are impressive.

3.4. Extremely Fast Phone Synchronous Decoding

3.4.1. Experiments on Decoding Speedup

Experiment results on phone synchronous decoding (PSD) are listed in Table 6, with the most popular CTC implementation (FSD) [12] as the baseline.

Table 6: *Phone Synchronization Compared with Frame*

testset	Sync	CER / WER	S-RTF	RTF	active tokens
Switch-board	Phone	20.8	0.0082	0.017(**2.6X**)	431(**-78%**)
	Frame	20.7	0.035	0.044	1947
Cell-Phone	Phone	10.1	0.0072	0.016 (**2.75X**)	388 (**-81%**)
	Frame	10.2	0.035	0.044	2042

Result shows that, there's no ASR performance degradation in phone synchronous decoding, while achieving **2-3X** speedup compared with common CTC implementation[12], and around **30X** speedup compared with popular CD-state-NN-HMM[27] in Section 3.2. As decoding time includes neural network propagation time and WFST search time, while phone synchronous decoding mainly speedup the latter one, WFST search time divided by waveform length is individually listed as *S-RTF* in the table, which reveals **4-5X** speedup in it. Besides, the decreasing number of active tokens is parallel with both *S-RTF* in the table and λ in Table 5, which is reasonable.

Our recent research on phone synchronous decoding also shows efficiency in CD-phone-CTC model[9], which achieves similar speedup rate and compression rate as Table 5 and Table 6, with no loss in ASR performance, compared with frame synchronous decoding.

3.4.2. Experiments on Speed Robustness

In another experimental setup, language model is scaled up to test the robustness of speedup (extendibility of more complex linguistic search space) from frame to phone synchronous decoding. To make comparison in LM search space clear, n-gram model is chosen, but the result can also extend to other language model like RNN[28]. In WFST-based decoder, RTF shows nearly linear relationship with average number of active token (NAT) [22][5], so NAT is taken to measure the speed of decoding.

Figure 2 shows that NAT of phone synchronous decoding in CI-phone-CTC is almost unchanged with the growing size of language model (it is regarded as growing LM search space). In the same experiment, NAT of CD-state-NN-HMM system is always far more than phone synchronous decoding, especially when LM scales up. Besides, NAT of frame synchronous decoding in CI-phone-CTC is also growing distinctly faster compared with phone synchronous decoding.

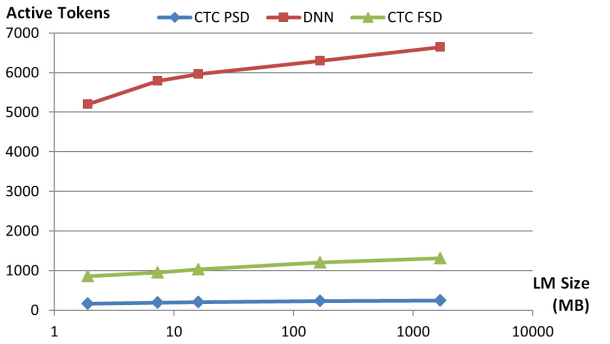

Figure 2: *LM scale-up v.s. active tokens on phone/frame synchronous decoding*

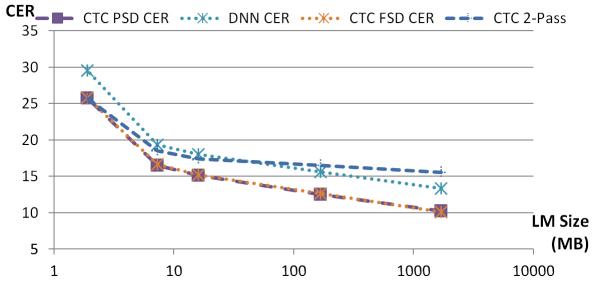

Figure 3: *LM scale-up v.s. CER on phone/frame synchronous decoding*

3.4.3. Overall Consideration of CER, Speed and Robustness

Regarding to CER, all systems above perform better with LM scaling up, while there's a small gap between CI-phone-CTC and CD-state-NN-HMM systems all the time. The performance of Phone and frame synchronous CI-phone-CTC system are very similar, which is reasonable for what have been testified in Table 6.

Besides, 1-pass (*CTC-PSD CER* in the figure) and 2-pass(*CTC 2-Pass* in the figure, *LM size* is the size of rescoring LM in 2-Pass decoder, while 1-Pass LM size is fixed) decoding of CI-phone-CTC systems is compared. Result shows that CER improvement in the latter is perceptibly smaller. The reason is that with less NAT, the word lattice generating from CI-phone-CTC decoding is smaller, which brings about worse oracle CER.

Therefore, compared to other frame synchronous system, it's more suitable to directly apply larger or more complex language model to 1-Pass phone synchronous decoder, which can bring about better performance with almost no deceleration in decoding speed.

4. Conclusions

In this paper, *frame synchronous decoding* is transformed into *phone synchronous decoding*, which can be viewed as a hybrid decoding framework of beam search and A* search for its self-adjusting decoding interval (smart and without performance deterioration) to remove tremendous search redundancy due to *blank* frames from CTC-trained model. Experiment result shows extra 2-3 times speed up compared to the traditional frame synchronous CTC decoding implementation[12]. Besides, the resultant CTC lattice as a extremely compact acoustic information preserver, shows great extensibility in applying larger and more complex language model.

Future work will be integrating CTC lattice with more knowledge sources, e.g., phonemic error model[29], prosody model[30] and more complex language model[23][31][32].

5. References

[1] X. Huang, A. Acero, and H.-W. Hon, *Spoken Language Processing: A Guide to Theory, Algorithm, and System Development*, 1st ed. Upper Saddle River, NJ, USA: Prentice Hall PTR, 2001.

[2] F. Jelinek, *Statistical methods for speech recognition*. MIT press, 1997.

[3] M. Mohri, F. Pereira, and M. Riley, "The design principles of a weighted finite-state transducer library," *Theoretical Computer Science*, vol. 231, no. 1, pp. 17–32, 2000.

[4] ——, "Weighted finite-state transducers in speech recognition," *Computer Speech & Language*, vol. 16, no. 1, pp. 69–88, 2002.

[5] T. Hori and A. Nakamura, "Speech recognition algorithms using weighted finite-state transducers," *Synthesis Lectures on Speech and Audio Processing*, vol. 9, no. 1, pp. 1–162, 2013.

[6] D. Nolden, R. Schlüter, and H. Ney, "Search space pruning based on anticipated path recombination in lvcsr." in *INTERSPEECH*. Citeseer, 2012, pp. 1015–1018.

[7] S. J. Young and P. C. Woodland, "State clustering in hidden markov model-based continuous speech recognition," *Computer Speech & Language*, vol. 8, no. 4, pp. 369–383, 1994.

[8] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, "Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks," in *Proceedings of the 23rd international conference on Machine learning*. ACM, 2006, pp. 369–376.

[9] T. Sainath, K. Rao *et al.*, "Acoustic modelling with cd-ctc-smbr lstm rnns," in *2015 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*. IEEE, 2015, pp. 604–609.

[10] D. Amodei *et al.*, "Deep speech 2: End-to-end speech recognition in english and mandarin," *arXiv preprint arXiv:1512.02595*, 2015.

[11] H. Sak, A. Senior, K. Rao, and F. Beaufays, "Fast and accurate recurrent neural network acoustic models for speech recognition," *arXiv preprint arXiv:1507.06947*, 2015.

[12] Y. Miao, M. Gowayyed, and F. Metze, "Eesen: End-to-end speech recognition using deep rnn models and wfst-based decoding," *arXiv preprint arXiv:1507.08240*, 2015.

[13] Y. Miao, M. Gowayyed *et al.*, "An empirical exploration of ctc acoustic models," in *the 41th IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*. IEEE, 2016.

[14] I. McGraw, R. Prabhavalkar, R. Alvarez, M. G. Arenas, K. Rao, D. Rybach, O. Alsharif, H. Sak, A. Gruenstein, F. Beaufays *et al.*, "Personalized speech recognition on mobile devices," *arXiv preprint arXiv:1603.03185*, 2016.

[15] K. Ponting and S. Peeling, "The use of variable frame rate analysis in speech recognition," *Computer Speech & Language*, vol. 5, no. 2, pp. 169–179, 1991.

[16] Y. Miao, J. Li, Y. Wang, S.-X. Zhang, and Y. Gong, "Simplifying long short-term memory acoustic models for fast training and decoding," in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2016, pp. 2284–2288.

[17] Z.-H. Tan and B. Lindberg, "Low-complexity variable frame rate analysis for speech recognition and voice activity detection," *Selected Topics in Signal Processing, IEEE Journal of*, vol. 4, no. 5, pp. 798–807, 2010.

[18] V. Vanhoucke, M. Devin, and G. Heigold, "Multiframe deep neural networks for acoustic modeling," in *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*. IEEE, 2013, pp. 7582–7585.

[19] F. Jelinek, L. R. Bahl, and R. L. Mercer, "Design of a linguistic statistical decoder for the recognition of continuous speech," *Information Theory, IEEE Transactions on*, vol. 21, no. 3, pp. 250–256, 1975.

[20] K. Demuynck, D. Van Compernolle, and H. Van Hamme, "Robust phone lattice decoding," in *in Proc. ICSLP*. Citeseer, 2006.

[21] S. M. Siniscalchi, T. Svendsen, and C.-H. Lee, "A bottom-up modular search approach to large vocabulary continuous speech recognition," *Audio, Speech, and Language Processing, IEEE Transactions on*, vol. 21, no. 4, pp. 786–797, 2013.

[22] T. Hori, C. Hori, Y. Minami, and A. Nakamura, "Efficient wfst-based one-pass decoding with on-the-fly hypothesis rescoring in extremely large vocabulary continuous speech recognition," *Audio, Speech, and Language Processing, IEEE Transactions on*, vol. 15, no. 4, pp. 1352–1365, 2007.

[23] X. Liu, Y. Wang, X. Chen, M. J. Gales, and P. C. Woodland, "Efficient lattice rescoring using recurrent neural network language models," in *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*. IEEE, 2014, pp. 4908–4912.

[24] J. J. Godfrey, E. C. Holliman, and J. McDaniel, "Switchboard: Telephone speech corpus for research and development," in *Acoustics, Speech, and Signal Processing, 1992. ICASSP-92., 1992 IEEE International Conference on*, vol. 1. IEEE, 1992, pp. 517–520.

[25] A. Stolcke *et al.*, "Srilm-an extensible language modeling toolkit." in *INTERSPEECH*, vol. 2002, 2002, p. 2002.

[26] P. C. Woodland, J. J. Odell, V. Valtchev, and S. J. Young, "Large vocabulary continuous speech recognition using htk," in *Acoustics, Speech, and Signal Processing, 1994. ICASSP-94., 1994 IEEE International Conference on*, vol. 2. IEEE, 1994, pp. II–125.

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

[28] T. Mikolov, S. Kombrink, L. Burget, J. H. Černockỳ, and S. Khudanpur, "Extensions of recurrent neural network language model," in *Acoustics, Speech and Signal Processing (ICASSP), 2011 IEEE International Conference on*. IEEE, 2011, pp. 5528–5531.

[29] J. Droppo and A. Acero, "Context dependent phonetic string edit distance for automatic speech recognition," in *Acoustics Speech and Signal Processing (ICASSP), 2010 IEEE International Conference on*. IEEE, 2010, pp. 4358–4361.

[30] C.-Y. Chiang, S. M. Siniscalchi, Y.-R. Wang, S.-H. Chen, and C.-H. Lee, "A study on cross-language knowledge integration in mandarin lvcsr," in *Chinese Spoken Language Processing (ISCSLP), 2012 8th International Symposium on*. IEEE, 2012, pp. 315–319.

[31] M. Lehr and I. Shafran, "Discriminatively estimated joint acoustic, duration, and language model for speech recognition." in *ICASSP*, 2010, pp. 5542–5545.

[32] N. Shazeer, J. Pelemans, and C. Chelba, "Sparse non-negative matrix language modeling for skip-grams," in *Proceedings of Interspeech*, 2015, pp. 1428–1432.