INTERSPEECH 2021
30 August – 3 September, 2021, Brno, Czechia

Variable Frame Rate Acoustic Models using Minimum Error Reinforcement Learning

Dongcheng Jiang, Chao Zhang, Philip C. Woodland

Department of Engineering, University of Cambridge, UK
{dj346,cz277,pcw}@eng.cam.ac.uk

Abstract

Frame selection in automatic speech recognition (ASR) systems can potentially improve the trade-off between speed and accuracy relative to fixed low frame rate methods. In this paper, a sequence training approach based on minimum error and reinforcement learning is proposed for a hybrid ASR system to operate at a variable frame rate, and uses a frame selection controller to predict the number of frames to skip before taking the next inference action. The controller is integrated into the acoustic model in a multi-task training framework as an additional regression task and the controller output can be used for distribution characterisation during reinforcement learning exploration. The reinforcement learning objective minimises a combined measure of the phone error and average frame rate. ASR experiments using British English multi-genre broadcast (MGB3) data show that the proposed approach achieved a smaller frame rate than using a fixed 1/3 low frame rate method and was able to reduce the word error rate relative to both fixed low frame rate and full frame rate systems.

Index Terms: Low frame rate, variable frame rate, minimum error rate, multi-task, reinforcement learning

1. Introduction

Hybrid automatic speech recognition (ASR) systems, that use acoustic models combining hidden Markov models (HMMs) with neural networks (NNs), have achieved great success in the past decade [1, 2]. For an utterance consisting of a sequence of frames, each typically representing 10 ms of audio, the HMM structure models transitions between hidden states while the NN estimates the HMM state output probabilities.

However, NN-HMM ASR systems have high computational cost, partly because they often compute and process a new set of output probabilities every frame. This is termed as full frame rate (FFR) system and increases the difficulty of implementing a highly efficient decoder. ASR with a fixed low frame rate (LFR) has also been widely studied for hybrid systems [3–7], the equivalent connectionist temporal classification systems [8, 9], and attention-based systems [9, 10]. For NN-based LFR systems, a processing rate of 1/3 of the full frame rate is often used which performs NN forwarding and decoder processing once every three frames (*e.g.* once every 30 ms). Since NNs can access a set of neighbouring frames when computing output probabilities (via a recurrent structure or an input context window), NN-HMMs for LFR use can be configured to ensure that the model uses all frames when computing output probabilities which reduces test-time computation without significantly increasing the word error rate (WER).

Recently, there has been increased interest in developing ASR systems that operate at a variable (low) frame rate (VFR) [11, 12]. This means that the ASR system can skip a variable number of frames after each step of NN forwarding and decoding, and is equivalent to test-time dynamic frame selection. The VFR idea can be traced back to various pioneering studies, some of which were published decades ago [13–15]. For hybrid systems, [11] introduced an extra softmax layer as a classification-based controller to the recurrent neural network (RNN) acoustic model, with the output units corresponding to the numbers of frames to be skipped. The controller was trained by reinforcement learning (RL) [16], whose reward function encourages the model to skip as many frames as possible but not any complete phonetic unit. The RNN-HMM acoustic model was jointly optimised with the controller based on the frame level cross-entropy (CE) loss. In [12], a dynamic sub-sampling RNN structure was proposed for end-to-end ASR, which uses a binary controller output to determine whether to skip the current frame or not.

In this paper, we propose a minimum error RL (MERL) approach for training VFR NN-HMMs, which uses a novel reward function for the trade-off between the overall average frame rate (*i.e.* the proportion of frames processed for decoding) and the number of phone errors. An extra output layer is added to a high order RNN with projection (HORNNP) acoustic model [17] to serve as the controller. In contrast to the classification-based controller in [11], our controller is regression-based with a single output unit and predicts the number of frames to skip, which can be characterised as a truncated exponential distribution for sampling-based exploration using policy gradient for RL [18]. The number of phone errors is approximated by the widely used minimum phone error (MPE) loss with pre-generated phone lattices [19]. Therefore, MERL is a discriminative sequence training approach while those used in [11, 12] are frame level training approaches. It is hence straightforward to jointly apply MPE training for the acoustic model and MERL for the controller via multi-task training. Furthermore, frame level training with the CE loss and the mean square error (MSE) loss can be used to initialise the acoustic model and the controller respectively. Experimental results with NN-HMMs trained on a 55-hour (55h) subset and the 275h full set of the multi-genre broadcast (MGB3) training data show that our MERL-based VFR systems can achieve both lower overall frame rates and lower WERs than the baseline fixed rate LFR systems.

The remainder of this paper is organised as follows. Section 2 presents the model structures for our VFR systems and the frame level pre-training. Section 3 describes the details of MERL, including the objective and exploration. The experimental setup and results can be found in Sections 4 and 5 respectively, and conclusions are given in Section 6.

2. Multi-task Model & Controller Training

Our VFR ASR system is designed to perform both acoustic modelling and frame selection. We adopt a multi-task joint training approach: one task is acoustic modelling, and the other

http://dx.doi.org/10.21437/Interspeech.2021-2198

is optimising a frame rate controller. For acoustic modelling, frame level CE training is first performed. The controller is designed to predict skip actions based on the remaining time (measured as the number of frames) of the current phone, which is set as the target for regression during frame level pre-training. Hence it is beneficial that the controller and the acoustic model share hidden layer information. The multi-task setup during frame level training also allows the combination of training information (gradients). After frame level training, the acoustic model can be further improved via sequence level MPE training [19–21]. Furthermore, the frame rate controller can be further optimised via MERL which will be explained in Section 3. The gradient for MERL is not propagated to the shared layers otherwise the noisy gradients may be harmful to the final accuracy. Thus the complete acoustic model and controller are trained using two sequence level training methods and performs sequence level multi-task training. Details of the acoustic and controller models are presented below.

2.1. Phone HMMs

In conventional FFR hybrid ASR systems, the phone HMMs often have 3 states and a left-to-right structure as shown in Figure 1(a). The phone HMM structure can affect duration modelling [22], and this structure ensures that at least three frames are aligned with each phone. However, for LFR systems, since some frames are being skipped, this restriction is no longer appropriate. Therefore a phone HMM structure which contains only one state was used in the LFR systems, as shown in Figure 1(b). Although the total number of states is reduced, decision tree clustering [23] is still needed so that the total number of context dependent acoustic model NN outputs can be kept at a reasonable level.

(a) 3-state HMM (b) 1-state HMM

Figure 1: *HMM structures for conventional FFR systems (3-state) and LFR systems (1-state). Each unshaded circle represents an HMM state.*

2.2. Acoustic model and frame rate controller

Following the Skip-RNN structure proposed in [11], our VFR systems use two output branches on the main acoustic model NN for the two tasks as shown in Figure 2: one for acoustic modelling and the other to determine frame skipping. The system starts processing an utterance from the beginning, and once the input has been fed forward through the NN, both the acoustic scores and the number of following frames to be skipped are found from the NN output, and then the system moves on to process the next selected frame.

Both output branches each contain one further separate hidden layer and an output layer. The controller branch has a single output node that predicts the number of frames to be skipped. All other NN parameters are shared. The controller performs a regression and the controller output need to be rounded to the nearest integer when decoding. Firstly, the acoustic model and the controller are trained simultaneously at the frame level via multi-task training. Conventional stochastic gradient descent is used for CE acoustic model training and MSE controller training. By carefully adjusting the gradient scaling from the controller branch, the whole system can be jointly pre-trained. The labels for frame level controller training are the number of remaining frames of the phone unit from the current frame. For example, if a phone is aligned with three frames, for the first frame, the controller label would indicate that two frames remain. This leads to the technique later used for MERL at the sequence level: skipped frames are assigned the same acoustic score as the most recent preceding non-skipped frame (see Figure 2) in lattice acoustic rescoring. Since skipping too many frames can be harmful to the accuracy, the experiments set a maximum number of frames that can be skipped at each step as a hyperparameter. This maximum skip value is used when determining the frame level training labels for the controller, test-time decoding, and MERL-based sequence level training.

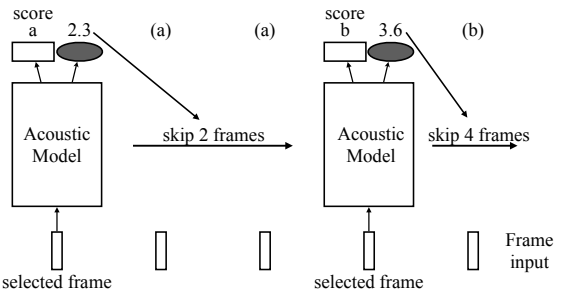

Figure 2: *VFR model with the frame rate controller shaded. Single lower case letters in the top row represent the acoustic scores (those in brackets are copied from the previous non-skipped frame and used in acoustic lattice rescoring in MERL). Values from controller output are rounded to the nearest integer.*

3. Minimum Error Reinforcement Learning

Having obtained an initial VFR system from frame level multi-task pre-training, both the acoustic model and controller can be further improved using sequence level discriminative training. Given an input frame sequence, the controller needs to take a sequence of actions to select which frames to process, which can be handled by deep RL [24] to calculate the gradients with a non-differential reward function for frame selection.

There are two objectives for MERL: reducing the frame rate and reducing the WER. Both of these are defined at the sequence level and should be suitably weighted to give good overall performance. Note that frame level training in general may not result in an optimal sequence level solution. By using a sequence level reward function and allowing random exploration, MERL can further optimise the controller via a better overall training objective. Starting from the first frame in an utterance, sampling and then proceeding to the corresponding next frame selected happen alternately until the end of the utterance and a sampled action sequence is obtained for gradient calculation and training. The controller is naturally a policy network which generates skip actions based on frame inputs, and hence it can be optimised via policy gradient [18].

3.1. Objective and policy gradient

Given an input utterance with a skip action sequence, the relative frame rate "FrameRate" is the proportion of frames selected in decoding. The MPE loss aligns better with the WER evaluation metric for ASR [19] and it approximates the number of

phone errors in a word recognition context. Define "MPEBase" as the MPE loss value (i.e. estimated number of phone errors[1]) for the given utterance calculated using all of its frames, and "MPESample" as the loss value calculated using the selected frames only. If the sequence of controller actions are denoted as a_1, a_2, ..., a_n, the negative reward function J to be minimised during MERL can be expressed as follows:

$$J(a_{1:n}) = \text{MPESample} - \text{MPEBase} + \alpha\,\text{FrameRate} \quad (1)$$

where α is a weighting factor and $a_{1:n}$ is shorthand for a_1, a_2, ..., a_n. To calculate "MPESample", lattice acoustic rescoring based on sampled skips can be performed using the technique mentioned in Section 2.2: for current frame f being processed, if the skip is k then the following k frames will be assigned the same acoustic scores as for frame f.

Denote the parameters in the controller branch as θ_c. Given an utterance with a total of T input frames f_1, f_2, ..., f_T, and using Ω to represent the space of action sequences, and $\pi_{\theta_c}(\cdot)$ for the policy, the expected objective can be expressed as follows considering how an action sequence is generated:

$$\begin{aligned}\mathbb{E}[J(a_{1:n})] &= \sum_{a_{1:n}\in\Omega} \pi_{\theta_c}(a_{1:n}|f_{1:T})J(a_{1:n}) \quad (2)\\ &= \sum_{a_{1:n}\in\Omega} [\prod_{i=1}^{n} \pi_{\theta_c}(a_i|f_{a_{1:i-1}})]J(a_{1:n})\end{aligned}$$

where $f_{a_{1:i-1}}$ is the frame input being processed after actions $a_{1:i-1}$ (define $f_{a_{1:0}} = f_1$ because the system starts at the first frame by default). The expectation can be approximated by sampling action sequences. Taking gradients with respect to θ_c, the following gradient expressions can be obtained:

$$\begin{aligned}\nabla_{\theta_c}\mathbb{E}[J(a_{1:n})] &= \sum_{a_{1:n}\in\Omega} J(a_{1:n})\nabla_{\theta_c}\pi_{\theta_c}(a_{1:n}) \quad (3)\\ &= \mathbb{E}[J(a_{1:n})\nabla_{\theta_c}\ln(\prod_{i=1}^{n}\pi_{\theta_c}(a_i|f_{a_{1:i-1}}))]\\ &= \mathbb{E}[J(a_{1:n})(\sum_{i=1}^{n}\nabla_{\theta_c}\ln\pi_{\theta_c}(a_i|f_{a_{1:i-1}}))]\end{aligned}$$

3.2. Exploration

In Section 2.2 the controller as a regressor has a deterministic output. In order to perform exploration, the output should be treated as a parameter characterising a probability distribution. For a Poisson process with arrival rate λ, the time interval between two arrivals can be modelled using an exponential distribution with mean $\frac{1}{\lambda}$ (measured in frames). Ideally, given the current input the controller output models the time interval locally, i.e. predicts the expectation $\frac{1}{\lambda}$. However, due to the limit on the number of frames that can be skipped at each time, each action needs to be drawn from a truncated exponential distribution, while this truncated distribution can still be characterised using λ via the controller output. When no more than M frames can be skipped, the probability distribution function is:

$$p(x,\lambda) = \frac{1}{1-\exp(-\lambda M)}\lambda\exp(-\lambda x), \quad 0 \le x \le M \quad (4)$$

Take the first gradient term in the last line of Eqn. (3) as an example, with controller output $\hat{y}^{(1)} = 1/\lambda$. After sampling a_1 is drawn and the first gradient term can be calculated as follows:

$$\begin{aligned}\frac{\partial \ln\pi_{\theta_c}(a_1|f_1)}{\partial\theta_c} &= \frac{\partial\ln p(a_1,\lambda)}{\partial\lambda}\frac{\partial\lambda}{\partial\hat{y}^{(1)}}\frac{\partial\hat{y}^{(1)}}{\partial\theta_c}\\ &= [a_1 - \frac{M\exp(-M/\hat{y}^{(1)})}{1-\exp(-M/\hat{y}^{(1)})} - \hat{y}^{(1)}]\frac{1}{(\hat{y}^{(1)})^2}\frac{\partial\hat{y}^{(1)}}{\partial\theta_c} \quad (5)\end{aligned}$$

where the last term $\partial\hat{y}^{(1)}/\partial\theta_c$ can be computed via backward propagation as indicated in [25, 26]. Once a sample is drawn, since the number of frames can only be non-negative integers, the sampled value needs to be rounded to get skips for lattice acoustic rescoring and frame rate calculation.

4. Experimental Setup

The proposed training methods together were tested on the data sets from the MGB3 English challenge [27, 28], which contains audio from BBC TV programmes covering different genres. As in [17], the full training set contains 275h of audio, and a 55h subset was sampled at the utterance level from the full set. To allow comparisons with [17], the same trigram language model, confusion network decoding [29] and 5.55h test set **dev17b** was used. All experiments were conducted using an extended version of HTK 3.5 [30].

Speech frames were parameterised into 40d log mel filter bank features using an analysis window size of 25 ms and a 10 ms time shift. For the 55h/275h training set, about 6k/9k triphone tied-states were obtained after decision tree clustering. For the (variable) low frame rate systems, the 1-state triphone HMM structure in Figure 1(b) was used. The 2-layer ReLU HORNNP in [17] was used for acoustic modelling, which is a 2-layer ReLU RNN using high order connections connecting current hidden state with not only the previous hidden state but also the hidden state 4 time steps before, and low-rank projection to reduce the total number of parameters. The FFR HORNNP setup from [17] will be referred as FFR below. When experimenting with the 55h training set, the hidden vector size and projected vector size were set to 500 and 250. When using the 275h training set, the sizes were increased to 1000 and 500 respectively for improved modelling. The RNN was unfolded for 20 time steps such that for each input frame there were also 5 steps of future input context and 14 steps of history. Input frame stacking allows better information coverage without significantly increasing the computational cost during decoding. Systems using a fixed low frame rate (LFR) were setup whose input frame at each time step was stacked with its -1 and +1 time shifts. For the variable frame rate systems (VFR), an input frame was stacked with its -2, -1, +1, and +2 time shifts. Therefore given the current frame the NN could access at most 7 time steps of future context, and as a result no more than 7 frames could be skipped at each time throughout all the experiments.

The experiments can be divided into 2 stages: frame level training using CE (and MSE) loss, and sequence training using MPE (and/or MERL). All the training data were shuffled at frame level in the first stage and at utterance level in the second stage. In VFR systems, during frame level training, scaling for the gradients in the controller output layer should be small and was carefully adjusted to optimise the overall performance and obtain desired systems, because the raw MSE loss had a larger magnitude compared to CE. The lattices used for VFR system sequence training were generated by the corresponding LFR systems. For training efficiency in MERL, one action sequence was drawn to approximate the expectation being minimised for

[1]Note that [19] maximises an accuracy measure in MPE training.

each utterance. The weighting coefficient α in Eqn. (1) was set to 0.01 after observing the training log files.

5. Experimental Results

The proposed approach was first evaluated using the 55h training set after frame level training. Since the frame rate is here defined as the relative frame rate, full frame rate (FFR) systems have a frame rate of one by definition. When decoding, language model scaling was adjusted according to the overall frame rate to maintain the insertion rate in a suitable range and ensure reasonable WERs. Table 1 shows the decoding WERs for different systems after frame level training. Relative decoding frame rates of 1/3 and 1/4 for the fixed low frame rate (LFR) setup were used as a comparison. It can be seen that the increase of WER from using fewer frames is not large: by reducing the frame rate from 1/3 to 1/4 in an LFR system, the WER increased by 0.5% absolute. However, the VFR system was able to get a better trade-off between frame rate and WER according to Table 1: compared to the 1/3 frame rate LFR system, the WER decreased by 0.2% absolute while the frame rate was further reduced by 16.9%. Therefore the VFR system after multi-task frame level training should be an appropriate starting point for sequence training.

Table 1: *%WERs of 55h systems after frame level training.*

System	Training	Frame Rate	%WER
FFR	CE	1	30.7
LFR	CE	1/3	30.5
LFR	CE	1/4	31.0
VFR	CE+MSE	1/3.61	30.3

During frame level training, if the gradient in the output layer of the controller is scaled to be too small, the trained VFR system tends to have a relatively high frame rate, and if the scaling is too large the system tends to have very low frame rate and hence a high WER which is not suitable for further training. Therefore the effect of MERL alone was observed through a VFR system with smaller gradient scaling during frame level training compared to the one in Table 1. It can be seen in Table 2 that the resulting VFR system after MERL has a significantly lower frame rate compared to the original system. According to Table 2, while the frame rate was reduced by 32.3%, the decoding WER almost remained the same. Therefore MERL should be effective in adjusting the controller to make larger skips on average. The experiments also showed that when a VFR system already had a relatively low frame rate overall, by applying MERL training while the frame rate could be reduced, the accuracy was likely to degrade as well.

Table 2: *%WERs of a 55h VFR system with relatively high frame rate before and after MERL training.*

System	Training	Frame Rate	%WER
VFR	CE+MSE	1/2.37	30.6
VFR	MERL	1/3.50	30.5

Having observed the effect of MERL, effort switched to applying both MERL and MPE to further optimise the VFR system in Table 1 with results given in Table 3. The WER in Table 3 for the combined MPE and MERL VFR system was reduced by 0.8% absolute compared to the VFR system in Table 1, while the frame rate increased slightly by 12.8%. MPE training was also applied to the fixed frame rate LFR system in Table 1 and the LFR system in Table 3 was obtained as a comparison. Table 3 shows that the combined MPE and MERL VFR system has both a lower frame rate compared to the fixed frame rate MPE LFR system and also a 0.7% absolute lower WER. Hence the use of MERL with MPE can help improve the accuracy of the CE+MSE VFR system.

Table 3: *%WERs of 55h systems after MPE (+MERL) training. To be compared with corresponding systems in Table 1.*

System	Training	Frame Rate	%WER
LFR	MPE	1/3	30.2
VFR	MPE+MERL	1/3.20	29.5

To investigate scaling MERL with MPE training for VFR systems to more data, some experiments were performed using the full 275h training set. We performed multi-task frame level training to form an initial VFR system, and then it was trained via multi-task sequence training to obtain the results in Table 4. The WERs of FFR and LFR systems were also included in Table 4 for comparison.

Table 4: *%WERs of 275h systems. The systems include frame-level training FFR, LFR and VFR systems as well as sequence level training for LFR (MPE) and VFR (MPE+MERL).*

System	Training	Frame Rate	%WER
FFR	CE	1	25.0
LFR	CE	1/3	25.3
LFR	MPE	1/3	24.8
VFR	CE+MSE	1/3.64	25.3
VFR	MPE+MERL	1/3.14	24.7

It can be seen in Table 4 that the WER difference on the 275h setup between the FFR and fixed rate LFR systems after frame level CE training is larger than in Table 1. The VFR system was tuned to have an even lower frame rate of 1/3.64, and in this case it has the same WER as the LFR system after frame level training. However, after applying both MPE and MERL to the 275h VFR system, while the overall frame rate increased to 1/3.14, the WER was reduced by 0.6% absolute compared to the frame level trained VFR system and is lower than the WERs of both the FFR system and the LFR system after MPE training as shown in Table 4. The general trends of results for VFR systems given by multi-task sequence training (MPE+MERL) are broadly similar to those demonstrated when using the 55h training set.

6. Conclusions

In this paper, a novel way of building and training variable low frame rate ASR systems that utilise multi-task training and minimum error reinforcement learning (MERL) has been proposed. After multi-task frame level training, variable frame rate systems can have a lower frame rate overall compared to the fixed frame rate systems with similar accuracy. MERL was found to be helpful in encouraging the frame rate controller to make appropriately larger skips on average within a reasonable range. Variable frame rate systems after frame level training can be further optimised by using MERL together with MPE training to obtain both improved recognition accuracy and on average a lower frame rate than the fixed low frame rate systems.

7. References

[1] H. Bourlard and N. Morgan, *Connectionist Speech Recognition: A Hybrid Approach*. Norwell, MA, USA: Kluwer Academic Publishers, 1993.

[2] G. Hinton, L. Deng, D. Yu, G. Dahl, A. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. Sainath, and B. Kingsbury, "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 82–97, 2012.

[3] V. Vanhoucke, M. Devin, and G. Heigold, "Multiframe deep neural networks for acoustic modeling," in *Proc. ICASSP*, 2013, pp. 7582–7585.

[4] Y. Miao, J. Li, Y. Wang, S. Zhang, and Y. Gong, "Simplifying long short-term memory acoustic models for fast training and decoding," in *Proc. ICASSP*, 2016, pp. 2284–2288.

[5] G. Pundak and T. N. Sainath, "Lower frame rate neural network acoustic models," in *Proc. Interspeech*, 2016, pp. 22–26.

[6] D. Povey, V. Peddinti, D. Galvez, P. Ghahremani, V. Manohar, X. Na, Y. Wang, and S. Khudanpur, "Purely sequence-trained neural networks for ASR based on lattice-free MMI," in *Proc. Interspeech*, 2016, pp. 2751–2755.

[7] V. Peddinti, Y. Wang, D. Povey, and S. Khudanpur, "Low latency acoustic modeling using temporal convolution and LSTMs," *IEEE Signal Processing Letters*, vol. 25, no. 3, pp. 373–377, 2018.

[8] H. Sak, A. Senior, K. Rao, and F. Beaufays, "Fast and accurate recurrent neural network acoustic models for speech recognition," in *Proc. Interspeech*, 2015, pp. 1468–1472.

[9] Q. Li, C. Zhang, and P. Woodland, "Integrating source-channel and attention-based sequence-to-sequence models for speech recognition," in *Proc. ASRU*, 2019, pp. 39–46.

[10] W. Huang, W. Hu, Y. Yeung, and X. Chen, "Conv-transformer transducer: low latency, low frame rate, streamable end-to-end speech recognition," in *Proc. Interspeech*, 2020, pp. 5001–5005.

[11] I. Song, J. Chung, T. Kim, and Y. Bengio, "Dynamic frame skipping for fast speech recognition in recurrent neural network based acoustic models," in *Proc. ICASSP*, 2018, pp. 4984–4988.

[12] S. Zhang, E. Loweimi, Y. Xu, P. Bell, and S. Renals, "Trainable dynamic subsampling for end-to-end speech recognition," in *Proc. Interspeech*, 2019, pp. 1413–1417.

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

[14] Q. Zhu and A. Alwan, "On the use of variable frame rate analysis in speech recognition," in *Proc. ICASSP*, 2000, pp. 1783–1786.

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

[16] R. Sutton and A. Barto, *Reinforcement Learning: An Introduction (2nd edition)*. Cambridge, MA, USA: The MIT Press, 2018.

[17] C. Zhang and P. Woodland, "High order recurrent neural networks for acoustic modelling," in *Proc. ICASSP*, 2018, pp. 5849–5853.

[18] R. Williams, "Simple statistical gradient-following algorithms for connectionist reinforcement learning," *Machine Learning*, vol. 8, no. 3, p. 229–256, 1992.

[19] D. Povey and P. Woodland, "Minimum phone error and I-smoothing for improved discriminative training," in *Proc. ICASSP*, 2002, pp. I–105–108.

[20] B. Kingsbury, "Lattice-based optimization of sequence classification criteria for neural-network acoustic modeling," in *Proc. ICASSP*, 2009, pp. 3761–3764.

[21] K. Vesely, A. Ghoshal, L. Burget, and D. Povey, "Sequence discriminative training of deep neural networks," in *Proc. Interspeech*, 2013, p. 2345–2349.

[22] A. Senior, H. Sak, and I. Shafran, "Context dependent phone models for LSTM RNN acoustic modelling," in *Proc. ICASSP*, 2015, pp. 4585–4589.

[23] S. Young, J. Odell, and P. Woodland, "Tree-based state tying for high accuracy acoustic modelling," in *Proc. HLT*, 1994, pp. 307–312.

[24] Y. Keneshloo, T. Shi, N. Ramakrishnan, and C. Reddy, "Deep reinforcement learning for sequence-to-sequence models," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 31, no. 7, pp. 2469–2489, 2020.

[25] M. Ranzato, S. Chopra, M. Auli, and W. Zaremba, "Sequence level training with recurrent neural networks," in *Proc. ICLR*, 2016, pp. 1–12.

[26] W. Zaremba and I. Sutskever, "Reinforcement learning neural turing machines – revised," *arXiv preprint:1505.00521*, 2015.

[27] Y. Wang, X. Chen, M. J. F. Gales, A. Ragni, and J. H. M. Wong, "Phonetic and graphemic systems for multi-genre broadcast transcription," in *Proc. ICASSP*, 2018, pp. 5899–5903.

[28] P. Bell, M. Gales, T. Hain, J. Kilgour, P. Lanchantin, X. Liu, A. McParland, S. Renals, O. Saz, M. Wester, and P. Woodland, "The MGB challenge: Evaluating multi-genre broadcast media transcription," in *Proc. ASRU*, 2015, pp. 687–693.

[29] G. Evermann and P. Woodland, "Large vocabulary decoding and confidence estimation using word posterior probabilities," in *Proc. ICASSP*, 2000, pp. 1655–1658.

[30] S. Young, G. Evermann, M. Gales, T. Hain, D. Kershaw, X. Liu, G. Moore, J. Odell, D. Ollason, D. Povey, A. Ragni, V. Valtchev, P. Woodland, and C. Zhang, *The HTK Book (for HTK version 3.5)*. Cambridge University Engineering Department, 2015.