TOKEN-WISE TRAINING FOR ATTENTION BASED END-TO-END SPEECH RECOGNITION

*Peidong Wang**

Department of Computer Science and Engineering
The Ohio State University, Columbus, USA
wang.7642@osu.edu

Jia Cui, Chao Weng, Dong Yu

Tencent AI Lab, Bellevue, USA
{jiaacui, cweng, dyu}@tencent.com

ABSTRACT

In attention based end-to-end (A-E2E) speech recognition systems, the dependency between output tokens is typically formulated as an input-output mapping in decoder. Due to such dependency, decoding errors can easily propagate along output sequence. In this paper, we propose a token-wise training (TWT) method for A-E2E models. The new method is flexible and can be combined with a variety of loss functions. Applying TWT to multiple hypotheses, we propose a novel TWT in beam (TWTiB) training scheme. Trained on the benchmark Switchboard (SWBD) 300h corpus, TWTiB outperforms the previous best training scheme on the SWBD evaluation subset.

***Index Terms*—** token-wise training, attention based end-to-end speech recognition, early update, optimization

1. INTRODUCTION

Attention based end-to-end (A-E2E) speech recognition systems map speech signals directly to tokens (characters/subwords/words) [1, 2, 3, 4, 5]. This enables their usage of evaluation metrics such as character error rate (CER) and word error rate (WER) directly as training objectives. In order to back propagate CER/WER loss to model parameters, Karita *et al.* introduced policy gradient [6] to A-E2E model training [7]. The expected CER/WER is first calculated by a sampling based method. Policy gradient is then applied to directly optimize the expected CER/WER [8]. Inspired by minimum Bayes risk (MBR) training for conventional hidden Markov model (HMM) based systems [9], Prabhavalkar *et al.* proposed minimum word error rate (MWER) training for A-E2E systems [10]. The loss of MWER training is the expected number of word errors. Due to the inefficiency of calculating such loss, they use two approximation methods. The first approach is based on sampling and the second one uses the n-best hypotheses generated during training. Experimental results show that the n-best hypotheses based loss estimation outperforms the sampling based one. Weng *et al.* [5] improved MWER training with softmax smoothing for the n-best hypotheses generation process. Competitive results were obtained on the Switchboard (SWBD) 300h corpus [5]. Recently, an investigation was conducted by Cui *et al.* comparing various sequence discriminative training criteria for A-E2E systems [11]. In addition to training criteria, investigations of using subword units as output tokens were also actively conducted [12, 13].

During the decoding process of A-E2E systems, output tokens are used as inputs to generate the next token in the sequence. Due to such sequential dependency, correcting the first few errors may be important to good recognition performances. The fine-grained partial error proposed by Karita *et al.* can be viewed as an attempt towards this direction [7]. With length-normalized edit distance, tokens at the beginning of the output sequence are assigned more weights in the loss. Their experimental results show that the employment of the fine-grained partial error is crucial to the performance improvement [7].

An extreme case of weighting based approaches is to completely mask out the errors after the first wrong token. Such a training scheme is called "early update" in natural language processing (NLP) tasks such as tagging and parsing [14], machine translation [15], and spoken dialogue processing [16].

Early update may assign nonzero gradients to the parameters corresponding to correct tokens. For A-E2E speech recognition, this may divert the training process and lead to overfitting. In this paper, we propose a token-wise training (TWT) method for A-E2E systems. During each epoch, TWT only updates the first wrong token in the output sequence, leaving all other tokens untouched. Applying TWT to multiple hypotheses, we propose a novel TWT in beam (TWTiB) training scheme. Experimental results on SWBD 300h corpus show that TWTiB outperforms not only MWER but also a pretraining scheme yielding the previous best result on the SWBD subset of the 2000 HUB5 evaluation set [4].

The rest of this paper is organized as follows. In Section 2, we describe the TWT method for A-E2E speech recognition systems. Section 3 and 4 contain the experimental setup and results, respectively. We make a conclusion in Section 5.

*work performed during an internship at Tencent AI Lab

2. SYSTEM DESCRIPTION

2.1. Output Token Dependency

A typical A-E2E system can be expressed as equations (1) - (3) during evaluation:

$$\mathbf{H}^{enc} = Encoder(\mathbf{X}) \tag{1}$$

$$\mathbf{p}_t = Decoder(y_{t-1}, \mathbf{H}^{enc}) \tag{2}$$

$$y_t = \underset{i}{\text{argmax}}\{p_{t,i}\} \tag{3}$$

where matrix $\mathbf{X}$ denotes the input audio features, $\mathbf{H}^{enc}$ the encoded features, y_t the output token at time t, $\mathbf{p}_t$ the posterior probability vector at t, and $p_{t,i}$ the probability at output node i and time t. Note that decoder states and attention mechanisms are omitted in these equations since they are not related to our analysis below.

Without scheduled sampling, y_{t-1} in equation (2) is substituted by r_{t-1} in the golden transcription during training:

$$\mathbf{p}_t = Decoder(r_{t-1}, \mathbf{H}^{enc}) \tag{4}$$

To alleviate the mismatch between training and evaluation, scheduled sampling [17] is commonly employed during the training process. It uses a mixture of output token y_{t-1} and reference token r_{t-1}:

$$\mathbf{p}_t = Decoder(s_{t-1} \in \{r_{t-1}, y_{t-1}\}, \mathbf{H}^{enc}) \tag{5}$$

where s_{t-1} is a token randomly selected from $\{r_{t-1}, y_{t-1}\}$.

In equation (5), due to the dependency of y_t on y_{t-1}, a wrong output token at time $t-1$ could easily cause errors in the following tokens $y_t, y_{t+1}, ...$ Assume that the first wrong token in y is at t^w. In an extreme case, the remaining tokens in y may have no overlap with those in r at all, as is shown in Fig. 1. The horizontal line on the left denotes the output segment $y_1 : y_{t^w-1}$ before y_{t^w}. Arrows pointing upwards and downwards correspond to the reference token r_{t^w} and the wrong token y_{t^w}, respectively. If we denote the lengths of r and y as M and N, respectively, then the remaining tokens in r and y after t^w can be expressed as $r_{t^w+1} : r_M$ and $y_{t^w+1} : y_N$, respectively. The horizontal dashed line on the right of Fig. 1 denotes that there is no overlap between $r_{t^w+1} : r_M$ and $y_{t^w+1} : y_N$. Based on Fig. 1, it may be important to correct the first wrong token y_{t^w} during training. If y_{t^w} is corrected, the optimization on $y_{t^w+1} : y_N$ could be handled with less effort.

2.2. Token-Wise Training

2.2.1. *Position-Aware Loss Functions*

Let $l_\theta(y_t, r_t)$ denote the loss of hypothesis y at time t, $L(\theta)$ the total loss over the training set, and θ the model parameters. A position-unaware training scheme can be expressed as follows:

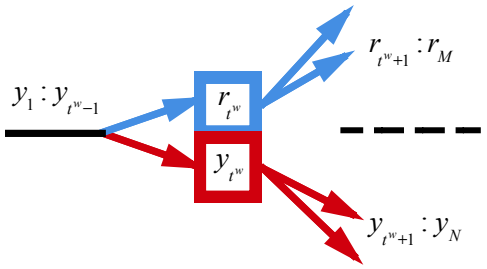

Fig. 1. An extreme case of r and y caused by y_{t^w}

$$L(\theta) = \sum_{(y,r)\in(Y,R)} \frac{1}{T} \sum_{t=1}^{T} l_\theta(y_t, r_t) \tag{6}$$

where (Y, R) denotes hypothesis-reference pairs in the whole training set and T is typically chosen to be $\min(M, N)$.

The above loss function treats all positions in the output sequence equally, making it difficult to model the output token dependency in A-E2E models. A simple modification of equation (6) is to assign more weights to $l_\theta(y_t, r_t)$'s at small t's:

$$L(\theta)_{weighted} = \sum_{(y,r)\in(Y,R)} \sum_{t=1}^{T} \frac{l_\theta(y_t, r_t)}{t} \tag{7}$$

Since the errors in $y_{t^w+1} : y_N$ may relate to y_{t^w} implicitly, it is difficult for weighting based methods such as $L(\theta)_{weighted}$ to assign appropriate weights to the errors after t^w. If we ignore the errors in $y_{t^w+1} : y_N$ and assign ones as weights for $y_1 : y_{t^w}$, we can get the early update training scheme as follows:

$$L(\theta)_{early} = \sum_{(y,r)\in(Y,R)} \frac{1}{t^w} \sum_{t=1}^{t^w} l_\theta(y_t, r_t) \tag{8}$$

Note that the gradients of $l_\theta(y_t, r_t)$'s w.r.t. the correctly decoded segment $y_1 : y_{t^w-1}$ may not be zero. Early update could thus enforce parameter updates for the correct tokens. This may divert the training process and lead to overfitting. In order to avoid such problems, we propose token-wise training (TWT) in equation (9) below:

$$L(\theta)_{TWT} = \sum_{(y,r)\in(Y,R)} l_\theta(y_{t^w}, r_{t^w}) \tag{9}$$

2.2.2. $l_\theta(y_t, r_t)$ *Selection*

TWT defined in equation (9) can be combined with various $l_\theta(y_t, r_t)$'s. In this study, we adopt two $l_\theta(y_t, r_t)$'s, *Ref* and *Ref+Err*.

Ref is essentially a cross entropy (CE) loss at time step t^w. It can be expressed as equation (10) below:

$$l_\theta(y_{t^w}, r_{t^w}) = -\log p_{t^w, r_{t^w}} \tag{10}$$

where $p_{t^w, r_{t^w}}$ is generated following equation (5).

In addition to increasing the probability at the correct output node r_{t^w} as in equation (10), *Ref+Err* reduces the probability at the wrong output node y_{t^w} at the same time:

$$l_\theta(y_{t^w}, r_{t^w}) = -\log p_{t^w, r_{t^w}} + \log p_{t^w, y_{t^w}} \tag{11}$$

2.3. TWT in Beam

In the analysis above, TWT is used on a single hypothesis y. In most cases, y is the hypothesis with the highest posterior probability (i.e. the 1-best hypothesis). When applying TWT to multiple hypotheses, we propose a novel TWT in beam (TWTiB) approach.

A straightforward way to extend the TWT strategy to multiple hypotheses is to apply it for each hypothesis separately. A potential problem of this method is that the TWT operations for different hypotheses may interfere with each other. The proposed TWTiB scheme thus performs TWT on only one hypothesis in the beam. The hypothesis is chosen to be the one with the largest t^w. This way, the beginning correct segments of other hypotheses may not be influenced. Let y^b denote the chosen hypothesis and $t^{b,w}$ the time step of its first wrong token, TWTiB can be expressed as follows:

$$L(\theta)_{TWTiB} = \sum_{(y,r)\in(Y,R)} l_\theta(y^b_{t^{b,w}}, r_{t^{b,w}}) \tag{12}$$

3. EXPERIMENTAL SETUP

3.1. Data and Model

Our experiments are conducted on the Switchboard-1 Release 2 (SWBD) corpus. It contains 2,400 two-sided English conversations among 543 speakers. The total duration of the recordings sums up to 260 hours. We use the 2000 HUB5 evaluation set in our experiments. The number of utterances in this evaluation set is 4,458.

The input to the A-E2E model is the 40 dimensional log-Mel feature extracted using Kaldi [18]. The output has 49 nodes corresponding to English letters, numbers, punctuations, special transcribed notations in SWBD, and indicators including 'space', 'SOS', and 'EOS'. Note that the training samples are selected by lengths. In our experiments, utterances longer than 1800 frames are removed from the training set.

The A-E2E model in our experiments is the input-feeding listen, attend, and spell (LAS) model [3, 5]. The encoder consists of six bidirectional long short-term memory (BLSTM) layers. The number of units in each layer is 512. Different from the encoder, the decoder uses two unidirectional LSTM layers. Each layer also has 512 units.

3.2. Implementation Details

The baseline model is trained from scratch using the pointwise cross entropy loss with r_{t-1} as part of the input to the decoder, as is shown in equation (4). The learning rate is initialized to 10^{-3} and is halved when the validation loss reduction is smaller than 0.01. The model is then further trained with scheduled sampling until convergence. The WER of the baseline model is 13.3%.

For TWT on a single hypothesis, y is selected to be the 1-best hypothesis during training and is compared with r to find t^w. Note that when $M > N$, i.e. hypothesis y is shorter than reference r, the losses corresponding to $r_{N+1} : r_M$ are ignored. Using *Ref* as $l_\theta(y_t, r_t)$, the gradient at output node r_{t^w} is set to -1 manually. Due to this manual gradient assignment, the A-E2E model cannot be optimized with the simple $backward$ function in PyTorch. In our implementation, the decoder projection layer (i.e. the last hidden layer) is first detached from the A-E2E model as a separate model. This detached model maps from the 512 dimensional decoder output to the 49 dimensional softmaxed A-E2E model output. After assigning -1 to the gradient corresponding to output node r_{t^w}, the $backward$ function of the detached model is used to calculate the gradients w.r.t. the model parameters. After generating the gradients for the detached layer, the *torch.autograd* class in PyTorch is used to calculate the gradients for the rest of the A-E2E model automatically [19]. With *Ref+Err* as the loss function, in addition to assigning -1 for output node r_{t^w}, $+1$ is assigned to output node y_{t^w}.

For TWTiB, multiple hypotheses are generated using the OpenNMT package [20]. A softmax smoothing factor of 0.8 is adopted to diversify the hypothesis space [21]. During training, a heuristic posterior probability rescoring function is used on the hypotheses in the beam [22]:

$$\text{score}(\mathbf{y}, \mathbf{x}) = \log P(\mathbf{y}|\mathbf{x}) / (\frac{5 + |\mathbf{y}|}{5 + 1})^\alpha \tag{13}$$

where $\mathbf{y}$ is the output sequence, $\mathbf{x}$ is the input utterance, and $\text{score}(\mathbf{y}, \mathbf{x})$ denotes the summation of log posterior probabilities. Hyper-parameter α is selected to be 1.1 in our experiments.

Different from MWER training [10], TWT does not need regularization terms. Moreover, the TWT methods in this paper do not use any word level information. During training, the scale of the TWT loss is set to 1. In other words, -1 and $+1$ are used directly as the gradients without scaling. The optimizer is chosen to be Adam, with a learning rate of 10^{-5}. No learning rate annealing is applied in our implementation. The dropout rate is set to 0.2 and a temporary model is saved every 131,072 frames. The model used for evaluation is selected from the saved models.

During evaluation, we apply both regular decoding as well as a joint decoding method using an external LSTM LM [23, 11]. The LM is trained with the same transcriptions as the A-E2E model.

4. EVALUATION RESULTS

4.1. Loss Functions and TWT Types

Our experiments involve two loss functions and two types of TWT. The loss functions are *Ref* and *Ref+Err*. The two TWT types are the standard TWT (operating on a single hypothesis) and TWTiB. The results of different combinations of loss functions and TWT types are shown in Table 1 below.

Table 1. Loss functions and TWT types

Loss	TWT Type	WER
Ref	TWT	12.4
Ref+Err	TWT	12.6
Ref	TWTiB	12.0

We first compare the two loss functions by fixing the TWT type to the standard TWT. As is shown in Table 1, *Ref* performs better than *Ref+Err*. This may relate to the softmax function in the A-E2E model. With the softmax function, when the probability at output node r_{t^w} increases, the probability at node y_{t^w} decreases automatically.

After the comparison between the loss functions, we fix the loss function to *Ref* and compare the two TWT types. Table 1 shows that TWTiB obtains a 3.2% relative improvement over the standard TWT. Note that although TWTiB requires the generation of multiple hypotheses, it only performs TWT on a single one. In the following sections, we use TWTiB as a representative of the TWT method.

4.2. Results and Comparisons of TWTiB

The results and comparisons of TWTiB on the SWBD subset of the 2000 HUB5 evaluation set are shown in Table 2 below. Note that the number of hypotheses used in MWER and TWTiB are both 4.

Table 2. Results and comparisons of TWTiB

LM	CE	MWER [5]	TWTiB
w/o	13.3	12.2	**12.0**
w/	-	12.0	**11.7**

TWTiB outperforms the baseline CE criterion by 9.8% relatively. It is also consistently better than MWER, both with and without the external LM. Note that TWTiB does not use regularization terms or word level information, whereas MWER needs both.

In addition to the TWT based training schemes, we also conducted experiments using the weighted loss and early update method shown in equations (7) and (8), respectively. Due to the reasons mentioned in Section 2, both of them diverged during training and did not improve the baseline 13.3% A-E2E model.

4.3. Comparisons with Previously Proposed Systems

The comparisons of TWTiB with some previously proposed A-E2E systems are shown in Table 3 below. The systems without the usage of external LMs are denoted as *w/o LM*. The SWBD subset of the 2000 HUB5 evaluation set is denoted as *SWBD* and the CallHome subset *CH*.

Table 3. Comparisons with other A-E2E speech recognition systems

Systems	SWBD	CH
Seq2Seq + Trigram LM [2]	25.8	46.0
Pretraining + LSTMLM [4]	11.8	25.7
MWER w/o LM [5]	12.2	23.3
MWER + LSTMLM [11]	12.0	**23.1**
TWTiB w/o LM (proposed)	12.0	24.0
TWTiB + LSTMLM (proposed)	**11.7**	23.5

As mentioned in Section 4.2, TWTiB is better than other training schemes when evaluated on the SWBD subset. Note that the result of TWTiB on the CH subset is substantially better than that of the pretraining method [4]. The best performance on the CH subset is yielded by MWER. This may be attributed to the fact that the optimization objective of MWER is the expected WER and that word level edit distance may have better generalization ability.

5. CONCLUDING REMARKS

In this paper, we have proposed a TWT method for A-E2E speech recognition. It addresses two concerns in conventional training schemes. The first one is that decoding errors could propagate along the output sequence due to the dependency between output tokens. The second is that the enforced parameter update on correct tokens may divert the training process and lead to overfitting. TWT avoids these problems by only updating the first wrong token in the output sequence. TWT is flexible and can be combined with various loss functions. In this study, we investigate two of them: *Ref* and *Ref+Err*. Applying TWT to multiple hypotheses, we propose a novel TWTiB training scheme. Using four hypotheses, TWTiB outperforms the previous best training scheme on the SWBD evaluation subset. Future work includes applying TWTiB on large real world corpora, incorporating more advanced loss functions and word level information into TWTiB, and combining TWTiB with other training methods.

6. REFERENCES

[1] J.K. Chorowski, D. Bahdanau, D. Serdyuk, K. Cho, and Y. Bengio, "Attention-based models for speech recognition," in *Advances in neural information processing systems*, 2015, pp. 577–585.

[2] L. Lu, X. Zhang, and S. Renals, "On training the recurrent neural network encoder-decoder for large vocabulary end-to-end speech recognition," in *ICASSP*, 2016.

[3] W. Chan, N. Jaitly, Q. Le, and O. Vinyals, "Listen, attend and spell: A neural network for large vocabulary conversational speech recognition," in *Acoustics, Speech and Signal Processing (ICASSP), 2016 IEEE International Conference on*. IEEE, 2016, pp. 4960–4964.

[4] A. Zeyer, K. Irie1, R. Schluter, and H. Ney, "Improved training of end-to-end attention models for speech recognition," in *Interspeech*, 2018.

[5] C. Weng, J. Cui, G. Wang, J. Wang, C. Yu, D. Su, and D. Yu, "Improving attention based sequence-to-sequence models for end-to-end english conversational speech recognition," in *Interspeech 2018*, 2018.

[6] D. Bahdanau, D. Serdyuk, P. Brakel, N. R. Ke, J. Chorowski, A. Courville, and Y. Bengio, "Task loss estimation for sequence prediction," *arXiv preprint arXiv:1511.06456*, 2015.

[7] S. Karita, A. Ogawa, M. Delcroix, and T. Nakatani, "Sequence training of encoder-decoder model using policy gradient for end-to-end speech recognition," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 5839–5843.

[8] J. Schulman, N. Heess, T. Weber, and P. Abbeel, "Gradient estimation using stochastic computation graphs," in *Advances in Neural Information Processing Systems*, 2015, pp. 3528–3536.

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

[10] R. Prabhavalkar, T.N. Sainath, Y. Wu, P. Nguyen, Z. Chen, C.C. Chiu, and A. Kannan, "Minimum word error rate training for attention-based sequence-to-sequence models," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 4839–4843.

[11] J. Cui, C. Weng, G. Wang, J. Wang, P. Wang, C. Yu, D. Su, and D. Yu, "Improving attention-based end-to-end asr systems with sequence-based loss functions," in *SLT 2018*, 2018.

[12] R. Sennrich, B. Haddow, and A. Birch, "Neural machine translation of rare words with subword units," in *Proc. of ACL*, 2016, pp. 1715–1725.

[13] J. Li, G. Ye, A. Das, R. Zhao, and Y. Gong, "Advancing acoustic-to-word ctc model," *arXiv preprint arXiv:1803.05566*, 2018.

[14] M. Collins and B. Roark, "Incremental parsing with the perceptron algorithm," in *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*. Association for Computational Linguistics, 2004, p. 111.

[15] P. Liang, A. Bouchard-Côté, D. Klein, and B. Taskar, "An end-to-end discriminative approach to machine translation," in *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, 2006, pp. 761–768.

[16] P. Lison and G.J.M. Kruijff, "An integrated approach to robust processing of situated spoken dialogue," in *Proceedings of the 2nd Workshop on Semantic Representation of Spoken Language*. Association for Computational Linguistics, 2009, pp. 58–65.

[17] S. Bengio, O. Vinyals, N. Jaitly, and N. Shazeer, "Scheduled sampling for sequence prediction with recurrent neural networks," in *Advances in Neural Information Processing Systems*, 2015, pp. 1171–1179.

[18] D. Povey, *Discriminative Training for Large Vocabulary Speech Recognition*, Ph.D. thesis, Cambridge University Engineering Department, 2003.

[19] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in pytorch," 2017.

[20] G. Klein, Y. Kim, Y. Deng, J. Senellart, and A.M. Rush, "Opennmt: Open-source toolkit for neural machine translation," in *Proc. ACL*, 2017.

[21] C. Shan, J. Zhang, Y.Wang, and L. Xie, "Attention-based end-to-end speech recognition in mandarin," *CoRR*, vol. abs/1707.07167, 2017.

[22] Y. Wu and et. al M. Schuster, "Google's neural machine translation system: Bridging the gap between human and machine translation," *CoRR*, vol. 1609.08144, 2016.

[23] T. Hori, S. Watanabe, and J. R. Hershey, "Joint ctc/attention decoding for end-to-end speech recognition," in *ACL*, 2017.