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

Multi-Task Learning for End-to-End ASR Word and Utterance Confidence with Deletion Prediction

David Qiu[1], *Yanzhang He*[1], *Qiujia Li*[2*], *Yu Zhang*[1], *Liangliang Cao*[1], *Ian McGraw*[1]
[1]Google, LLC, USA
[2]University of Cambridge, UK
{qdavid, yanzhanghe, ngyuzh, llcao, imcgraw}@google.com, ql264@cam.ac.uk

Abstract

Confidence scores are very useful for downstream applications of automatic speech recognition (ASR) systems. Recent works have proposed using neural networks to learn word or utterance confidence scores for end-to-end ASR. In those studies, word confidence by itself does not model deletions, and utterance confidence does not take advantage of word-level training signals. This paper proposes to jointly learn word confidence, word deletion, and utterance confidence. Empirical results show that multi-task learning with all three objectives improves confidence metrics (NCE, AUC, RMSE) without the need for increasing the model size of the confidence estimation module. Using the utterance-level confidence for rescoring also decreases the word error rates on Google's Voice Search and Long-tail Maps datasets by 3-5% relative, without needing a dedicated neural rescorer.

Index Terms: automatic speech recognition, confidence estimation, deletion prediction, multi-task learning, Transformer

1. Introduction

Confidence scores are often used to assess the reliability of speech recognizers [1, 2]. They have been widely used for various downstream tasks, including semi-supervised and active learning [3, 4, 5, 6, 7, 8], system combination [9, 10, 11], dialog systems [12, 13, 14] and keyword spotting [15, 16, 17]. For traditional hidden Markov model (HMM)-based automatic speech recognition (ASR) systems, the simplest approach is to use the posterior probability for each hypothesized word. More advanced methods have been proposed to better estimate word-level confidence, including linear models [11, 18], conditional random fields [19], and neural networks [20, 21, 22, 23, 24].

However, for end-to-end (E2E) ASR models such as recurrent neural network transducers (RNN-T) and attention-based sequence-to-sequence models, word posteriors cannot be approximated well from the tree-like "lattice" [25] where the prediction of each token conditions on the full history of previous tokens. Autoregressive decoders also tend to be overconfident [26]. To solve this challenge, several model-based methods have been proposed to estimate word and utterance-level confidence for E2E models. For examples, [27, 26] proposed to train a token-level (e.g. graphemes or word-pieces [28]) confidence estimation module (CEM) on top of a given E2E model and the word-level confidence can be simply obtained by averaging the token-level scores. Since word-level confidence is more commonly used for applications such as keyword spotting [15, 16, 17] and systems combination [9, 10], [29] proposed to train CEMs for word-level confidence directly, where issues such as multiple possible word-piece tokenizations are addressed. Utterance-level confidence scores are also important for applications such as data selection [4, 6, 7]. One can directly average the word-level confidence scores for utterance-level confidence, but averaging is sub-optimal since deletion errors are not accounted for. [30, 31] proposed to directly learn utterance-level confidence scores for E2E systems and [31] showed that rescoring n-best hypotheses based on utterance-level confidence estimator can also reduce word error rates (WER). Previous works generally model confidence scores at a single level (*i.e.* token, word or utterance). This paper explores joint optimization of word-level and utterance-level confidence estimation in a single model with multi-task learning.

Model-based approaches typically formulate the confidence estimation problem as a binary classification task [21, 22, 23, 24, 27, 26, 29, 30, 31], where correct tokens, words, or utterances should have confidence scores close to 1, and 0 otherwise. For word-level confidence estimation, scores between 1 and 0 are only assigned to words that appear in the hypotheses. This covers substitution and insertion errors for ASR. However, deletion errors are either ignored or not explicitly addressed. In other words, averaging the confidence scores yields an estimate for the word correct ratio (WCR), but deletion errors need to be considered for estimating WER. As shown in [32, 23], deletion estimation is also important for downstream tasks. For example, even with a good confidence estimator, utterances with high confidence may have high WER because of deletion errors. Tasks such as semi-supervised training can be adversely affected without explicit modeling of deletion errors [23]. In this work, we explore estimating deletion errors along with word-level and utterance-level confidence scores in a single model.

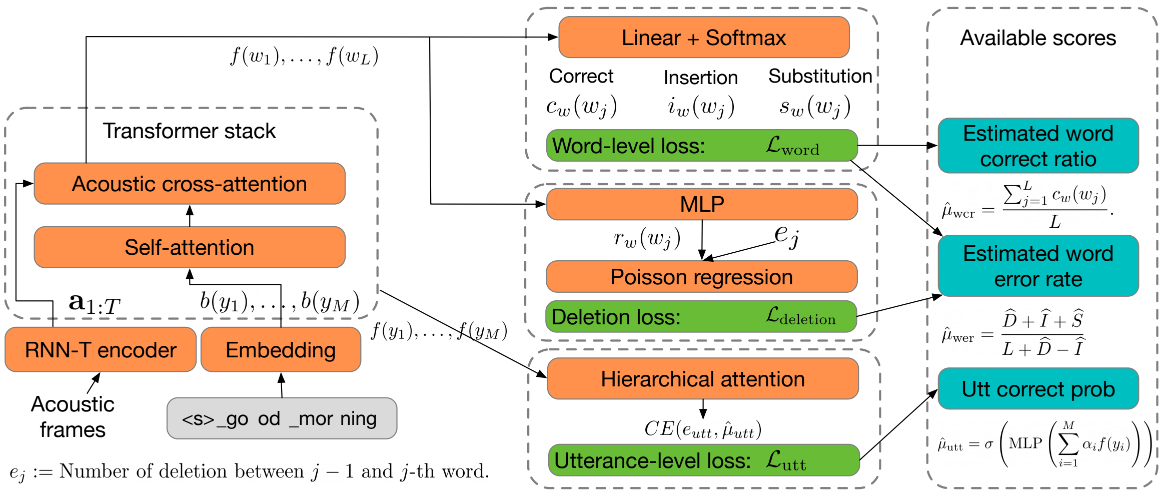
This paper makes the following contributions: 1) Analyzes the word-level CEM in [29] to show that it mitigates the over-confidence problem when subword deletions are present. 2) Proposes using neural networks and Poisson regression to learn arbitrary deletion length, and integrate it with the word-level CEM to estimate WER. 3) Demonstrates that jointly learning the word confidence, utterance confidence, and deletion length prediction tasks improves all reported confidence metrics over single-task word-level or utterance-level CEMs. 4) Shows that using the utterance-level confidence to rescore RNN-T hypotheses reduces WERs on industry-scale datasets.

2. Model

2.1. Notation

In general, an E2E ASR model processes a sequence of acoustic features $\mathbf{a}$, and outputs a token sequence $[y_1, \ldots, y_M]$ as its hypothesis. Popular choices for the set of tokens include graphemes and word-pieces (WP). In this paper, we focus on WP. The goal of the confidence model is to use both $\mathbf{a}$, and the embedding of the WP sequence, $\mathbf{b}$, as inputs, and output a score

*Work was done while the author interned at Google.



$f(w_1), \dots, f(w_L)$
Transformer stack
Acoustic cross-attention
Self-attention
$\mathbf{a}_{1:T}$
$b(y_1), \dots, b(y_M)$
RNN-T encoder
Embedding
Acoustic frames
<s>_go od _mor ning
$f(y_1), \dots, f(y_M)$
$e_j :=$ Number of deletion between $j-1$ and j-th word.
Linear + Softmax
Correct
Insertion
Substitution
$c_w(w_j)$
$i_w(w_j)$
$s_w(w_j)$
Word-level loss: $\mathcal{L}_{\text{word}}$
MLP
$r_w(w_j)$
e_j
Poisson regression
Deletion loss: $\mathcal{L}_{\text{deletion}}$
Hierarchical attention
$CE(e_{utt}, \hat{\mu}_{utt})$
Utterance-level loss: $\mathcal{L}_{\text{utt}}$
Available scores
Estimated word correct ratio
$\hat{\mu}_{\text{wcr}} = \frac{\sum_{j=1}^{L} c_w(w_j)}{L}$.
Estimated word error rate
$\hat{\mu}_{\text{wer}} = \frac{\widehat{D} + \widehat{I} + \widehat{S}}{L + \widehat{D} - \widehat{I}}$
Utt correct prob
$\hat{\mu}_{\text{utt}} = \sigma\left(\text{MLP}\left(\sum_{i=1}^{M} \alpha_i f(y_i)\right)\right)$

Figure 1: *Model architecture. $f(\cdot)$ denotes token/word level features. c_w, i_w, s_w denote the estimated correction, insertion and substitution. $\mathcal{L}_{deletion}$ refers to Poisson regression for estimating deletion errors. More details in Section 2.*

for every word and every utterance. To distinguish between WP and words, we use M and L to refer to the number of WPs and words in a hypothesis, respectively. We use y_i and w_j to denote the i-th WP and j-th word, respectively.

We use the Transformer [33, 34] as the base feature extractor for all of the proposed tasks:

$$[f(y_1), \dots, f(y_M)] = \text{Transformer}(\text{CA}(\mathbf{a}), \text{SA}(\mathbf{b})), \quad (1)$$

where the cross-attention block CA attends to $\mathbf{a}$, and the self-attention block SA attends to $\mathbf{b}$. We refer to the Transformer's output at the i-th token as $f(y_i)$.

2.2. Baseline Word-Level Confidence Estimation

For the baseline CEM, we use a multi-layer perceptron (MLP) to process the Transformer's output f into 3-D outputs:

$$[c(y_i), i(y_i), s(y_i)] = \text{Softmax}(\text{MLP}(f(y_i))). \quad (2)$$

For training, we adapt the word-level training loss introduced in [29]. To recap, the loss gathers the output of (2), but only at the indices corresponding to the last WP of each word (see Table 1). The gathered output can be re-indexed as $[c_w(w_1), \dots, c_w(w_L)]$, $[i_w(w_1), \dots, i_w(w_L)]$, and $[s_w(w_1), \dots, s_w(w_L)]$, to denote the probabilities that the words are tagged by Levenshtein edit distance [35] as "correct", "insertion", and "substitution", respectively. Then, over every word, we compute the cross-entropy loss $\mathcal{L}_{\text{word}}$ between the ground truth tags and the outputs:

$$-\sum_{j=1}^{L}\left[\mathbb{1}_j^{\text{cor}} \log c_w(w_j) + \mathbb{1}_j^{\text{ins}} \log i_w(w_j) + \mathbb{1}_j^{\text{sub}} \log s_w(w_j)\right]. \quad (3)$$

This also yields a simple estimate for the WCR:

$$\hat{\mu}_{\text{wcr}} = \frac{\sum_{j=1}^{L} c_w(w_j)}{L}. \quad (4)$$

Although word-level deletions are not modeled, the sub-word/WP deletion problem is implicitly addressed by the loss. Table 1 shows an example where the first word in the hypothesis contains a WP deletion. WP edit distance completely ignores any WP deletions because deletions are not part of the hypothesis. On the other hand, because the word-level CEM is trained with the word-level edit distance serving as the ground truth, hypothesized words that contain WP deletions are guaranteed to be labeled as word-level substitutions. This resolves an overconfidence problem that is common in token-level CEMs trained to model WP edit distance, especially for a smaller set of tokens such as graphemes.

Table 1: *Top: example of the WP edit distance outputting "correct" for a WP when the whole word is actually a "substitution" error. Bottom: example of computing the word-level loss at the last WP of each word.*

Hyp:	_go		_morn	ing
Ref:	_go	od	_morn	ing
WP edit:	*cor*	*del*	*cor*	*cor*
Word edit:	*sub*	–	–	*cor*
$\mathcal{L}_{\text{word}}(w_j)$:	$-\log s_{\text{w}}(w_1)$	–	–	$-\log c_{\text{w}}(w_2)$

2.3. Deletion Length Prediction

To estimate the WER, the model needs to estimate the number of deletions as a secondary task. Here, we define a sequence of deletion length random variables for $1 \le j \le L+1$ as:

$$E_j = \begin{cases} \#(\text{del}) \text{ before the first word}, & \text{if } j = 1 \\ \#(\text{del}) \text{ after the last word}, & \text{if } j = L+1 \\ \#(\text{del}) \text{ between } j-1 \text{ and } j\text{-th word}, & \text{otherwise.} \end{cases}$$

The number of deletions can be any non-negative integer. Existing works such as [23] converts the problem into binary prediction by clipping E_j at 1. In contrast, we predict E_j directly by modeling $P(E_j | f(w_j); \theta)$ to follow the Poisson distribution, which is well-suited for count data. We use a MLP to further process the features from the transformer and predict the mean of E_j as below:

$$\mathbb{E}(E_j | f(w_j); \theta) = \exp(r_w(w_j)) \quad (5)$$
$$r_w(w_j) = \text{MLP}(f(w_j); \theta) \quad (6)$$

Let e_j be the ground truth realization of E_j. With maximum likelihood estimation for Poisson regression, the training loss for deletions is given by:

$$\mathcal{L}_{\text{deletion}} = -\sum_{j=1}^{L+1} \left[e_j r_w(w_j) - \exp\left(r_w(w_j)\right)\right]. \quad (7)$$

During inference, the WER estimate can be computed as

$$\hat{\mu}_{\text{wer}} = \frac{\widehat{D} + \widehat{I} + \widehat{S}}{L + \widehat{D} - \widehat{I}},$$

where $\widehat{D} = \sum_{j=1}^{L+1} \exp(r_{\text{w}}(w_j))$, $\widehat{I} = \sum_{j=1}^{L} i_{\text{w}}(w_j)$, and $\widehat{S} = \sum_{j=1}^{L} s_{\text{w}}(w_j)$.

2.4. Utterance-Level Confidence Estimation

A coarse way of including deletion information is to define a tertiary utterance-level prediction task, with the ground truth

$$e_{\text{utt}} = \begin{cases} 1, & \text{if utterance WER} = 0 \\ 0, & \text{otherwise.} \end{cases}$$

The presence of any deletions causes the value of e to be 0, and this signal is backpropagated to the internal features in the Transformer feature extractor to help the word-level confidence task. This objective was proposed in [30], albeit in a single-task setup that ignores word confidences. Intuitively, an utterance feature is a summary of the sequence of WP features. Thus, to extract the utterance features and make the prediction, we use the hierarchical attention proposed in [36]:

$$u_i = \tanh\left(W_1 f(y_i) + b\right) \quad (8)$$

$$\alpha_i = \frac{\exp\left(w_2^T u_i\right)}{\sum_{m=1}^{M} \exp\left(w_2^T u_m\right)} \quad (9)$$

$$\hat{\mu}_{\text{utt}} = \sigma\left(\text{MLP}\left(\sum_{i=1}^{M} \alpha_i f(y_i)\right)\right). \quad (10)$$

The parameters W_1, b, and w_2 that generate $\hat{\mu}_{\text{utt}}$ can be trained with the binary cross-entropy loss:

$$\mathcal{L}_{\text{utt}} = -[e_{\text{utt}} \log \hat{\mu}_{\text{utt}} + (1 - e_{\text{utt}}) \log(1 - \hat{\mu}_{\text{utt}})]. \quad (11)$$

$\hat{\mu}_{\text{utt}}$ provides an estimate of the probability that the entire utterance is recognized with zero WER, which is useful for ranking utterances in the order of their confidences.

2.5. Multi-task Objective

The model (Figure 1) is trained by combining (3), (7), and (11):

$$\mathcal{L}_{\text{total}} = \frac{1}{L}\mathcal{L}_{\text{word}} + \frac{\lambda_{\text{deletion}}}{L+1}\mathcal{L}_{\text{deletion}} + \lambda_{\text{utt}}\mathcal{L}_{\text{utt}}.$$

In this paper, $\lambda_{\text{deletion}} = 0.5$, $\lambda_{\text{utt}} = 1$, when the respective losses are active, and $\mathcal{L}_{\text{total}}$ is further normalized over the mini-batch.

3. Experimental Setup

3.1. Model Architecture

The ASR's architecture is a RNN-T that follows [37]. It has 8 LSTM layers in the encoder and 2 LSTM layers in the prediction network. Each LSTM layer is unidirectional, with 2,048 units and a projection layer with 640 units. A time-reduction layer is added between 2nd and 3rd encoder layers, which halves the sequence length by concatenating every two input frames. The RNN-T is frozen during CEM training.

Table 2: *Training objectives for five proposed models.*

Model	Model Size	Loss		
		$\mathcal{L}_{\text{word}}$	$\mathcal{L}_{\text{deletion}}$	$\mathcal{L}_{\text{utt}}$
`W`	16.6M	✓	✗	✗
`U`	17.0M	✗	✗	✓
`WD`	16.9M	✓	✓	✗
`WU`	17.2M	✓	✗	✓
`WUD`	17.5M	✓	✓	✓

We report the performance on the five combinations of confidence training tasks shown in Table 2. The feature extractor in (1) consists of two Transformer decoder blocks with input embedding and internal dimensions of 640. For $\mathcal{L}_{\text{deletion}}$, the MLP in (6) has three layers, with hidden and output dimensions of 320, 160, and 1. For $\mathcal{L}_{\text{utt}}$, we use the hierarchical attention mechanism in (8) and (9) to generate the utterance-level feature. The MLP in (10) has two layers, with hidden and output dimensions of 320 and 1, respectively. All training is performed in TensorFlow with the Lingvo [38] toolkit, using four hypotheses from the RNN-T. The optimizer is Adam [39] with learning rate 0.001, and the global batch size is 4,096 across 8×8 TPU.

3.2. Dataset

The models are trained on the multi-domain training set used in [37], which spans search, farfield, telephony and YouTube. All datasets are anonymized and hand-transcribed. The transcription for YouTube utterances is done in a semi-supervised fashion [40]. Multi-condition training (MTR) [41] and random data downsampling to 8kHz [42] are applied to increase data diversity.

The main test set includes ~14K *Voice Search* (VS) utterances extracted from Google traffic, which is anonymized and hand-transcribed. To test the generalization of the CEM, we create a list of proper nouns identified by an in-house named entity tagger that are common in the LM training data for conventional ASR but rare in the audio-text paired training data for E2E ASR models. We select 10K sentences from the LM test data for the Maps domain, such that each sentence contains at least one of these proper nouns, then synthesize the TTS audio (as in [43]) for them to create the *Long-tail Maps* test set.

3.3. Evaluation Metrics

All evaluation metrics are reported on the top hypothesis. For word confidence, we adopt the same metrics in [29]: normalized cross-entropy (NCE) [44], area under the receiver operating characteristic curve (AUC-ROC) and the *negative* class precision recall curve (AUC-PR). Higher is better for these metrics.

To measure the ranking quality of the utterance confidence, we use the utterance-level AUC-ROC and AUC-PR (higher is better), where the positive / negative classes are the sets of utterances with zero / non-zero WER, respectively. To measure the accuracy of estimating WERs, we use the root mean squared error (RMSE, lower is better) between the utterance confidence and the ground truth $(1 - \text{WER})$.

Table 4 shows the availability of utterance-level confidence scores, and which one is preferred when multiple are available. Recall that $\hat{\mu}_{\text{utt}}$ models the probability that the utterance is recognized with zero WER. Thus, we prioritize using it for the utterance ranking tasks (AUC-ROC, AUC-PR). However, $\hat{\mu}_{\text{utt}}$ does not attempt to estimate the WER, and we de-prioritize it for the WER accuracy task (RMSE).

Table 3: *Confidence metrics on Voice Search and a long-tail Maps dataset for all proposed CEMs.*

	Voice Search						Long-tail Maps					
	Word-level			Utterance-level			Word-level			Utterance-level		
Model	NCE	AUC ROC	AUC PR	AUC ROC	AUC PR	(1-WER) RMSE	NCE	AUC ROC	AUC PR	AUC ROC	AUC PR	(1-WER) RMSE
`W`	0.348	0.927	0.451	0.765	0.428	0.226	0.285	0.875	0.494	0.704	0.549	0.301
`U`	–	–	–	0.788	0.515	–	–	–	–	0.721	0.593	–
`WD`	0.350	0.928	0.459	0.767	0.440	0.216	0.291	0.875	0.504	0.704	0.558	0.296
`WU`	**0.378**	**0.931**	**0.489**	**0.810**	0.549	0.223	**0.304**	**0.883**	**0.524**	**0.755**	**0.636**	0.296
`WUD`	0.365	0.929	0.476	**0.810**	**0.552**	**0.213**	**0.304**	0.882	0.516	0.746	0.634	**0.292**

Table 4: *Utterance confidence scores' priorities by metrics.*

Model	Available Scores	AUC-ROC & PR	RMSE
`W`	$\hat{\mu}_{\text{wcr}}$	$\hat{\mu}_{\text{wcr}}$	$\hat{\mu}_{\text{wcr}}$
`U`	$\hat{\mu}_{\text{utt}}$	$\hat{\mu}_{\text{utt}}$	N/A
`WD`	$\hat{\mu}_{\text{wcr}}$, $\hat{\mu}_{\text{wer}}$	$1 - \hat{\mu}_{\text{wer}}$	$1 - \hat{\mu}_{\text{wer}}$
`WU`	$\hat{\mu}_{\text{wcr}}$, $\hat{\mu}_{\text{utt}}$	$\hat{\mu}_{\text{utt}}$	$\hat{\mu}_{\text{wcr}}$
`WUD`	$\hat{\mu}_{\text{wcr}}$, $\hat{\mu}_{\text{utt}}$, $\hat{\mu}_{\text{wer}}$	$\hat{\mu}_{\text{utt}}$	$1 - \hat{\mu}_{\text{wer}}$

4. Results

4.1. Joint Training Confidence Metrics

This section discusses the performance of all models on the confidence metrics. In Table 3, comparing Models `WU`, `WD`, and `WUD` vs Model `W` shows that any joint training (especially with the utterance-level loss) helps improve word-level metrics. Comparing Models `WU` and `WUD` vs Model `U` shows that joint training also helps improve the performance of $\hat{\mu}_{\text{utt}}$ on utterance-level AUC-ROC. Comparing Models `WD` and `WUD` vs Model `W` shows that modeling deletion and insertion rates helps to achieve the most accurate WER estimate in terms of RMSE. Since all five models have similar number of parameters (see Table 2), it is clear that the three tasks do not significantly increase the required model capacity, and their training signals complement each other to improve overall confidence accuracy.

4.2. Qualitative Example

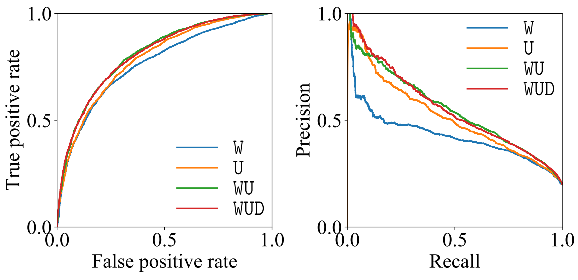
To visualize the benefit of adding $\mathcal{L}_{\text{deletion}}$ and $\mathcal{L}_{\text{utt}}$, Table 5 shows an example utterance that contains a deletion. Model `WUD`, through learning the utterance correctness and deletion lengths at every position, is able to use that knowledge to aid word confidence predictions. That is seen from the lower confidence score at the word "absolut", which follows a deletion.

Table 5: *The CEM output $c_w(w_j)$ on an example TTS utterance, for Models `W` and `WUD`.*

Hyp:		absolut	green	philadelphia	pa
Ref:	upsal	at	greene	philadelphia	pa
Word edit:	*del*	*sub*	*sub*	*cor*	*cor*
`W:`	–	0.62	0.95	0.99	0.99
`WUD:`	–	0.24	0.92	0.99	0.88

4.3. Rescoring Using Utterance Confidence

Figure 2 shows the improved utterance-level ROC and PR curves for the multi-task Model `WU` over single-task Models `W` and `U`. This improvement in utterance-level confidence scores can serve as an alternative signal to determine which RNN-T beam search hypothesis has the lowest WER among alternative hypotheses in the beam. For every utterance, we use confidence to rescore the top four RNN-T hypotheses. In Table 6, we report the average WER on the top hypothesis after rescoring. On Voice Search, both the $\hat{\mu}_{\text{wcr}}$ and $\hat{\mu}_{\text{utt}}$ outputs of Model `WU` reduce the Voice Search WER from 6.4 to 6.1, which shows the advantage of joint training. On the long-tail Maps set, $\hat{\mu}_{\text{wcr}}$ achieves the lowest WER of 13.6, which suggests that word-level confidence shows better generalization to previously unseen data.



(a) *ROC cuves.* (b) *PR curves.*

Figure 2: *Utterance-level ROC and PR curves on Voice Search for Models `W`, `U`, `WU`, `WUD`.*

Table 6: *Rescoring WERs (%).*

Model	Rank	Voice Search	Long-tail Maps
RNN-T	Beam search	6.4	14.0
`W`	$\hat{\mu}_{\text{wcr}}$	6.2	13.8
`U`	$\hat{\mu}_{\text{utt}}$	6.3	14.4
`WU`	$\hat{\mu}_{\text{wcr}}$	**6.1**	**13.6**
`WU`	$\hat{\mu}_{\text{utt}}$	**6.1**	14.1
`WUD`	$\hat{\mu}_{\text{wcr}}$	6.2	**13.6**
`WUD`	$\hat{\mu}_{\text{utt}}$	6.2	14.1

5. Conclusions

In this paper, we propose to extend the word-level CEM to jointly learn the length of deletions between every pair of words and the probability that the entire utterance is recognized correctly. Experimental results show that the proposed multi-task learning setup improves word and utterance-level confidence metrics without incurring significant model size increase. The utterance confidence can also be used to rescore first-pass RNN-T hypotheses and reduce WER by 3-5% relative.

6. References

[1] F. Wessel, R. Schluter, K. Macherey, and H. Ney, "Confidence measures for large vocabulary continuous speech recognition," *IEEE Trans. on Speech and Audio Processing*, 2001.

[2] H. Jiang, "Confidence measures for speech recognition: A survey," *Speech Communication*, 2005.

[3] L. Wang, M. J. Gales, and P. C. Woodland, "Unsupervised training for mandarin broadcast news and conversation transcription," in *ICASSP*, 2007.

[4] K. Yu, M. Gales, L. Wang, and P. C. Woodland, "Unsupervised training and directed manual transcription for LVCSR," *Speech Communication*, 2010.

[5] D. Yu, B. Varadarajan, L. Deng, and A. Acero, "Active learning and semi-supervised learning for speech recognition: A unified framework using the global entropy reduction maximization criterion," *Computer Speech & Language*, 2010.

[6] Y. Huang, D. Yu, Y. Gong, and C. Liu, "Semi-supervised GMM and DNN acoustic model training with multi-system combination and confidence re-calibration." in *Interspeech*, 2013.

[7] D. Park, Y. Zhang, Y. Jia, W. Han, C.-C. Chiu *et al.*, "Improved noisy student training for automatic speech recognition," in *Interspeech*, 2020.

[8] M. Li and I. K. Sethi, "Confidence-based active learning," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 2006.

[9] J. G. Fiscus, "A post-processing system to yield reduced word error rates: Recognizer output voting error reduction (ROVER)," in *ASRU*, 1997.

[10] G. Evermann and P. C. Woodland, "Posterior probability decoding, confidence estimation and system combination," in *NIST Speech Transcription Workshop*, 2000.

[11] G. Evermann and P. C. Woodland, "Large vocabulary decoding and confidence estimation using word posterior probabilities," in *ICASSP*, 2000.

[12] D. Hakkani-Tur, G. Tur, M. Rahim, and G. Riccardi, "Unsupervised and active learning in automatic speech recognition for call classification," in *ICASSP*, 2004.

[13] G. Riccardi and D. Hakkani-Tur, "Active learning: Theory and applications to automatic speech recognition," *IEEE Trans. on Speech and Audio Processing*, 2005.

[14] G. Tur, D. Hakkani-Tür, and R. E. Schapire, "Combining active and semi-supervised learning for spoken language understanding," *Speech Communication*, 2005.

[15] Y. Benayed, D. Fohr, J. P. Haton, and G. Chollet, "Confidence measures for keyword spotting using support vector machines," in *ICASSP*, 2003.

[16] J. Keshet, D. Grangier, and S. Bengio, "Discriminative keyword spotting," *Speech Communication*, 2009.

[17] M. S. Seigel, P. C. Woodland, and M. Gales, "A confidence-based approach for improving keyword hypothesis scores," in *ICASSP*, 2013.

[18] L. Gillick, Y. Ito, and J. Young, "A probabilistic approach to confidence estimation and evaluation," in *ICASSP*, 1997.

[19] M. S. Seigel and P. C. Woodland, "Combining information sources for confidence estimation with CRF models," in *Interspeech*, 2011.

[20] T. Schaaf and T. Kemp, "Confidence measures for spontaneous speech recognition," in *ICASSP*, 1997.

[21] K. Kalgaonkar, C. Liu, Y. Gong, and K. Yao, "Estimating confidence scores on ASR results using recurrent neural networks," in *ICASSP*, 2015.

[22] M. A. Del-Agua, A. Gimenez, A. Sanchis, J. Civera, and A. Juan, "Speaker-adapted confidence measures for ASR using deep bidirectional recurrent neural networks," *IEEE/ACM Trans. on Audio, Speech, and Language Processing*, 2018.

[23] A. Ragni, Q. Li, M. J. F. Gales, and Y. Wang, "Confidence estimation and deletion prediction using bidirectional recurrent neural networks," in *SLT*, 2018.

[24] Q. Li, P. Ness, A. Ragni, and M. J. F. Gales, "Bi-directional lattice recurrent neural networks for confidence estimation," in *ICASSP*, 2019.

[25] T. Kemp and T. Schaaf, "Estimating confidence using word lattices," in *Eurospeech*, 1997.

[26] Q. Li, D. Qiu, Y. Zhang, B. Li, Y. He *et al.*, "Confidence estimation for attention-based sequence-to-sequence models for speech recognition," in *ICASSP*, 2021.

[27] A. Woodward, C. Bonnín, I. Masuda, D. Varas, E. Bou-Balust, and J. C. Riveiro, "Confidence measures in encoder-decoder models for speech recognition," in *Interspeech*, 2020.

[28] M. Schuster and K. Nakajima, "Japanese and korean voice search," in *ICASSP*, 2012.

[29] D. Qiu, Q. Li, Y. He, Y. Zhang, B. Li, L. Cao, R. Prabhavalkar, D. Bhatia, W. Li, K. Hu *et al.*, "Learning word-level confidence for subword end-to-end ASR," in *ICASSP*, 2021.

[30] A. Kumar, S. Singh, D. Gowda, A. Garg, S. Singh, and C. Kim, "Utterance confidence measure for end-to-end speech recognition with applications to distributed speech recognition scenarios," in *Interspeech*, 2020.

[31] Q. Li, Y. Zhang, B. Li, L. Cao, and P. C. Woodland, "Residual energy-based models for end-to-end speech recognition," in *Interspeech*, 2021.

[32] M. S. Seigel and P. C. Woodland, "Detecting deletions in ASR output," *ICASSP*, 2014.

[33] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones *et al.*, "Attention is all you need," in *NeurIPS*, 2017.

[34] W. Li, J. Qin, C.-C. Chiu, R. Pang, and Y. He, "Parallel rescoring with transformer for streaming on-device speech recognition," in *Interspeech*, 2020.

[35] V. I. Levenshtein, "Binary codes capable of correcting deletions, insertions, and reversals," in *Soviet physics doklady*, vol. 10, no. 8. Soviet Union, 1966, pp. 707–710.

[36] Z. Yang, D. Yang, C. Dyer, X. He, A. Smola, and E. Hovy, "Hierarchical attention networks for document classification," in *NAACL*, 2016.

[37] T. Sainath, Y. He, B. Li, A. Narayanan, R. Pang *et al.*, "A streaming on-device end-to-end model surpassing server-side conventional model quality and latency," in *ICASSP*, 2020.

[38] J. Shen, P. Nguyen, Y. Wu, Z. Chen, M. X. Chen *et al.*, "Lingvo: A modular and scalable framework for sequence-to-sequence modeling," *arXiv:1902.08295*, 2019.

[39] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *ICLR*, 2015.

[40] H. Liao, E. McDermott, and A. Senior, "Large scale deep neural network acoustic modeling with semi-supervised training data for YouTube video transcription," in *ASRU*, 2013.

[41] C. Kim, A. Misra, K. K. Chin, T. Hughes, A. Narayanan *et al.*, "Generation of large-scale simulated utterances in virtual rooms to train deep-neural networks for far-field speech recognition in Google Home," in *Interspeech*, 2017.

[42] J. Li, D. Yu, J.-T. Huang, and Y. Gong, "Improving wideband speech recognition using mixed-bandwidth training data in CD-DNN-HMM," in *SLT*, 2012.

[43] X. Gonzalvo, S. Tazari, C.-a. Chan, M. Becker, A. Gutkin *et al.*, "Recent advances in Google real-time HMM-driven unit selection synthesizer," in *Interspeech*, 2016.

[44] M. Siu, H. Gish, and F. Richardson, "Improved estimation, evaluation and applications of confidence measures for speech recognition," in *Eurospeech*, 1997.