IMPROVING THE FUSION OF ACOUSTIC AND TEXT REPRESENTATIONS IN RNN-T

Chao Zhang, Bo Li, Zhiyun Lu, Tara N. Sainath and Shuo-yiin Chang

Google LLC, USA
{chaoz, boboli, zhiyunlu, tsainath, shuoyiin}@google.com

ABSTRACT

The recurrent neural network transducer (RNN-T) has recently become the mainstream end-to-end approach for streaming automatic speech recognition (ASR). To estimate the output distributions over subword units, RNN-T uses a fully connected layer as the joint network to fuse the acoustic representations extracted using the acoustic encoder with the text representations obtained using the prediction network based on the previous subword units. In this paper, we propose to use gating, bilinear pooling, and a combination of them in the joint network to produce more expressive representations to feed into the output layer. A regularisation method is also proposed to enable better acoustic encoder training by reducing the gradients back-propagated into the prediction network at the beginning of RNN-T training. Experimental results on a multilingual ASR setting for voice search over nine languages show that the joint use of the proposed methods can result in 4%–5% relative word error rate reductions with only a few million extra parameters.

Index Terms— ASR, RNN-T, fusion, gating, bilinear pooling

1. INTRODUCTION

In contrast to the traditional modular-based ASR system that consists of an acoustic model, a language model (LM) and a rule-based decoder, the recent end-to-end (E2E) approach aims at implementing the ASR process using a single neural network model. In particular, both attention-based encoder-decoder [1–4] and recurrent neural network transducer (RNN-T) [5–10] methods achieve a coherently integrate acoustic and text information using a recurrent structure between the previous and current subword units in the output text sequence in contrast to modular-based ASR systems. Recently, RNN-T has become more prevalent due to its streaming benefits [11–17].

RNN-T was first proposed to extend a connectionist temporal classification (CTC) *acoustic encoder* [18] with a *prediction network* serving as an LM. Acoustic and text representations over phonemes are derived separately from the acoustic encoder and prediction network, and fused using an addition followed by a softmax function to produce the final output distributions [5]. Shortly afterward, an improvement was introduced to fuse more compressed hidden representations using concatenation and a fully connected (FC) layer with a hyperbolic tangent (tanh) function [6], which is termed as the *joint network*. The fused representations are further transformed by the output layer, which is another FC layer with the softmax function, into the output distributions. Since then, there has been a research focus to improve the RNN-T encoder structure, from using the long short-term memory (LSTM) model [19, 20] to Transformer and Conformer [14, 21], and to improve their streaming performance and on-device efficiency [11, 22, 23]. More recently, many studies focus on improving the recognition accuracy on long tail words/phrases and long-form utterances, which results in the use of extra model components [15, 24], novel prediction network structures and decoding algorithms [7, 25, 26], alternative subword units to output [11, 20], test-time external language model integration [27–29], and synthetic data augmentation and knowledge distillation methods [11, 30–32].

Though joint network is the component closest to the output layer and is important to RNN-T performance, there are only a few studies related to it [6, 28, 33]. In this paper, by viewing the function of the joint network as to fuse the representations of the acoustic and text modalities, we propose to improve the joint network implementation using different structures for information fusion, including gating and a low-rank approximation of bilinear pooling with shortcut connections and a tanh transform. A better-performing structure is further proposed by stacking bilinear pooling on top of gating. Furthermore, since text priors are often easier to learn than complex acoustic patterns, the prediction network often converges much faster than the acoustic encoder that causes the joint network overly biased towards the prediction network. To alleviate this issue, a novel regularisation method is proposed to penalise the gradients back-propagated into the prediction network in early training stages, which can improve RNN-T performance without bringing any cost to both training and test. Experiments were conducted on a large-scale multilingual ASR setup for voice search, similar to the one used in [34]. As a result, by jointly using these proposed methods, more than 4% relative word error rate (WER) reductions were achieved by only increasing a few million extra parameters.

The rest of the paper is organised as follows: Sec. 2 reviews RNN-T and the work related to joint network. Sec. 3 gives details of the proposed joint network structures and the regularisation method. Sec. 4 describes our experimental setup, followed by the discussions on the results in Sec. 5. We conclude in Sec. 6.

2. BACKGROUND

2.1. RNN Transducer

In the traditional statistical ASR framework, [35], speech is produced and encoded via a noisy channel and the ASR system is to find the most probable source text sequence $\mathbf{y}^*$ given the acoustic feature sequence $\mathbf{x}_{1:T}$ of length T observed as the output of the channel. Based on Bayes' rule, decoding follows the *maximum a posteriori* rule to search over each possible hypothesized text sequence $\mathbf{y}$ by

$$P(\mathbf{y}|\mathbf{x}_{1:T}) \propto p(\mathbf{x}_{1:T}|\mathbf{y})P(\mathbf{y}), \tag{1}$$

where $p(\mathbf{x}_{1:T}|\mathbf{y})$ is estimated by the acoustic model and is the likelihood of generating $\mathbf{x}_{1:T}$ through the channel; $P(\mathbf{y})$ is estimated by an LM, describing the underlying probabilistic distribution of the source text.

Thanks Dr. Zhehuai Chen and Dr. Matt Shannon for useful suggestions.

Instead of modelling $p(\mathbf{x}_{1:T}|\mathbf{y})$ and $P(\mathbf{y})$ by independent models in a modularised system, E2E methods, such as RNN-T, directly models $P(\mathbf{y}|\mathbf{x}_{1:T})$ by a single model. Let $\mathbf{y} = y_{1:U}$ where U is the number of subword units in $\mathbf{y}$. For a streaming setting without any look ahead frame and time reduction, $\mathbf{h}^{\text{enc}}_t$, the D^{enc}-dimensional (-dim) acoustic representation extracted by the acoustic encoder at time t, $\mathbf{h}^{\text{pred}}_u$, the D^{pred}-dim text representation of the u-th subword unit by the prediction network, and $\mathbf{h}^{\text{joint}}_{t,u}$, the D^{joint}-dim fused representation generated by the joint network, are calculated as follows:

$$
\begin{aligned}
\mathbf{h}^{\text{enc}}_t &= \text{AcousticEncoder}(\mathbf{x}_{1:t}), && (2)\\
\mathbf{h}^{\text{pred}}_u &= \text{PredictionNetwork}(y_{1:u-1}), && (3)\\
\mathbf{h}^{\text{joint}}_{t,u} &= \text{JointNetwork}(\mathbf{h}^{\text{enc}}_t, \mathbf{h}^{\text{pred}}_u), && (4)\\
P(\hat{y}_i = k|y_{0:u-1}, \mathbf{x}_{1:t}) &= \text{Softmax}(\text{W}^{\text{out}}\mathbf{h}^{\text{joint}}_{t,u})|_k, && (5)
\end{aligned}
$$

where y_0 refers to the special start of sentence symbol; k and $\mathbf{W}^{\text{out}}$ are the k-th node and weights of the output layer. Regarding a set of subword units $\mathcal{V}$, the symbol that k represents belongs to $\mathcal{V} \cup \{\varnothing\}$, where $\varnothing$ is the blank symbol indicating no subword is emitted. During training, let $\hat{\mathbf{y}} = \{\hat{y}_1, \hat{y}_2, \ldots, \hat{y}_{T+U}\}$ be an alignment sequence of $\mathbf{y}$ that can be converted into $\mathbf{y}$ by removing all occurrences of $\varnothing$, $\mathcal{A}(\mathbf{x}_{1:T}, \mathbf{y})$ be the reference lattice including all possible alignment sequences between $\mathbf{y}$ and $\mathbf{x}_{1:T}$, $P(\mathbf{y}|\mathbf{x}_{1:T})$ can be computed efficiently with the *forward-backward procedure*. There is

$$
P(\mathbf{y}|\mathbf{x}_{1:T}) = \sum\nolimits_{\hat{\mathbf{y}} \in \mathcal{A}(\mathbf{x}_{1:T}, \mathbf{y})} \prod\nolimits_{i=1}^{T+U} P(\hat{y}_i|\mathbf{x}_{1:t_i}, y_{1:u_i}), \qquad (6)
$$

where t_i and u_i are the values of t and u corresponding to $\hat{y}_i$ in $\hat{\mathbf{y}}$.

In practice, AcousticEncoder$(\cdot)$ can be Conformer with a fixed number of look ahead frames and a fixed time reduction rate. PredictionNetwork$(\cdot)$ is often a multi-layer LSTM. The joint network is often defined as an FC layer [6] that

$$
\mathbf{h}^{\text{joint}}_{t,u} = \tanh(\mathbf{W}^{\text{joint}}_1 \mathbf{h}^{\text{enc}}_t + \mathbf{W}^{\text{joint}}_2 \mathbf{h}^{\text{pred}}_u), \qquad (7)
$$

where $\mathbf{W}^{\text{joint}}_1$ and $\mathbf{W}^{\text{joint}}_2$ are weight matrices. For simplicity, bias is ignored in Eqn. (7) and the rest of the paper.

If $\mathbf{W}^{\text{joint}}_2$ is $\mathbf{0}_{D^{\text{joint}} \times D^{\text{pred}}}$ and the joint network transforms only the acoustic representation, apart from their difference in $\mathcal{A}(\mathbf{x}_{1:T}, \mathbf{y})$ [11, 28], RNN-T becomes CTC that calculates $P(\hat{y}_i|\mathbf{x}_{1:t_i})$ by making an independence assumption between any subword units in $\mathbf{y}$. This reveals the importance of the joint network.

2.2. Related work

In Eqn. (7), by enforcing $\mathbf{h}^{\text{enc}}_t = \mathbf{0}$, the prediction network, joint network, and output layer jointly form an LSTM LM that is often referred to as the *internal LM* [28]. Studies showed that more WER reductions can be found from shallow fusion with external LMs by discounting the internal LM scores at test-time [27–29]. More recently, stateless RNN-T has been proposed to truncate the LM history embedded in the prediction network to n-gram [25].

The fusion of representations associated with different modalities plays a key role in multimodal intelligence [36]: attention, gating, and bilinear pooling are the most commonly used structures for the purpose. In RNN-T, the standard joint network implemented based on Eqn. (7) can be viewed as to fuse the acoustic representation $\mathbf{h}^{\text{enc}}_t$ and text representation $\mathbf{h}^{\text{pred}}_u$ using an FC layer. Recently, an alternative joint network structure is proposed to model the multiplicative interactions between the two representations [33]:

$$
\mathbf{h}^{\text{joint}}_{t,u} = \tanh(\mathbf{W}^{\text{joint}}_1 \mathbf{h}^{\text{enc}}_t \odot \mathbf{W}^{\text{joint}}_2 \mathbf{h}^{\text{pred}}_u), \qquad (8)
$$

where $\odot$ is the Hadamard product.

3. FUSING ACOUSTIC AND TEXT REPRESENTATIONS

Fusing acoustic and text representations is arguably a difficult task, and the standard joint network simply uses one FC layer. Sec. 3.1 and 3.2 propose more complex joint network structures, and Sec. 3.3 proposes to improve the balance between the two modalities.

3.1. Gating mechanism

Gating has been widely used in recurrent and shortcut structures [19, 37], whose most famous application is LSTM. It allows each element in each representation vector to be scaled with a different dynamic weight, before being integrated via vector addition. Specifically,

$$
\begin{aligned}
\mathbf{g}_{t,u} &= \sigma(\mathbf{W}^{\text{gate}}_1 \mathbf{h}^{\text{enc}}_t + \mathbf{W}^{\text{gate}}_2 \mathbf{h}^{\text{pred}}_u), \qquad (9)\\
\mathbf{h}^{\text{joint}}_{t,u} &= \mathbf{g}_{t,u} \odot \tanh(\mathbf{W}^{\text{joint}}_1 \mathbf{h}^{\text{enc}}_t) + (1 - \mathbf{g}_{t,u}) \odot \tanh(\mathbf{W}^{\text{joint}}_2 \mathbf{h}^{\text{text}}_u),
\end{aligned}
$$

where $\mathbf{g}_{t,u}$ is the gating vector, $\sigma(\cdot)$ is sigmoid function, and $\mathbf{W}^{\text{gate}}_1$ and $\mathbf{W}^{\text{gate}}_2$ are weight matrices of the gating layer. Notably a different gating vector can be computed to replace $1 - \mathbf{g}_{t,u}$. However, we observed worse WERs when using two separate gating vectors.

3.2. Bilinear pooling

Compared to gating, bilinear pooling is a more powerful and expensive method for fusing multimodal representations [38], which combines two vectors using the bilinear form (with bias ignored)

$$
h^{\text{joint}}_{t,u,d} = (\mathbf{h}^{\text{enc}}_t)^{\text{T}} \mathbf{W}^{\text{bi}}_d \, \mathbf{h}^{\text{pred}}_u, \qquad (10)
$$

where $\mathbf{W}^{\text{bi}}_d$ is a $D^{\text{enc}} \times D^{\text{pred}}$-dim matrix, and $h^{\text{joint}}_{t,u,d}$ is the d-th element of $\mathbf{h}^{\text{joint}}_{t,u}$. Considering all elements in $\mathbf{h}^{\text{joint}}_{t,u}$, $[\mathbf{W}^{\text{bi}}_1, \ldots, \mathbf{W}^{\text{bi}}_{D^{\text{joint}}}]$ is a $D^{\text{enc}} \times D^{\text{pred}} \times D^{\text{joint}}$-dim weight tensor. Fusing $\mathbf{h}^{\text{enc}}_t$ and $\mathbf{h}^{\text{pred}}_u$ into $\mathbf{h}^{\text{joint}}_{t,u}$ is therefore equivalent to perform

$$
\mathbf{h}^{\text{joint}}_{t,u} = [\text{Vector}(\mathbf{W}^{\text{bi}}_1), \ldots, \text{Vector}(\mathbf{W}^{\text{bi}}_{D^{\text{joint}}})]^{\text{T}} \text{Vector}(\mathbf{h}^{\text{enc}}_t \otimes \mathbf{h}^{\text{pred}}_u),
$$

where Vector$(\cdot)$ and $\otimes$ are the vectorisation and outer product operations. Compared to gating, bilinear pooling first computes the outer product of the two vectors to capture the multiplicative interactions between all possible element pairs in a more expressive $D^{\text{enc}} \times D^{\text{pred}}$-dim space, and then projects it into a D^{joint}-dim vector space.

To avoid the issues when estimating a high dimensional weight tensor, a low-rank approximation $\mathbf{W}^{\text{bi}}_d \approx \mathbf{W}^{\text{low}}_{1,d}(\mathbf{W}^{\text{low}}_{2,d})^{\text{T}}$ is suggested [39], where $\mathbf{W}^{\text{low}}_{1,d}$ and $\mathbf{W}^{\text{low}}_{2,d}$ are $D^{\text{enc}} \times D^{\text{rank}}$-dim and $D^{\text{pred}} \times D^{\text{rank}}$-dim matrices, and D^{rank} is the rank of $\mathbf{W}^{\text{bi}}_d$. Therefore Eqn. (10) can be rewritten as $h^{\text{join}}_{t,u,d} \approx (\mathbf{h}^{\text{enc}}_t)^{\text{T}} \mathbf{W}^{\text{low}}_{1,d} (\mathbf{W}^{\text{low}}_{2,d})^{\text{T}} \mathbf{h}^{\text{pred}}_u = \mathbf{1}^{\text{T}}((\mathbf{W}^{\text{low}}_{1,d})^{\text{T}} \mathbf{h}^{\text{enc}}_t \odot (\mathbf{W}^{\text{low}}_{2,d})^{\text{T}} \mathbf{h}^{\text{pred}}_u)$. It was proposed to tie all $\mathbf{W}^{\text{low}}_{1,d}$ as $\mathbf{W}^{\text{low}}_1$ and all $\mathbf{W}^{\text{low}}_{2,d}$ as $\mathbf{W}^{\text{low}}_2$, and to use a projection matrix $\mathbf{W}^{\text{proj}}$ to distinguish the elements in $\mathbf{h}^{\text{joint}}_{t,u}$ [40]. When a tanh function is used to transform the vectors before the Hadamard product, there is

$$
\hat{\mathbf{h}}^{\text{joint}}_{t,u} = \mathbf{W}^{\text{proj}}(\tanh((\mathbf{W}^{\text{low}}_1)^{\text{T}} \mathbf{h}^{\text{enc}}_t) \odot \tanh((\mathbf{W}^{\text{low}}_2)^{\text{T}} \mathbf{h}^{\text{pred}}_u)). \qquad (11)
$$

We found using shortcut connections [41] and a final tanh transform are important for bilinear pooling for RNN-T, and thus propose

$$
\mathbf{h}^{\text{joint}}_{t,u} = \tanh(\hat{\mathbf{h}}^{\text{joint}}_{t,u} + \mathbf{W}^{\text{joint}}_1 \mathbf{h}^{\text{enc}}_t + \mathbf{W}^{\text{joint}}_2 \mathbf{h}^{\text{pred}}_u), \qquad (12)
$$

where $\mathbf{W}^{\text{joint}}_1 \mathbf{h}^{\text{enc}}_t$ and $\mathbf{W}^{\text{joint}}_2 \mathbf{h}^{\text{pred}}_u$ are the shortcut connections here.

At last, we propose a stack structure to combine gating and bilinear pooling to leverage their complementarity. It is implemented by replacing Eqn. (11) with

$$\hat{\mathbf{h}}^{\text{joint}}_{t,u} = \mathbf{W}^{\text{proj}}(\tanh((\mathbf{W}_1^{\text{low}})^{\text{T}}\mathbf{h}_t^{\text{enc}}) \odot \tanh((\mathbf{W}_2^{\text{low}})^{\text{T}}\mathbf{h}^{\text{gate}}_{t,u})), \quad (13)$$

where $\mathbf{h}^{\text{gate}}_{t,u}$ refers to the joint representation computed by Eqn. (9).

3.3. Prediction network regularisation

It has been observed that strong and prevalent text priors (*e.g.* "bananas are yellow") often caused image-text multimodal systems to overfit to the statistical biases and tendencies, and largely circumvents the need to understand visual scenes [42]. Similary in RNN-T, since text priors are often easier to learn than the acoustic patterns, it is possible that the faster converging speed of the prediction network (than the acoustic encoder) makes $\mathbf{h}^{\text{pred}}_u$ overly weighted in $\mathbf{h}^{\text{joint}}_{u,t}$. In that situation, the acoustic encoder is less well trained to handle the audio samples with high internal LM scores.

In order to resolve this issue, we propose a prediction network regularisation method applied to the beginning of RNN-T training. It is implemented by recomputing the text representation as

$$\mathbf{h}^{\text{pred}}_u = \alpha_m\,\mathbf{h}^{\text{pred}}_u - \text{sg}((\alpha_m - 1)\,\mathbf{h}^{\text{pred}}_u), \quad (14)$$

where m is the index of current training step, α_m is a scaling factor, sg$(\cdot)$ is the stop gradient function whose input tensor will have zero gradients. When $0 \leqslant \alpha_m \leqslant 1$, the value of $\mathbf{h}^{\text{pred}}_u$ will not be changed but its corresponding gradients that back-propagate into the prediction network will be reduced by a factor of α_m. This slows down the convergence of the prediction network and makes the joint network fuse $\mathbf{h}^{\text{enc}}_t$ and $\mathbf{h}^{\text{pred}}_u$ in a more balanced way. In this paper, a piece-wise linear scheduler is used to control the value of α_m:

$$\alpha_m = \begin{cases} 0 & \text{if } m < m_1 \\ 1 & \text{else if } m \geqslant m_2 \\ (m - m_1)/(m_2 - m_1) & \text{otherwise} \end{cases}, \quad (15)$$

where m_1 and m_2 are two pre-defined hyper-parameters. Notably, this method is different from initialising RNN-T with a pre-trained CTC model even when $\alpha_m = 0$, since the prediction network serves as a random but fix-valued projection, through which RNN-T is still able to obtain y_{u-1}. This links to the stateless RNN-T [25].

4. EXPERIMENTAL SETUP

4.1. Datasets

Experiments were conducted on a dataset with 9 language locales: US English (en-US), UK English (en-GB), French (fr-FR), Italian (it-IT), Germany (de-DE), US Spanish (es-US), ES Spanish (es-ES), Taiwan Chinese (zh-TW) and Japanese (ja-JP). All data are anonymised and hand-transcribed. There are totally 214.2M utterances which correspond to 142.3K hours of speech data collected from Google's Voice Search traffic. en-US and en-GB take about 25% and 5% of the training data, while each of the rest 7 languages takes about 10% of the data. The SpecAugment method is used to improve ASR robustness against noisy conditions [43]. The training data is mixed with all languages without using any language id information. The test sets are kept distinct for each language with each of them containing 3.3K$\sim$15.4K utterances. The testing utterances are also sampled from Google's Voice Search traffic with a maximum duration constraint of 5.5 second long for each utterance. The test sets have no overlapping from the training set for evaluation purpose.

4.2. Model setup

The 80-dim log-mel filter bank features are used, which are computed using a 32ms frame length and a 10ms shift. Acoustic features from 3 contiguous frames are stacked and subsampled to form a 240-dim input representation with 30ms frame rate, which are then transformed using a linear projection to 512-dim and added with positional embeddings. Twelve Conformer [21] encoder blocks with 8-head self-attention and a convolution kernel size of 15 are used to further transform the stacked features. A concatenation operation is performed after the 3rd block to achieve a time reduction rate of 2, and the resulted 1024-dim vectors are transformed by the 4th Conformer block and then projected back to 512-dim using another linear transform. Afterwards comes with another 8 Conformer blocks followed by a final linear normalisation layer. These layers combined together form the RNN-T acoustic encoder. The prediction network consists of two layers of 2,048-dim LSTM with a 640-dim linear projection to make D^{pred} =640. The dimension of the fused representation D^{joint} is also set to 640. All models are trained to predict 16,384 word-piece units [44]. As a result, the final RNN-T baseline has 110M parameters in the acoustic encoder and 33M parameters in the rest of the model. All models are trained in Tensorflow using the Lingvo toolkit [45] on Google's Tensor Processing Units V3 with a global batch size of 4,096 utterances. Models are optimized using synchronized stochastic gradient descent based on the Adam optimiser with $\beta_1 = 0.9$ and $\beta_2 = 0.999$. During test, the models are tested in a fully E2E fashion without any external LMs.

5. EXPERIMENTAL RESULTS

In this section, RNN-T systems with different joint network structures are first compared at the 200K-th training step, whose details are listed in Table 1. Next, prediction network regularisation with different hyper-parameter values are compared using S4 at the 500K-th step. Final results are presented with 800K training steps.

Table 1. Details of systems with different joint network structures. "#Params" refers to the number of joint network parameters.

ID	Structure	Equations	D^{joint}	D^{rank}	#Params
S0	FC with add.	(7)	640	–	0.73M
S1	FC with add.	(7)	790	–	3.36M
S2	FC with mul.	(8)	640	–	0.73M
S3	Gating	(9)	640	–	1.47M
S4	Bilinear	(11) & (12)	640	640	1.88M
S5	Bilinear	(11) & (12)	640	1280	3.03M
S6	Combination	(13) & (12)	640	640	3.36M

5.1. On joint network structures

The results of RNN-T models with different joint networks are presented in Table 2. First, S0 and S2 result in similar WERs, which validates the finding in [33]. Next, by increasing D^{joint} from 640 (S0) to 790 (S1), the averaged WER was only slightly reduced by 0.03%. We also tried to add more FC layers to the joint network that resulted in worse training loss values and higher WERs.

Both gating (S3) and bilinear pooling (S4) outperformed FC fusion systems S0 and S1, where S1 has the same amount of parameters as S3 and S4 indicating that the lower WERs of S3 and S4 do not come from the extra model parameters. Furthermore, S4 outperformed S3 meaning that bilinear pooling can produce more expressive joint representations. Another advantage of bilinear pooling is

Table 2. The 200K training step WERs with different joint network structures.

ID	de-DE	en-GB	en-US	es-ES	es-US	fr-FR	it-IT	ja-JP	zh-TW	Avg.
S0 [6]	15.4	7.1	7.9	8.1	8.1	13.4	9.7	13.9	6.1	9.97
S1 [6]	15.2	7.4	7.8	8.1	7.8	13.3	9.7	14.1	6.1	9.94
S2 [33]	15.3	7.4	7.9	8.1	7.9	13.1	9.7	14.0	6.2	9.96
S3	15.1	7.4	7.6	7.9	8.0	12.9	9.4	13.7	6.1	9.78
S4	15.1	7.2	7.5	7.8	7.8	13.1	9.2	13.7	5.9	**9.71**
S5	15.0	7.2	7.5	7.7	7.6	13.3	9.1	13.6	6.0	9.67
S6	14.6	7.1	7.4	7.8	7.7	13.1	9.4	13.6	6.0	**9.63**

Table 3. Final 800K training step WERs. Superscript "reg" means to use regularisation with m_1 =25K and m_2 =200K.

ID	de-DE	en-GB	en-US	es-ES	es-US	fr-FR	it-IT	ja-JP	zh-TW	Avg.
S0	14.2	6.6	6.8	7.5	7.5	13.0	8.6	12.9	5.6	9.19
S0$^{\text{reg}}$	13.9	6.5	6.4	7.3	7.4	12.7	8.1	12.5	5.6	**8.93**
S1	14.3	6.6	7.0	7.5	7.6	13.0	8.6	12.6	5.6	9.20
S1$^{\text{reg}}$	14.0	6.5	6.8	7.2	7.5	12.7	8.3	12.4	5.4	**8.98**
S4	13.8	6.4	6.7	7.2	7.7	12.8	8.1	12.3	5.4	8.93
S4$^{\text{reg}}$	13.7	6.4	6.6	7.0	7.3	12.7	7.9	12.2	5.4	**8.80**
S5	13.8	6.4	6.6	7.1	7.4	12.8	8.2	12.3	5.5	8.90
S6	13.8	6.2	6.6	7.1	7.4	12.7	8.2	12.3	5.4	8.86
S6$^{\text{reg}}$	13.5	6.3	6.5	7.0	7.2	12.6	8.0	12.1	5.3	**8.72**

having the flexibility to control D^{joint} and D^{rank} separately. Increasing D^{rank} from 640 to 1280 allows S5 to have more parameters and lower WERs without increasing the input size of the big output layer with 16,384 nodes. At last, by stacking gating and bilinear pooling in the joint network, S6 leverages the complementarity of both structures and outperforms any other systems. In particular, with the same amount of parameters, S6 outperformed S1 by a 0.31% WER reduction. In conclusion, all of our proposed systems (S3 – S6) improved WERs considerably by increasing only a small percent of the model parameters (S0 has 144M parameters as given in Sec. 4.2).

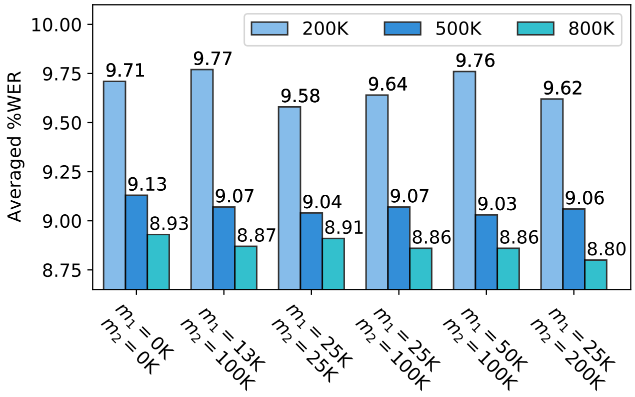

Fig. 1. S4 system WERs with different hyper-parameters for prediction network regularsation.

5.2. On prediction network regularisation

The results of using the prediction network regularsation defined in Eqns. (14) and (15) with different m_1 and m_2 values are shown in Fig 1. S4, the bilinear pooling joint network with $D^{\text{rank}} = 640$, is used in this section. The best results were found with m_1 =25K and m_2 =200K, which improved the averaged WER by 0.09%, 0.07%, and 0.13% absolute with 200K, 500K, and 800K steps separately.

5.3. Final results

The models are trained for 800K training steps and the final results are shown in Table 3. Comparing S4, S5, and S6 to S1, better joint network structures lead to lower WERs without requiring more parameters. The prediction network regularisation improved averaged WERs by 0.26%, 0.13% and 0.14% for S0, S4, and S6, although the improvements are not always consistent regarding each individual language. Compared to baseline system S0, our best-performing systems S4 and S6 with the regularisation method achieved 4.2% and 5.1% relative reductions in averaged WER by increasing only 1.15M and 2.63M parameters.

6. CONCLUSIONS

By viewing the function of the joint network of RNN-T as to fuse the acoustic and text representations derived from the acoustic encoder and prediction network, we propose in this paper to apply the structures widely used for multimodal representation fusion, including gating and bilinear pooling, to improve the joint network implementation. These structures are modified and combined to fit into RNN-T. A novel prediction network regularisation method is also proposed to make the fusion between acoustic and text representations more balanced. When evaluated using a large-scale multilingual voice search setup, 4.2% and 5.1% relative WER reductions were found by using the bilinear pooling and the combination structure separately along with the regularisation.

7. REFERENCES

[1] J. Chorowski, D. Bahdanau, D. Serdyuk, K. Cho, and Y. Bengio, "Attention-based models for speech recognition," in *Proc. NIPS*, 2015.

[2] L. Lu, X. Zhang, K. Cho, and S. Renals, "A study of the recurrent neural network encoder-decoder for large vocabulary speech recognition," in *Proc. Interspeech*, Dresden, 2015.

[3] W. Chan, N. Jaitly, Q.V. Le, and O. Vinyals, "Listen, attend and spell: A neural network for large vocabulary conversational speech recognition," in *Proc. ICASSP*, Shanghai, 2016.

[4] S. Kim, T. Hori, and S. Watanabe, "Joint CTC-attention based end-to-end speech recognition using multi-task learning," in *Proc. ICASSP*, New Orleans, 2017.

[5] A. Graves, "Sequence transduction with recurrent neural networks," in *Proc. ICML Representation Learning Workshop*, Edinburgh, 2012.

[6] A. Graves, A. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in *Proc. ICASSP*, Vancouver, 2013.

[7] R. Prabhavalkar, K. Rao, T. Sainath, B. Li, L. Johnson, and N. Jaitly, "A comparison of sequence-to-sequence models for speech recognition," in *Proc. Interspeech*, Stockholm, 2017.

[8] E. Battenberg, J. Chen, R. Child, A. Coates, Y. Gaur, Y. Li, H. Liu, S. Satheesh, D. Seetapun, A. Sriram, and Z. Zhu, "Exploring neural transducers for end-to-end speech recognition," in *Proc. ASRU*, 2017.

[9] J. Li, R. Zhao, H. Hu, and Y. Gong, "Improving RNN transducer modeling for end-to-end speech recognition," in *Proc. ASRU*, 2019.

[10] A. Zeyer, A. Merboldt, R. Schlüter, and H. Ney, "A new training pipeline for an improved neural transducer," in *Proc. Interspeech*, 2020.

[11] Y. He, T. Sainath, R. Prabhavalkar, I. McGraw, R. Alvarez, D. Zhao, D. Rybach, A. Kannan, Y. Wu, R. Pang, Q. Liang, D. Bhatia, Y. Shangguan, B. Li, G. Pundak, K.C. Sim, T. Bagby, S. Chang, R. Rao, and A. Gruenstein, "Streaming end-to-end speech recognition for mobile devices," in *Proc. ICASSP*, Brighton, 2019.

[12] S. Chang, R. Prabhavalkar, Y. He, T.N. Sainath, and G. Simko, "Joint endpointing and decoding with end-to-end models," in *Proc. ICASSP*, Brighton, 2019.

[13] B. Li, S. Chang, T.N. Sainath, R. Pang, Y. He, T. Strohman, and Y. Wu, "Towards fast and accurate streaming end-to-end ASR," in *Proc. ICASSP*, Barcelona, 2020.

[14] Q. Zhang, H. Lu, H. Sak, A. Tripathi, E. McDermott, S. Koo, and S. Kumar, "Transformer Transducer: A streamable speech recognition model with Transformer encoders and RNN-T loss," in *Proc. ICASSP*, Barcelona, 2020.

[15] D. Le, M. Jain, G. Keren, S. Kim, Y. Shi, J. Mahadeokar, J. Chan, Y. Shangguan, C. Fuegen, O. Kalinli, Y. Saraf, and M.L. Seltzer, "Contextualized streaming end-to-end speech recognition with Trie-based deep biasing and shallow fusion," in *Proc. Interspeech*, Brno, 2021.

[16] Y. Shi, Y. Wang, C. Wu, C.-F. Yeh, J. Chan, F. Zhang, D. Le, and M. Seltzer, "Emformer: Efficient memory transformer based acoustic model for low latency streaming speech recognition," in *Proc. ICASSP*, Toronto, 2021.

[17] X. Chen, Y. Wu, Z. Wang, S. Liu, and J. Li, "Developing real-time streaming Transformer transducer for speech recognition on large-scale dataset," in *Proc. ICASSP*, Toronto, 2012.

[18] A. Graves, S. Fernandez, F. Gomez, and J. Schmidhuber, "Connectionist temporal classification: Labeling unsegmented sequenece data with recurrent neural networks," in *Proc. ICML*, Pittsburgh, 2006.

[19] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, pp. 1735–1780, 1997.

[20] K. Rao, H. Sak, and R. Prabhavalkar, "Exploring architectures, data and units for streaming end-to-end speech recognition with RNN-transducer," in *Proc. ASRU*, Okinawa, 2017.

[21] A. Gulati, J. Qin, C.-C. Chiu, N. Parmar, Y. Zhang, J. Yu, W. Han, S. Wang, Z. Zhang, Y. Wu, and R. Pang, "Conformer: Convolution-augmented transformer for speech recognition," in *Proc. Interspeech*, Shanghai, 2020.

[22] A. Narayanan, T.N. Sainath, R. Pang, J. Yu, C.-C. Chiu, R. Prabhavalkar, E. Variani, and T. Strohman, "Cascaded encoders for unifying streaming and non-streaming ASR," in *Proc. ICASSP*, Toronto, 2021.

[23] Y. Zhang, S. Sun, and L. Ma, "Tiny transducer: A highly-efficient speech recognition model on edge devices," in *Proc. ICASSP*, 2021.

[24] G. Sun, C. Zhang, and P.C. Woodland, "Tree-constrained pointer generator for end-to-end contextual speech recognition," in *Proc. ASRU*, Cartagena, 2021.

[25] M. Ghodsi, X. Liu, J. Apfel, R. Cabrera, and E. Weinstein, "RNN transducer with stateless prediction network," in *Proc. ICASSP*, 2020.

[26] G. Saon, Z. Tüske, and K. Audhkhasi, "Alignment-length synchronous decoding for RNN transducer," in *Proc. ICASSP*, Barcelona, 2020.

[27] E. McDermott, H. Sak, and E. Variani, "A density ratio approach to language model fusion in end-to-end automatic speech recognition," in *Proc. ASRU*, Signapore, 2019.

[28] E. Variani, D. Rybach, C. Allauzen, and M. Riley, "Hybrid autoregressive transducer (HAT)," in *Proc. ICASSP*, Barcelona, 2020.

[29] Z. Meng, N. Kanda, Y. Gaur, S. Parthasarathy, E. Sun, L. Lu, X. Chen, J. Li, and Y. Gong, "Internal language model training for domain-adaptive end-to-end speech recognition," in *Proc. ICASSP*, 2021.

[30] J. Li, R. Zhao, Z. Meng, Y. Liu, W. Wei, S. Parthasarathy, V. Mazalov, Z. Wang, L. He, S. Zhao, and Y. Gong, "Developing RNN-T models surpassing high-performance hybrid models with customization capability," in *Proc. Interspeech*, Shanghai, 2020.

[31] X. Zheng, Y. Liu, D. Gunceler, and D. Willett, "Using synthetic audio to improve the recognition of out-of-vocabulary words in end-to-end ASR systems," in *Proc. ICASSP*, Toronto, 2021.

[32] T. Doutre, W. Han, M. Ma, Z. Lu, C.-C. Chiu, R. Pang, A. Narayanan, A. Misra, Y. Zhang, and L. Cao, "Improving streaming automatic speech recognition with non-streaming model distillation on unsupervised data," in *Proc. ICASSP*, Toronto, 2021.

[33] G. Saon, Z. Tüske, D. Bolanos, and B. Kingsbury, "Advancing RNN transducer technology for speech recognition," in *Proc. ICASSP*, 2021.

[34] B. Li, R. Pang, T.N. Sainath, A. Gulati, Y. Zhang, J. Qin, P. Haghani, W.R. Huang, and M. Ma, "Scaling end-to-end models for large-scale multilingual ASR," in *Proc. ASRU*, Cartagena, 2021.

[35] F. Jelinek, *Statistical Methods for Speech Recognition*, MIT Press, 1997.

[36] C. Zhang, Z. Yang, X. He, and L. Deng, "Multimodal intelligence: Representation learning, information fusion, and applications," *IEEE Journal of Selected Topics in Signal Proc.*, vol. 14, pp. 478–493, 2020.

[37] Y. Zhang, G. Chen, D. Yu, K. Yao, S. Khudanpur, and J. Glass, "Highway long short-term memory RNNs for distant speech recognition," in *Proc. ICASSP*, Shanghai, 2016.

[38] J. Tenenbaum and W. Freeman, "Separating style and content with bilinear models," *Neural computation*, vol. 12, pp. 1247–1283, 2000.

[39] H. Pirsiavash, D. Ramanan, and C. Fowlkes, "Bilinear classifiers for visual recognition," in *Proc. NIPS*, Vancouver, 2009.

[40] J.H. Kim, K.W. On, W. Lim, J. Kim, J.W. Ha, and B.T. Zhang, "Hadamard product for low-rank bilinear pooling," in *Proc. ICLR*, Toulon, 2017.

[41] G. Sun, C. Zhang, and P.C. Woodland, "Combination of deep speaker embeddings for diarisation," *Neural Networks*, vol. 141, pp. 372–384, 2021.

[42] S. Ramakrishnan, A. Agrawal, , and S. Lee, "Overcoming language priors in visual question answering with adversarial regularization," in *Proc. NeurIPS*, Montreal, 2018.

[43] D.S. Park, Y. Zhang, C.-C. Chiu, Y. Chen, B. Li, W. Chan, Q.V. Le, and Y. Wu, "SpecAugment on large scale datasets," in *Proc. ICASSP*, Toronto, 2020.

[44] M. Schuster and K. Nakajima, "Japanese and Korean voice search," in *Proc. ICASSP*, Kyoto, 2012.

[45] J. Shen, P. Nguyen, et al., "Lingvo: A modular and scalable framework for sequence-to-sequence modeling," *arXiv:2005.08100*, 2019.