

BACK-TRANSLATION-STYLE DATA AUGMENTATION FOR END-TO-END ASR

¹Tomoki Hayashi, ²Shinji Watanabe, ³Yu Zhang, ¹Tomoki Toda, ⁴Takaaki Hori, ⁵Ramon Astudillo, ¹Kazuya Takeda

¹Nagoya University, Nagoya, JAPAN

²Center for Language and Speech Processing, Johns Hopkins University, Baltimore, MD, USA

³Google, Inc., USA

⁴Mitsubishi Electric Research Laboratories (MERL), Cambridge MA, USA

⁵Spoken Language Systems Lab, INESC-ID Lisboa, Portugal

ABSTRACT

In this paper we propose a novel data augmentation method for attention-based end-to-end automatic speech recognition (E2E-ASR), utilizing a large amount of text which is not paired with speech signals. Inspired by the back-translation technique proposed in the field of machine translation, we build a neural text-to-encoder model which predicts a sequence of hidden states extracted by a pre-trained E2E-ASR encoder from a sequence of characters. By using hidden states as a target instead of acoustic features, it is possible to achieve faster attention learning and reduce computational cost, thanks to sub-sampling in E2E-ASR encoder, also the use of the hidden states can avoid to model speaker dependencies unlike acoustic features. After training, the text-to-encoder model generates the hidden states from a large amount of unpaired text, then E2E-ASR decoder is retrained using the generated hidden states as additional training data. Experimental evaluation using LibriSpeech dataset demonstrates that our proposed method achieves improvement of ASR performance and reduces the number of unknown words without the need for paired data.

Index Terms— automatic speech recognition, end-to-end, data augmentation, back-translation

1. INTRODUCTION

Automatic speech recognition (ASR) is the task of converting a continuous speech signal into a sequence of discrete characters, and is a key technology for the realization of natural interaction between humans and machines. ASR technology has great potential in various applications such as voice search and voice input, making our lives more convenient. Typical ASR systems [1] consist of multiple modules such as an acoustic model, a lexicon model, and a language model. Dividing ASR systems into modules makes it possible to optimize each of them separately, but this also results in more complex systems and imposes performance limitations. Over the past few decades, this approach has been the basis of ASR systems.

With the improvement of deep learning techniques, end-to-end (E2E) approaches have begun to attract attention [2]. While typical ASR systems convert a sequence of acoustic features into text step-by-step using several modules trained separately, E2E-ASR systems directly convert speech using a single neural network. Therefore, the whole E2E-ASR system can be optimized jointly, making system construction much easier than with typical ASR systems. Furthermore, it does not require costly lexical information or morphological analysis.

The present E2E-ASR approaches can be divided into two types. First type is based on connectionist temporal classification (CTC) [2–6]. The CTC approach makes it possible to map the input sequences of acoustic features to output sequences of symbols of shorter length without using a hidden Markov model (HMM). However, it requires assumptions of conditional independence in the output sequence, i.e., each output symbol such as a character or phoneme is independently predicted in each frame. The second E2E-ASR approach utilizes an attention-based sequence-to-sequence (Seq2Seq) model [7]. In this approach, a sequence of acoustic features is directly mapped into text using an encoder-decoder architecture [8, 9]. In contrast to the CTC-based approach, the attention-based Seq2Seq approach is not bound by any assumptions, therefore it can be trained to directly maximize the probability of a word sequence given a sequence of acoustic features. However, in exchange for its generality, the Seq2Seq approach requires large amounts of data for training. Furthermore, since the language model is not a separate module, the large amounts of text typically available cannot be used to improve its performance. This actually yields significant degradation of proper noun recognition, which are not appeared in the paired speech and text data, and affects negatively to production when evaluated on live production data according to [10].

One straightforward approach to address these issues is to integrate a language model with the Seq2Seq model, including shallow fusion, deep fusion, and their variants [11–13]. Shallow fusion [11, 14] is the most simple approach in that we

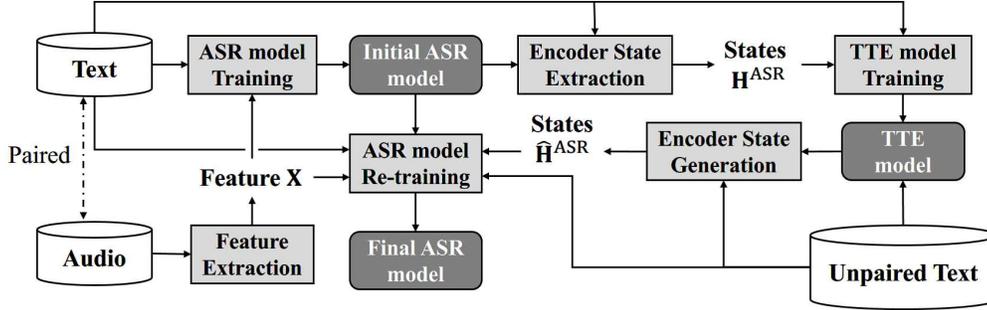


Fig. 1: Overview of proposed back-translation-style data augmentation method.

separately train a Seq2Seq model and a language model and then combine the score of two models in the decoding phase. Deep fusion [12] is an approach which has been proposed in the field of neural machine translation. A seq2seq model and a language model are trained separately, and then the hidden states of the decoder of the Seq2Seq model and those of the language model are concatenated using a gating matrix which controls the importance of each model. The parameters used to calculate the gating matrix are then trained using a small amount of training data while fixing all of the other parameters. These fusion approaches enable us to utilize a large amount of unpaired text to improve ASR performance. The resulting model is not actually end-to-end, however, since it requires additional steps and fine-tuning to integrate the separate modules.

A simpler approach is back-translation [15, 16], a method which has been proposed in the field of machine translation. In this approach, a pre-trained target-to-source translation model is used to generate source text from unpaired target text. Augmenting training data with back-translated data led to notable improvements in performance of neural machine translation models [15]. Similar techniques have also led to performance improvements in related tasks such as automatic post edition [17].

Inspired by the back-translation approach, in this paper we propose a novel data augmentation method for attention-based E2E-ASR models allows them to utilize large amounts of text not paired with speech signals. Instead of using a text-to-speech system on unpaired text to produce synthetic speech [18] or using grapheme to phoneme conversion to generate paired text and pseudo speech sequences based on phonemes [19], we build a text-to-encoder model which learns to predict the hidden states of the E2E-ASR encoder. Targeting the states of the speech encoder, rather than speech itself makes it possible to achieve faster attention learning and reduce computational cost, thanks to sub-sampling present in E2E-ASR encoder. Furthermore, the use of the hidden states can avoid to model speaker dependencies unlike acoustic features. After training, the text-to-encoder model generates the hidden states from a large amount of unpaired text, and then the decoder of the E2E-ASR model is retrained using the generated hidden states as additional training data.

To evaluate our proposed method, we conduct experimental evaluation using LibriSpeech dataset [20]. The experimental results demonstrate that our proposed method achieves the improvement of ASR performance and makes it possible to improve the recognition results for unknown words.

2. BACK-TRANSLATION-STYLE DATA AUGMENTATION

2.1. Overview

An overview of our proposed back-translation-style data augmentation method is shown in Fig. 1. First, the attention-based E2E-ASR model is trained using paired training data which consists of text and speech. Next, the final layer hidden states of the ASR encoder are extracted, providing paired training data which consists of text and the corresponding hidden states. Using this paired training data, a neural text-to-encoder (TTE) model is trained to predict the hidden states of the ASR encoder from a sequence of characters. Finally, the text-to-encoder model generates hidden states from a large amount of unpaired text and the ASR decoder is retrained using the generated states as additional training data.

2.2. ASR model training

An overview of an attention-based ASR model is shown in Fig. 2(a). This model directly estimates posterior $p(\mathbf{C}|\mathbf{X})$, where $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_T\}$ represents a sequence of input features, and $\mathbf{C} = \{c_1, c_2, \dots, c_L\}$ represents a sequence of output characters. Posterior $p(\mathbf{C}|\mathbf{X})$ is factorized with a probabilistic chain rule as follows:

$$p(\mathbf{C}|\mathbf{X}) = \prod_{l=1}^L p(c_l|c_{1:l-1}, \mathbf{X}), \quad (1)$$

where $c_{1:l-1}$ represents subsequence $\{c_1, c_2, \dots, c_{l-1}\}$, and $p(c_l|c_{1:l-1}, \mathbf{X})$ is calculated as follows:

$$\mathbf{h}_t^{\text{asr}} = \text{Encoder}^{\text{asr}}(\mathbf{X}), \quad (2)$$

$$a_{lt}^{\text{asr}} = \text{Attention}^{\text{asr}}(\mathbf{q}_{l-1}^{\text{asr}}, \mathbf{h}_t^{\text{asr}}, \mathbf{a}_{l-1}^{\text{asr}}), \quad (3)$$

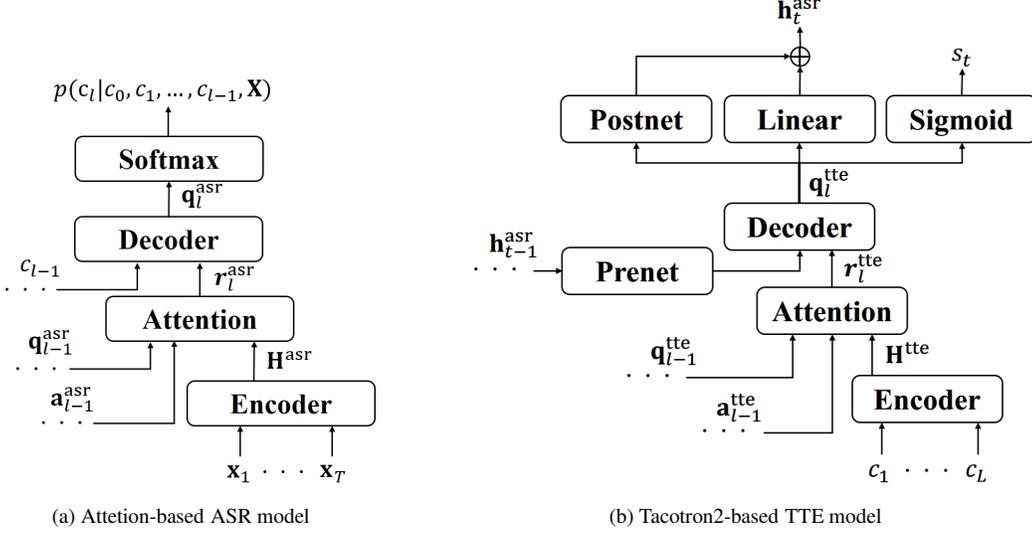


Fig. 2: Overview of attention-based ASR and Tacotron2-based TTE network architectures.

$$\mathbf{r}_l^{\text{asr}} = \sum_{t=1}^T a_{lt}^{\text{asr}} \mathbf{h}_t^{\text{asr}}, \quad (4)$$

$$\mathbf{q}_l^{\text{asr}} = \text{Decoder}^{\text{asr}}(\mathbf{r}_l^{\text{asr}}, \mathbf{q}_{l-1}^{\text{asr}}, c_{l-1}), \quad (5)$$

$$p(c_l | c_{1:l-1}, \mathbf{X}) = \text{Softmax}(\text{LinB}(\mathbf{q}_l^{\text{asr}})), \quad (6)$$

where a_{lt}^* represents an attention weight, \mathbf{a}_l^* represents an attention weight vector (sequence of attention weights $\{a_{l0}^*, a_{l1}^*, \dots, a_{lt}^*\}$), \mathbf{h}_t^* and \mathbf{q}_l^* represent the hidden states of encoder and decoder networks, respectively, \mathbf{r}_l^* represents a letter-wise hidden vector, which is a weighted summarization of the hidden vectors using attention weight vector \mathbf{a}_l^* , and $\text{LinB}(\cdot)$ represents a linear layer with a trainable matrix and bias parameters.

All of the above networks are optimized using back-propagation through time (BPTT) [21] to minimize the following objective function:

$$\begin{aligned} \mathcal{L}_{\text{asr}} &= -\log p(\mathbf{C} | \mathbf{X}) \\ &= -\log \left(\sum_{l=1}^L p(c_l | c_{1:l-1}^*, \mathbf{X}) \right), \end{aligned} \quad (7)$$

where $c_{1:l-1}^* = \{c_1^*, c_2^*, \dots, c_{l-1}^*\}$ represents the ground truth of the previous characters.

2.3. TTE model training

As our neural text-to-encoder (TTE) model we use Tacotron2, which has demonstrated superior performance in the field of text-to-speech synthesis [22]. An overview of its network architecture is shown in Fig. 2(b). In our framework, the network predicts ASR encoder state $\mathbf{h}_t^{\text{asr}}$ and the probability of the end of sequence s_t at each frame t from a sequence of input characters $\mathbf{C} = \{c_1, c_2, \dots, c_L\}$ as follows:

$$\mathbf{h}_l^{\text{tte}} = \text{Encoder}^{\text{tte}}(\mathbf{C}), \quad (8)$$

$$a_{lt}^{\text{tte}} = \text{Attention}^{\text{tte}}(\mathbf{q}_{l-1}^{\text{tte}}, \mathbf{h}_l^{\text{tte}}, \mathbf{a}_{t-1}^{\text{tte}}), \quad (9)$$

$$\mathbf{r}_l^{\text{tte}} = \sum_{l=1}^L a_{tl}^{\text{tte}} \mathbf{h}_l^{\text{tte}}, \quad (10)$$

$$\mathbf{v}_{t-1} = \text{Prenet}(\mathbf{h}_{t-1}^{\text{asr}}), \quad (11)$$

$$\mathbf{q}_l^{\text{tte}} = \text{Decoder}^{\text{tte}}(\mathbf{r}_l^{\text{tte}}, \mathbf{q}_{l-1}^{\text{tte}}, \mathbf{v}_{t-1}), \quad (12)$$

$$\hat{\mathbf{h}}_t^{b,\text{asr}} = \tanh(\text{LinB}(\mathbf{q}_l^{\text{tte}})), \quad (13)$$

$$\mathbf{d}_t = \text{Postnet}(\mathbf{q}_l^{\text{tte}}), \quad (14)$$

$$\hat{\mathbf{h}}_t^{a,\text{asr}} = \tanh(\text{LinB}(\mathbf{q}_l^{\text{tte}}) + \mathbf{d}_t), \quad (15)$$

$$\hat{s}_t = \text{Sigmoid}(\text{LinB}(\mathbf{q}_l^{\text{tte}})), \quad (16)$$

where $\text{Prenet}(\cdot)$ is a shallow feed-forward network to convert the network outputs before feedback to the decoder, $\text{Postnet}(\cdot)$ is a convolutional neural network to refine the network outputs, and $\hat{\mathbf{h}}_t^{b,\text{asr}}$ and $\hat{\mathbf{h}}_t^{a,\text{asr}}$ represent predicted hidden states of the ASR encoder before and after refinement by Postnet. Note that the indices of the encoder and decoder states are reversed in comparison to the ASR formulation in Eqs. (2)-(6), and that we use an additional activation function $\tanh(\cdot)$ in Eqs. (13) and (15) to avoid mismatching of the ranges of the outputs, in contrast to the original Tacotron2 architecture [22].

All of the networks are jointly optimized to minimize the following objective functions:

$$\begin{aligned} \mathcal{L}_{\text{tte}} &= \text{MSE}(\hat{\mathbf{h}}_t^{a,\text{asr}}, \mathbf{h}_t^{\text{asr}}) + \text{MSE}(\hat{\mathbf{h}}_t^{b,\text{asr}}, \mathbf{h}_t^{\text{asr}}) \\ &+ \text{L1}(\hat{\mathbf{h}}_t^{a,\text{asr}}, \mathbf{h}_t^{\text{asr}}) + \text{L1}(\hat{\mathbf{h}}_t^{b,\text{asr}}, \mathbf{h}_t^{\text{asr}}) \\ &+ \frac{1}{T} \sum_{t=1}^T s_t \ln \hat{s}_t + (1 - s_t) \ln(1 - \hat{s}_t), \end{aligned} \quad (17)$$

where $\text{MSE}(\cdot)$ represents mean square error, $\text{L1}(\cdot)$ represent an L1 norm, and the last two terms represent the binary cross entropy for the probability of the end of sequence. The use of losses for both outputs, before and after Postnet aids fast convergence. In the original Tacotron2 paper [22], the L1 norm was not used as the objective function, however, in [23]

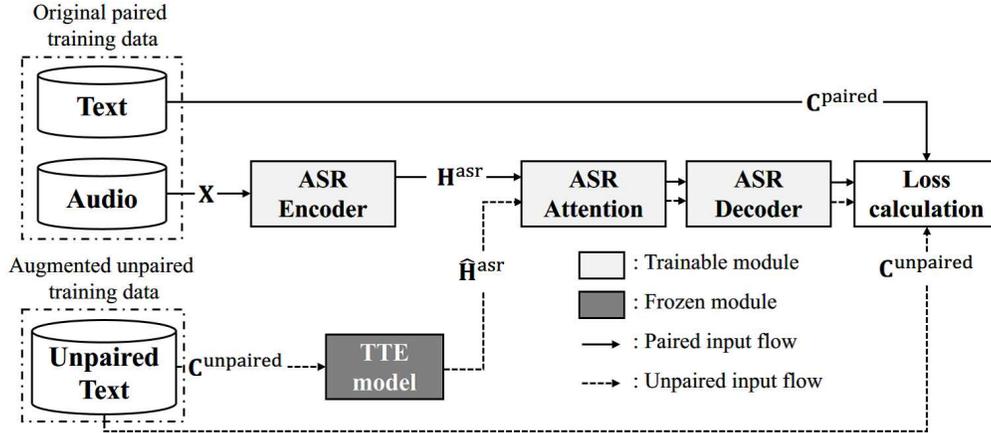


Fig. 3: Flowchart of proposed retraining.

it was reported that the use of L1 norm improves performance, especially when using noisy training data.

2.4. ASR decoder retraining

After training of the TTE model, we retrain the ASR decoder using both the paired and unpaired training data. A flowchart of this retraining is shown in Fig. 3. We concatenate the paired and unpaired text datasets, and then for each text, if there is paired speech data, the acoustic features of that speech are used as inputs. If not, the hidden states generated by the TTE model are used as inputs. Using both the generated hidden states and the original acoustic features produces a regularization effect which prevents overfitting to the generated states.

3. EXPERIMENTAL EVALUATION

3.1. Experimental conditions

We conducted an experimental evaluation using the LibriSpeech dataset [20], which consists of two sets of clean speech data (100 hours + 360 hours), and noisy speech data (500 hours) for training. We used 100 hours of clean speech data to train the initial ASR model and the text-to-encoder (TTE) model, and the text of 360 hours of clean speech data to retrain the ASR decoder. We used five hours of clean development data as a validation set, and five hours of clean test data as an evaluation set. To evaluate the effectiveness of our proposed method, we compared the recognition performance of the following seven methods:

Baseline

model trained with 100 hours of acoustic features;

Retrain-State

model retrained with 360 hours of generated hidden states and 100 hours of extracted hidden states;

Retrain-State-Frozen

model retrained with 360 hours of generated hidden states and 100 hours of extracted hidden states while the attention layers are frozen;

Retrain-Joint

model retrained with 360 hours of generated hidden states and 100 hours of acoustic features;

Oracle-State

model trained with 460 hours of extracted hidden states;

Oracle-State-Frozen

model trained with 460 hours of extracted hidden states while the attention layers are frozen;

Oracle-Feature

model trained with 460 hours of acoustic features;

where “generated hidden states” represent the hidden states generated by the TTE model, and “extracted hidden states” represent the hidden states extracted from the ASR encoder using raw acoustic features.

We used an acoustic feature vector consisting of an 80-dimensional log Mel-filter bank and three-dimensional pitch features, which were extracted using the open-source speech recognition toolkit Kaldi [24]. The ASR encoder consisted an eight-layered bidirectional long short-term memory with a projection layer [25] (BLSTMP), and the ASR decoder consisted a one-layered LSTM. In the second and third layers from the bottom of the ASR encoder, sub-sampling was performed to reduce the length of utterances T , yielding the length $T/4$. The ASR attention network used location-aware attention [7], which is more robust to long sequences than dot-product [26] or additive attention [27]. For decoding, we used a beam search algorithm [9] with beam size of 20. We manually set the maximum and minimum lengths of the output sequence to 0.3 and 0.8 times the length of the subsampled input sequence, respectively. Details of the experimental conditions for the ASR model are shown in Table 1.

Table 1: Experimental conditions for the ASR model.

Encoder type	BLSTMP
# encoder layers	8
# encoder units	320
# projection units	320
Decoder type	LSTM
# decoder layers	1
# decoder units	320
# dimension in attention	300
# filter in attention	10
Filter size in attention	100
Learning rate	1.0
Gradient clipping norm	5
Batch size	50
Maximum epoch	30 (for initial training) 15 (for retraining)
Optimization method	AdaDelta [28]
AdaDelta ρ	0.95
AdaDelta ϵ	10^{-8}
AdaDelta ϵ decay rate	10^{-2}
Beam size	20
Maximum length	0.8
Minimum length	0.3

The architecture of the TTE model followed the original Tacotron2 settings [22]. The input characters were converted into 512-dimensional character embeddings. The TTE encoder consisted of a three-layered 1D convolutional neural network (CNN) containing 512 filters with the shape 5, a batch normalization and rectified linear unit (ReLU) activation function, and an one-layered BLSTM with 512 units (256 units for forward processing, the rest for backward processing). Although the attention mechanism of the TTE model was based on location-aware attention [7], we additionally cumulated the attention weight feedback to next step to accelerate attention learning. The TTE decoder consisted of a two-layered LSTM with 1024 units. Prenet was a two-layered feed forward network with 256 units and ReLU. Postnet was a five-layered CNN containing 512 filters with the shape 5, a batch normalization, and tanh activation function except in the final layer. Dropout [30] with a probability of 0.5 was applied to all of the convolution and Prenet layers. Zoneout [31] with a probability of 0.1 was applied to the decoder LSTM. During generation, we applied dropout to Prenet in the same manner as in [22], and set the threshold value of the probability of the end of sequence at 0.75 to prevent from cutting off the end of the input sequence. Details of the experimental conditions for the TTE model are shown in Table. 2.

All of the networks were trained using the end-to-end speech processing toolkit ESPnet [32] with a single GPU (Titan X pascal). Character error rate (CER) and word error rate (WER) were used as metrics.

3.2. Experimental results

First, we focus on the effectiveness of adding the L1 norm to the objective function of the TTE model. Mean square error loss for validation data with teacher forcing is shown in Table 3. We can confirm that the use of the L1 norm results

Table 2: Experimental conditions for the TTE model.

Encoder type	CNN + BLSTM
# embedding dimension	512
# encoder layers	3 (CNN) 1 (BLSTM)
# encoder BLSTM units	512
# encoder CNN filters	512
Encoder CNN filter size	5
Decoder type	LSTM
# decoder layers	2
# decoder units	1024
# dimension in attention	128
# filters in attention	32
Filter size in attention	31
# Prenet layers	2
# Prenet units	256
# Postnet layers	5
# Postnet filters	512
Postnet filter size	5
Dropout rate	0.5
Zoneout rate	0.1
Learning rate	10^{-3}
Gradient clipping norm	1
Batch size	50
Maximum epoch	100
Optimization method	Adam [29]
Adam ϵ	10^{-6}
Threshold to stop generation	0.75

Table 3: MSE of the TTE model for validation set.

With L1 norm	0.0298
Without L1 norm	0.0335

in improved performance. Furthermore, we found that use of the L1 norm also leads to much faster attention learning. The attention weights for the validation data are shown in Fig. 4. While the TTE model without the L1 norm is unable to learn the attention until after epoch 40, use of the L1 norm make the model to learn the attention in less than 1/3 the number of epochs. This is because use of the L1 norm makes the model focus on reducing smaller error, which prevents the decoder of the model from becoming something like an auto-encoder.

Next, we focus on the effectiveness of our proposed data augmentation method. Our experimental results are shown in Table 4. Compared with the baseline, we can confirm that our proposed ‘‘Retrain-Joint’’ approach improved the recognition performance. However, when only hidden states were used during retraining, no improvement was observed. This is because using only the hidden states resulted in overfitting. However, in comparison to the oracle results, we can

Table 4: ASR performance using various retraining methods.

	CER / WER [%]	
	Validation	Evaluation
Baseline	11.2 / 24.9	11.1 / 25.2
Retrain-State	12.0 / 27.6	12.4 / 28.3
Retrain-State-Frozen	11.9 / 27.1	11.9 / 27.6
Retrain-Joint	10.3 / 23.5	10.3 / 23.6
Oracle-State	7.6 / 16.8	8.7 / 18.4
Oracle-State-Frozen	8.1 / 18.9	8.5 / 19.6
Oracle-Feature	4.7 / 11.4	4.6 / 11.8

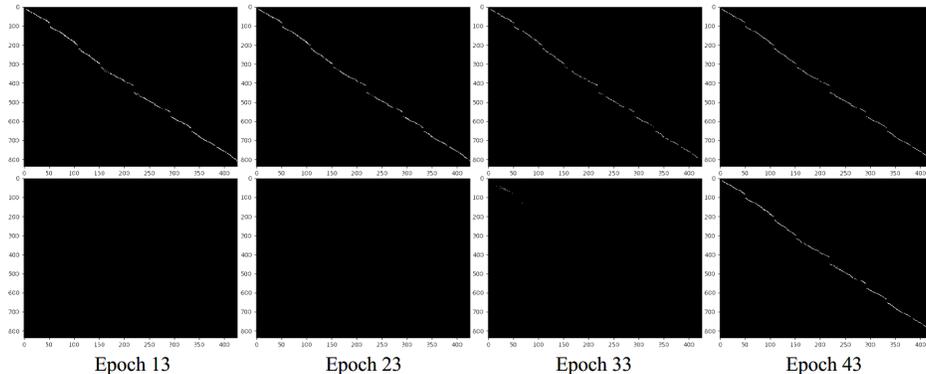


Fig. 4: Visualization of attention weight for validation data. Upper figures are w/ L1 norm, bottom ones are w/o L1 norm.

Table 5: Some recognition examples including unknown words before retraining.

GT	... in a reclining attitude being RIGIDLY bound both hands and feet by strong and painful withs
ORG	... in a reclining attitude being RIGILLY bound both hands and feet by strong and painful with
RET	... in a reclining attitude being RIGIDLY bound both hands and feet by strong and painful withs
GT	the first of our vague but INDUBITABLE data is that there is knowledge of the past
ORG	the first of our vague but INDUPINABLE data as that there is knowledge of the past
RET	the first of our vague but INDUBITABLE data it set there is knowledge of the past
GT	... children that ran about and prattled when they were in the woods looking for wild STRAWBERRIES
ORG	... children that ran about and pratelled when they were in the wood slooking for wild STRAWBERRES
RET	... children that ran about and prattled when they were in the woods looking for wild STRAWBERRIES

Table 6: ASR performance of shallow fusion with LM.

	CER / WER [%]	
	Validation	Evaluation
Baseline + LM	10.6 / 23.0	10.4 / 22.9
Retrain-Joint + LM	9.8 / 21.6	10.0 / 22.0

see that there is a still room for improvement. These results imply that the use of data of various speakers is more important than the use of various text. Since hidden states contains less information about speaker characteristics than acoustic features, using hidden states at the targets of the TTE model likely results in the generated hidden states representing the characteristics of an intermediate speaker. As a result, there is not enough speaker variation among the generated hidden states, degrading the effectiveness of data augmentation. To address this issue, we will extend our scheme using multi-speaker Tacotron2 [23] in future work.

Some recognition examples including unknown words before retraining are shown in Table 5. We can see that our proposed data augmentation method can improve the performance of the ASR decoder as language model, making it possible to extend the vocabulary. A similar effect was observed in the original back-translation work [15].

Finally, the results of shallow fusion with a character-based language model (LM) [14] are shown in Table 6, where the LM was trained using text of 360 hours of clean speech, and the balancing weight parameter between two models was decided to achieve the best recognition performance on CER. We can see that the use of LM improved the recognition per-

formance in both cases, indicating that our proposed method can still be combined with LM integration methods.

4. CONCLUSION

In this paper we proposed a novel data augmentation method for attention-based E2E-ASR, utilizing a large amount of text which was not paired with speech signals, an approach inspired by the back-translation technique has been proposed in the field of machine translation. We built a neural text-to-encoder (TTE) model which predicted a sequence of hidden states extracted by a pre-trained E2E-ASR decoder from a sequence of characters. Using the hidden states as targets makes it possible to achieve faster attention learning and reduces computational cost thanks to sub-sampling in the ASR encoder. After training, the TTE model generated the hidden states from a large amount of unpaired text, and then the decoder of the E2E-ASR model was retrained using the generated hidden states as additional training data. An experimental evaluation using LibriSpeech dataset demonstrated that our proposed method achieved the improvement of ASR performance and made it possible to improve the recognition results due to the smaller number of unknown words. Furthermore, we could confirm that our proposed method can be combined with language model integration methods.

In future works, we will extend the text-to-encoder model to multi-speaker model using speaker embedding vector [23] to generate more variable hidden states, and apply our proposed method using much larger amount of unpaired text.

5. REFERENCES

- [1] Frederick Jelinek, "Continuous speech recognition by statistical methods," *Proceedings of the IEEE*, vol. 64, no. 4, pp. 532–556, 1976.
- [2] Jan Chorowski, Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio, "End-to-end continuous speech recognition using attention-based recurrent NN: First results," *arXiv preprint arXiv:1412.1602*, 2014.
- [3] Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber, "Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks," in *Proceedings of the 23rd international conference on Machine learning*. ACM, 2006, pp. 369–376.
- [4] Alex Graves and Navdeep Jaitly, "Towards end-to-end speech recognition with recurrent neural networks," in *International Conference on Machine Learning*, 2014, pp. 1764–1772.
- [5] Dario Amodei, Sundaram Ananthanarayanan, Rishita Anubhai, Jingliang Bai, Eric Battenberg, Carl Case, Jared Casper, Bryan Catanzaro, Qiang Cheng, Guoliang Chen, et al., "Deep speech 2: End-to-end speech recognition in english and mandarin," in *International Conference on Machine Learning*, 2016, pp. 173–182.
- [6] Hagen Soltau, Hank Liao, and Hasim Sak, "Neural speech recognizer: Acoustic-to-word LSTM model for large vocabulary speech recognition," *arXiv preprint arXiv:1610.09975*, 2016.
- [7] Jan K Chorowski, Dzmitry Bahdanau, Dmitriy Serdyuk, Kyunghyun Cho, and Yoshua Bengio, "Attention-based models for speech recognition," in *Advances in neural information processing systems*, 2015, pp. 577–585.
- [8] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio, "Learning phrase representations using RNN encoder-decoder for statistical machine translation," *arXiv preprint arXiv:1406.1078*, 2014.
- [9] Ilya Sutskever, Oriol Vinyals, and Quoc V Le, "Sequence to sequence learning with neural networks," in *Advances in neural information processing systems*, 2014, pp. 3104–3112.
- [10] Tara Sainath and Yonghui Wu, "Improving end-to-end models for speech recognition," <https://ai.googleblog.com/2017/12/improving-end-to-end-models-for-speech-recognition.html>, 2017.
- [11] Jan Chorowski and Navdeep Jaitly, "Towards better decoding and language model integration in sequence to sequence models," *arXiv preprint arXiv:1612.02695*, 2016.
- [12] Caglar Gulcehre, Orhan Firat, Kelvin Xu, Kyunghyun Cho, Loic Barrault, Huei-Chi Lin, Fethi Bougares, Holger Schwenk, and Yoshua Bengio, "On using monolingual corpora in neural machine translation," *arXiv preprint arXiv:1503.03535*, 2015.
- [13] Anuroop Sriram, Heewoo Jun, Sanjeev Satheesh, and Adam Coates, "Cold fusion: Training seq2seq models together with language models," *arXiv preprint arXiv:1708.06426*, 2017.
- [14] Takaaki Hori, Shinji Watanabe, Yu Zhang, and William Chan, "Advances in joint CTC-attention based end-to-end speech recognition with a deep CNN encoder and RNN-LM," *arXiv preprint arXiv:1706.02737*, 2017.
- [15] Rico Sennrich, Barry Haddow, and Alexandra Birch, "Improving neural machine translation models with monolingual data," *arXiv preprint arXiv:1511.06709*, 2015.
- [16] Guillaume Lample, Ludovic Denoyer, and Marc'Aurelio Ranzato, "Unsupervised machine translation using monolingual corpora only," *arXiv preprint arXiv:1711.00043*, 2017.
- [17] Marcin Junczys-Dowmunt and Roman Grundkiewicz, "Log-linear combinations of monolingual and bilingual neural machine translation models for automatic post-editing," *arXiv preprint arXiv:1605.04800*, 2016.
- [18] Andros Tjandra, Sakriani Sakti, and Satoshi Nakamura, "Listening while speaking: Speech chain by deep learning," in *Automatic Speech Recognition and Understanding Workshop (ASRU), 2017 IEEE*. IEEE, 2017, pp. 301–308.
- [19] Adithya Renduchintala, Shuoyang Ding, Matthew Wiesner, and Shinji Watanabe, "Multi-modal data augmentation for end-to-end asr," *arXiv preprint arXiv:1803.10299*, 2018.
- [20] Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur, "Librispeech: an ASR corpus based on public domain audio books," in *Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on*. IEEE, 2015, pp. 5206–5210.
- [21] Paul J Werbos, "Backpropagation through time: what it does and how to do it," *Proceedings of the IEEE*, vol. 78, no. 10, pp. 1550–1560, 1990.

- [22] Jonathan Shen, Ruoming Pang, Ron J Weiss, Mike Schuster, Navdeep Jaitly, Zongheng Yang, Zhifeng Chen, Yu Zhang, Yuxuan Wang, RJ Skerry-Ryan, et al., “Natural TTS synthesis by conditioning wavenet on mel spectrogram predictions,” *arXiv preprint arXiv:1712.05884*, 2017.
- [23] Ye Jia, Yu Zhang, Ron J Weiss, Quan Wang, Jonathan Shen, Fei Ren, Zhifeng Chen, Patrick Nguyen, Ruoming Pang, Ignacio Lopez Moreno, et al., “Transfer learning from speaker verification to multispeaker text-to-speech synthesis,” *arXiv preprint arXiv:1806.04558*, 2018.
- [24] Daniel Povey, Arnab Ghoshal, Gilles Boulianne, Lukas Burget, Ondrej Glembek, Nagendra Goel, Mirko Hannemann, Petr Motlicek, Yanmin Qian, Petr Schwarz, et al., “The kaldı speech recognition toolkit,” in *IEEE 2011 workshop on automatic speech recognition and understanding*. IEEE Signal Processing Society, 2011, number EPFL-CONF-192584.
- [25] Haşim Sak, Andrew Senior, and Françoise Beaufays, “Long short-term memory recurrent neural network architectures for large scale acoustic modeling,” in *Fifteenth annual conference of the international speech communication association*, 2014.
- [26] Minh-Thang Luong, Hieu Pham, and Christopher D Manning, “Effective approaches to attention-based neural machine translation,” *arXiv preprint arXiv:1508.04025*, 2015.
- [27] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio, “Neural machine translation by jointly learning to align and translate,” *arXiv preprint arXiv:1409.0473*, 2014.
- [28] Matthew D Zeiler, “ADADELTA: an adaptive learning rate method,” *arXiv preprint arXiv:1212.5701*, 2012.
- [29] Diederik P Kingma and Jimmy Ba, “Adam: A method for stochastic optimization,” *arXiv preprint arXiv:1412.6980*, 2014.
- [30] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov, “Dropout: a simple way to prevent neural networks from overfitting,” *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.
- [31] David Krueger, Tegan Maharaj, János Kramár, Mohammad Pezeshki, Nicolas Ballas, Nan Rosemary Ke, Anirudh Goyal, Yoshua Bengio, Aaron Courville, and Chris Pal, “Zoneout: Regularizing rnns by randomly preserving hidden activations,” *arXiv preprint arXiv:1606.01305*, 2016.
- [32] Shinji Watanabe, Takaaki Hori, Shigeaki Karita, Tomoki Hayashi, Jiro Nishitoba, Yuya Unno, Nelson Enrique Yalta Soplın, Jahn Heymann, Matthew Wiesner, Nanxin Chen, et al., “ESPnet: End-to-end speech processing toolkit,” *arXiv preprint arXiv:1804.00015*, 2018.