INTERSPEECH 2020
October 25–29, 2020, Shanghai, China

Language Model Data Augmentation Based on Text Domain Transfer

Atsunori Ogawa, Naohiro Tawara, and Marc Delcroix

NTT Corporation, Japan

atsunori.ogawa.gx@hco.ntt.co.jp

Abstract

To improve the performance of automatic speech recognition (ASR) for a specific domain, it is essential to train a language model (LM) using text data of the target domain. In this study, we propose a method to transfer the domain of a large amount of source data to the target domain and augment the data to train a target domain-specific LM. The proposed method consists of two steps, which use a bidirectional long short-term memory (BLSTM)-based word replacing model and a target domain-adapted LSTMLM, respectively. Based on the learned domain-specific wordings, the word replacing model converts a given source domain sentence to a confusion network (CN) that includes a variety of target domain candidate word sequences. Then, the LSTMLM selects a target domain sentence from the CN by evaluating its grammatical correctness based on decoding scores. In experiments using lecture and conversational speech corpora as the source and target domain data sets, we confirmed that the proposed LM data augmentation method improves the target conversational speech recognition performance of a hybrid ASR system using an n-gram LM and the performance of N-best rescoring using an LSTMLM.

Index Terms: speech recognition, language modeling, data augmentation, domain transfer, text generation

1. Introduction

Based on the recent introduction of state-of-the-art neural network (NN) modeling, the performance of automatic speech recognition (ASR) has been greatly improved [1, 2] and various types of ASR-based applications, including voice search services and smart speakers, have been actively developed. Despite this great progress, since ASR is inherently a highly domain-dependent technology, its performance for some low-resourced domains, such as conversational speech recognition and low-resourced language ASR, remains at an unsatisfactory level [3–6].

To improve the ASR performance for such low-resourced domains, a simple but promising way is to augment the data of the target domain. Training data augmentation for ASR acoustic modeling has been performed extensively [7, 8]. In this study, we focus on text data augmentation for ASR language modeling [9–17]. The simplest and most popular method is to use a large amount of text data from a well-resourced domain in addition to a small amount of text data from the target domain [9–11]. In this method, domain-specific n-gram language models (LMs) are first trained using each domain's text data and then linearly interpolated using mixing weights that are optimized to reduce the development data perplexity. Another popular method is based on web searching [12–14]. In this method, queries that represent the target domain are cast to a search engine to collect documents that are relevant to the target domain. Machine translation (MT) has also been used to translate a large amount of text data of a well-resourced language to the corresponding text data in a low-resourced language [15–17]. Good ASR performance improvements are reported with these methods, although they have some drawbacks. The performance of the n-gram LM interpolation method depends on how close the domain of the additional data is to the target domain. The web search-based method needs careful data cleaning (text normalization) since the collected web documents are written in a wide variety of formats. The MT-based method requires parallel data of the source and target language pairs to train an MT system.

As in the conventional methods described above, in this study, we assume a very common situation where a small amount of target domain text data and a large amount of another (source) domain text data are available. In this situation, we propose a method that transfers the domain of a large amount of source data to the target domain and augments the data to train a target domain-specific LM. Our method can thus be used in combination with the widely used n-gram LM interpolation method [9–11] to obtain a better target domain LM.

Our proposed method consists of two steps, which use a bidirectional long short-term memory (BLSTM)-based word replacing model [18] and a target domain-adapted LSTMLM, respectively. By training the word replacing model using source and target domain data sets with the domain label, it learns domain-specific wordings (Section 2.1). Using the model with a sampling method [19], we convert a given source domain sentence to a confusion network (CN) [20], which includes a variety of target domain candidate word sequences (Section 2.2). Then, by using the LSTMLM, which can evaluate the validity of a long word sequence, we perform decoding on the CN and select a target domain sentence that is expected to be grammatically correct (Section 2.3). To confirm the effectiveness of the proposed method, we use lecture and conversational speech corpora as the source and target domain data sets and conduct experiments to augment the data to train an n-gram LM used in a deep NN/hidden Markov model hybrid ASR system and an LSTMLM used in N-best rescoring (Section 4).

2. Proposed LM data augmentation method

We describe in detail the proposed LM data augmentation method. We first describe the BLSTM-based word replacing model [18], which was originally proposed for text data augmentation in text classification tasks. Then, we describe the two-step text generation procedure, i.e. CN generation by word replacing with sampling and sentence selection from the CN using an LSTMLM.

2.1. BLSTM-based word replacing model

Figure 1 shows the domain-conditioned word prediction procedure using the BLSTM-based word replacing model [18]. As with a BLSTMLM [21], given a sentence (word sequence) $W = w_{1:T} = w_1, \cdots, w_T$ of length (number of words) T, the model estimates word probability distributions for each time step $t = 1, \cdots, T$.

Given a forward (backward) partial word sequence $w_{1:t-1} = w_1, \cdots, w_{t-1}$ ($w_{T:t+1} = w_T, \cdots, w_{t+1}$), a forward (backward) LSTM unit recursively estimates forward (backward) hidden state vectors and, as a result, provides the hidden state vector at time step $t-1$ ($t+1$) as,

$$\overrightarrow{\mathbf{h}}_{t-1} = \texttt{fwlstm}(w_{t-1}, \overrightarrow{\mathbf{h}}_{t-2}), \quad (1)$$

$$\overleftarrow{\mathbf{h}}_{t+1} = \texttt{bwlstm}(w_{t+1}, \overleftarrow{\mathbf{h}}_{t+2}). \quad (2)$$

http://dx.doi.org/10.21437/Interspeech.2020-1524

Then, these hidden state vectors and a scalar value d are concatenated as,

$$\mathbf{h}_t^d = \texttt{concat}(\overrightarrow{\mathbf{h}}_{t-1}, \overleftarrow{\mathbf{h}}_{t+1}, d), \tag{3}$$

where d is the domain label and $\mathbf{h}_t^d$ is the domain-conditioned hidden state vector at time step t. Based on our experimental settings described in Section 4, we define d as,

$$d = \begin{cases} 0, & \text{Lecture (source domain)}, \\ 1, & \text{Conversational (target domain)}. \end{cases} \tag{4}$$

$\mathbf{h}_t^d$ is then input to a linear layer, which is followed by the softmax activation function, and finally, we obtain the domain-conditioned word probability distribution at time step t as,

$$\mathbf{z}_t^d = \texttt{linear}(\mathbf{h}_t^d), \tag{5}$$

$$P(\hat{w}_t | W \setminus \{w_t\}, d) = \texttt{softmax}(\mathbf{z}_t^d)_{\texttt{idx}(\hat{w}_t)}, \tag{6}$$

where $\mathbf{z}_t^d$ is the domain-conditioned logit vector (its dimension equals the vocabulary size V), $\hat{w}_t$ is the predicted word at time step t, $\texttt{idx}(\hat{w}_t)$ is the index of $\hat{w}_t$ in the vocabulary, and $W \setminus \{w_t\}$ is the given sentence W without w_t. The vocabulary is defined to cover all the words included in the two domain text corpora.

In the model training, we first pretrain the model using the two domain text corpora without the domain label, and then we fine-tune the model again using the two domain text corpora with the domain label. With this fine-tuning using the domain label, the model learns the domain-specific wordings. For example, as shown in Fig. 1, given the forward partial word sequence, $w_{1:t-1} = \{\cdots, i, like\}$, and the backward partial word sequence, $w_{T:t+1} = \{\cdots, much, very\}$, the model gives high probabilities for the words, e.g. *asr*, *tts*, and *dnn*, when the domain label is set at 0 (lecture). In contrast, it gives high probabilities for the words, e.g. *movie*, *golf*, and *cooking*, when the domain label is set at 1 (conversational).

2.2. CN generation by word replacing and sampling

In the text generation, we input a source lecture domain ($d = 0$) sentence to the model but change its domain label to conversational ($d = 1$). As a result of word replacing for each time step, we obtain a sequence of domain-conditioned word probability distributions. Then, by selecting the highest probability words from the sequence, we can obtain a target conversational domain ($d = 1$) sentence.

However, with this procedure, we can generate only one target domain sentence from one input source domain sentence. To generate the desired number of target domain sentences from one input source domain sentence, we employ a sampling method that is based on a Gumbel distribution [19]. We sample i.i.d. samples V times from the normal Gumbel distribution and make a V-dimensional sample vector as, $\mathbf{g} = [g_1, \cdots, g_V]$, where $g_i \sim \text{Gumbel}(0, 1)$. Then, we add $\mathbf{g}$ to $\mathbf{z}_t^d$ and obtain a perturbed version of the domain-conditioned word probability distribution at time step t as,

$$\tilde{\mathbf{z}}_t^d = (\mathbf{z}_t^d + \mathbf{g})/\tau, \tag{7}$$

$$\tilde{P}(\hat{w}_t | W \setminus \{w_t\}, d) = \texttt{softmax}(\tilde{\mathbf{z}}_t^d)_{\texttt{idx}(\hat{w}_t)}, \tag{8}$$

where τ is the temperature parameter ($\tau > 0$). By repeating this procedure for each time step and selecting the highest probability word at each time step, we can generate one target domain sentence from one source domain sentence. Then, by repeating the above procedure M times, we can generate M (= the desired number of) target domain sentences from one source domain sentence. Note that, if we use a smaller τ value, higher probability words tend to be selected at each time step (small variety). In contrast, if we use a larger τ value, other words will also have a chance to be selected (large variety). We need to adjust τ to generate target domain sentences with a reasonable level of variety.

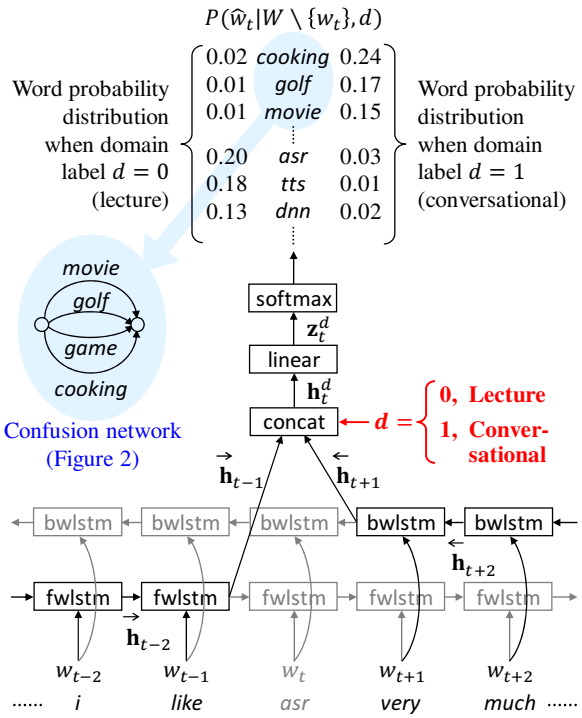

Figure 1: *Domain-conditioned word prediction procedure at time step t using the BLSTM-based word replacing model.*

With the above procedure, we can generate the desired number of target domain sentences from one input source domain sentence. However, in the above procedure, since words are selected individually at each time step, the generated sentences are not necessarily grammatically correct. To improve the grammatical correctness of the generated sentences, we perform sentence selection using an LSTMLM as described in Section 2.3. To perform it efficiently, we convert a sequence of word probability distributions to a confusion network (CN) [20]. Given the word probability distribution at a time step, we keep only higher probability words and prune the other lower probability words, as shown on the left of Fig. 1. The pruning threshold is defined based on the number of words to keep (K) and the cumulative probability of the kept words (Q). The word probabilities are normalized within the kept words. By repeating this procedure for each time step, we can obtain a CN that includes a variety of target domain candidate word sequences, as shown in the lower half of Fig. 2.

2.3. Sentence selection from CN using LSTMLM

As shown in the upper half of Fig. 2, we select a grammatically correct target domain sentence from the CN by performing decoding (beam search) on the CN using a unidirectional LSTMLM that can evaluate the validity of a long word sequence. The LSTMLM is adapted to the target domain using the target domain text data. We define the score (log probability) function of the partial word sequence (hypothesis) $w_{1:t}$ during decoding as,

$$\log P(w_{1:t}) = \lambda \log P_{\texttt{lstm}}(w_t | w_{1:t-1}) + (1 - \lambda) \log P_{\texttt{cn}}(w_t) + \log P(w_{1:t-1}), \tag{9}$$

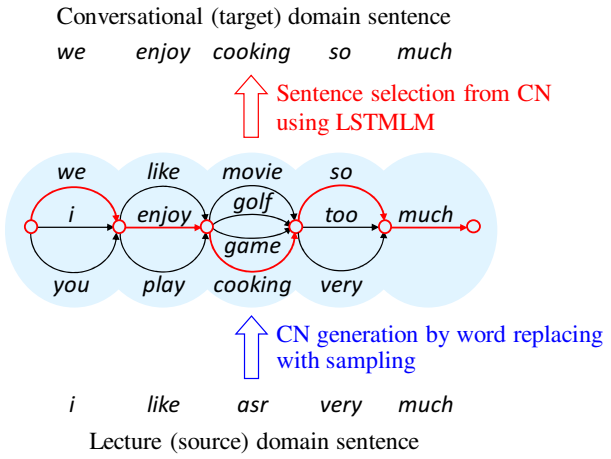

Figure 2: *Procedure for generating a conversational (target) domain sentence from a lecture (source) domain sentence.*

where $P_{\texttt{lstm}}(w_t|w_{1:t-1})$ is the probability of the word w_t predicted by the LSTMLM given the word history $w_{1:t-1}$, $P_{\texttt{cn}}(w_t)$ is the probability of w_t attached to the CN, $P(w_{1:t-1})$ is the probability of the hypothesis $w_{1:t-1}$, and λ $(0 \leq \lambda \leq 1)$ is the weight that balances $P_{\texttt{lstm}}(w_t|w_{1:t-1})$ and $P_{\texttt{cn}}(w_t)$. Using the above-defined score function, we perform decoding with hypothesis pruning that is based on the maximum number of hypotheses after the pruning (B). Then, at time step T, we can obtain a target domain sentence as the highest score hypothesis. If we use a larger λ value, the grammatical correctness of the selected sentences is emphasized. In contrast, if we use a smaller λ value, the variety of the selected sentences is increased.

Figure 2 summarizes the procedure for generating a conversational (target) domain sentence from a lecture (source) domain sentence. We repeat this procedure M times for one input source domain sentence and repeat them for all sentences included in the source domain corpus. As a result, we can generate a target domain corpus that is M times larger than the original source domain corpus.

3. Relation to prior work

The most relevant studies to our study are [22] and [23], which are based on neural text generation [24]. In these studies, neural LMs are adapted to the target domain using a small amount of target domain text data, and the adapted neural LMs are used to generate a large amount of target domain text data by receiving the prefix contexts. Then, using the generated data, domain-specific n-gram LMs are trained to improve the performance of hybrid ASR systems for the target domain. Good ASR performance improvements are reported in [22] and [23]. However, in these studies, text domain transfer is performed by using only the domain-adapted neural LMs, and their effectiveness is evaluated only with the n-gram LMs used in hybrid ASR systems (since the n-gram LM-based hybrid systems still show high ASR performance [25–27]).

In contrast to [22] and [23], in this study, we perform text domain transfer by using a BLSTM-based word replacing model [18], which learns domain-specific wordings, in combination with a domain-adapted LSTMLM, i.e. our method includes the methods of [22,23] (and that of [18]). If we set λ at 1 in Eq. (9), our method becomes close to [22, 23] (and, if we set λ at 0, our method equals [18]). We also evaluate the effectiveness of our method with LSTMLMs used for N-best rescoring in addition to the evaluation with the n-gram LMs.

As for rescoring, a recurrent NN LM trained with neural LM generated texts (as with [22, 23]) is used for rescoring in [28]. Good performance is reported, but in [28], text generation is performed in a closed domain scenario, which is different from our (and [22, 23]'s) targeting domain transfer scenario.

Table 1: *Details of the CSJ and NTT speech corpora.*

	Training		**Dev**	**Eval**
Corpus	CSJ	NTT	NTT	NTT
Domain	Lect.	Conv.	Conv.	Conv.
Hours	537	106	11	11
Number of sentences	413k	59k	4748	4646
Number of words	7M	396k	32k	32k
Ave. sent. len. (#words)	17.3	6.7	6.8	6.9
Vocabulary size	66.6k	11.1k	2381	2325

Table 2: *Structures of the BLSTM-based word replacing model and LSTMLMs for sentence selection and N-best rescoring.*

	Word replacing	**LSTMLM**
Number of LSTM layers	1	2
Number of LSTM nodes	1024	650
Number of linear layers	1	1
Number of linear nodes	1024	650
Softmax (=vocabulary) size	68.9k	68.9k

4. Experiments

To confirm the effectiveness of the proposed LM data augmentation method, we conducted experiments using two different domain speech corpora in two ASR scenarios, i.e. performing ASR with a hybrid system using n-gram LMs as with [22, 23] and performing N-best rescoring using LSTMLMs.

4.1. Experimental settings

As the source domain ($d = 0$) corpus, we used the corpus of spontaneous Japanese (CSJ) [29], which is a large scale lecture speech corpus. As the target domain ($d = 1$) corpus, we used an in-house Japanese conversational speech corpus called NTT meeting (NTT) [4], which contains casual conversations in group meetings. Table 1 shows the details of these corpora and their large different characteristics.

Using the CSJ and NTT training data sets, with the procedure described in Section 2.1, we trained a BLSTM-based word replacing model [18] with the structure shown in Table 2. We used PyTorch [30] for all the NN modeling in this study. We pretrained the model for five epochs without using the domain label and fine-tuned the model for another five epochs using the domain label. The final model was used for CN generation.

In the CN generation, with the procedure described in Section 2.2, we input sentences from the CSJ training data ($d = 0$) to the word replacing model with the target conversational domain label ($d = 1$) and generated target domain CNs. Then, with the procedure described in Section 2.3, we selected target domain sentences from the CNs using a unidirectional LSTMLM with the structure shown in Table 2. The LSTMLM was first trained using the CSJ and NTT training data sets for 20 epochs and then adapted using the NTT training data for one epoch with a small learning rate [22, 23]. From the results of preliminary experiments, we set the hyperparameters of the CN generation and sentence selection as, $\tau = 1$, $K = 5$, $Q = 0.8$, $\lambda = 0.3$, and $B = 5$. We generated M target domain sentences from one source domain sentence. As a result, we obtained the target domain generated CSJ training data (hereafter, referred to as GenCSJ) that is M times larger than the original CSJ training data. We set M at 1, 10, and 100, i.e. we obtained GenCSJx$\{1 \mid 10 \mid 100\}$.

Table 3: *Number 1 to 9 are results obtained with the hybrid ASR system using the nine trigram LMs. PPLs shown in parentheses cannot be compared with the others because of the different vocabulary sizes. Number 10 to 13 are 100-best rescoring results obtained with the four LSTMLMs applied to the 100-best hypotheses obtained with model 9, i.e. the best trigram LM results.*

No.	Type	Model	Vocab. size	Development data			Evaluation data		
				OOV	PPL	WER	OOV	PPL	WER
1.	Trigram	NTT	11.1k	1.42	(44.9)	21.8	1.34	(44.5)	21.3
2.	Trigram	CSJ	66.6k	2.07	(136.9)	23.0	2.04	(134.9)	22.2
3.	Trigram	GenCSJx1	32.1k	0.79	(85.7)	21.6	0.71	(89.2)	20.7
4.	Trigram	GenCSJx10	42.6k	0.51	(79.7)	20.7	0.47	(82.1)	19.9
5.	Trigram	GenCSJx100	43.1k	0.49	(76.2)	20.4	0.45	(78.3)	19.6
6.	Trigram	NTT + CSJ [9–11]	68.9k	0.39	46.4	20.2	0.40	45.3	19.4
7.	Trigram	NTT + CSJ + GenCSJx1	68.9k	0.39	46.4	20.0	0.40	45.4	19.1
8.	Trigram	NTT + CSJ + GenCSJx10	68.9k	0.39	45.1	19.8	0.40	44.1	18.9
9.	Trigram	NTT + CSJ + GenCSJx100	68.9k	0.39	44.3	**19.6**	0.40	43.3	**18.7**
10.	LSTMLM	NTT + CSJ	68.9k	0.39	28.0	18.8	0.40	27.6	17.9
11.	LSTMLM	NTT + CSJ + GenCSJx1	68.9k	0.39	27.6	18.4	0.40	27.3	17.7
12.	LSTMLM	NTT + CSJ + GenCSJx10	68.9k	0.39	27.7	18.3	0.40	27.0	17.5
13.	LSTMLM	NTT + CSJ + GenCSJx100	68.9k	0.39	27.7	**18.2**	0.40	27.4	**17.4**

We trained five trigram LMs with SRILM [10] using the CSJ training data, the NTT training data, and GenCSJx$\{1 \mid 10 \mid 100\}$, respectively. We applied Kneser-Ney smoothing [31] to these LMs. Hereafter, we refer to the LMs by their training data names. We also obtained four interpolated trigram LMs [9–11]. One was obtained by interpolating the two trigram LMs, NTT and CSJ, i.e. NTT+CSJ, and the other three were obtained by interpolating the three trigram LMs, NTT, CSJ, and GenCSJx$\{1 \mid 10 \mid 100\}$, i.e. NTT+CSJ+GenCSJx$\{1 \mid 10 \mid 100\}$. Their mixing weights were optimized to reduce the perplexity (PPL) of the NTT development data [10].

We also trained four unidirectional LSTMLMs, i.e. NTT+CSJ and NTT+CSJ+GenCSJx$\{1 \mid 10 \mid 100\}$, whose structure is shown in Table 2. We trained them using the corresponding data sets for 20 epochs and selected the models that showed the lowest PPLs for the NTT development data.

We performed ASR using the Kaldi hybrid ASR system [32]. A time delay NN-based acoustic model [33] was trained using the CSJ and NTT training data sets. Using this acoustic model and a trigram LM described above, one-pass decoding was performed using a weighted finite-state transducer [34]. We also performed 100 ($= N$)-best rescoring with Kaldi using an LSTMLM described above. We evaluated the nine trigram LMs and the four LSTMLMs with their out-of-vocabulary rates (OOVs), PPLs, and ASR word error rates (WERs) for the NTT development and evaluation data sets.

4.2. Experimental results

Table 3 shows the experimental results. We first look at the results obtained with the hybrid ASR system using the nine trigram LMs. Comparing GenCSJx1 (model 3) with CSJ (model 2), GenCSJx1 shows the lower OOVs and WERs than CSJ. From this result, we can confirm that the text domain transfer is successfully performed with our proposed method. We found that 38.5% of the words in the CSJ training data are replaced with other words to generate GenCSJx1. GenCSJx1 also outperforms NTT (model 1). Table 4 shows the effect of λ on the performance of GenCSJx1 (Section 2.3). From this table, we can confirm that, in the domain-transferred text generation, using both the word replacing model and the LSTMLM with an appropriate weight ($\lambda = 0.3$) is better than using only the word replacing model ($\lambda = 0.0$) [18] or using only the LSTMLM ($\lambda = 1.0$) [22, 23] (Section 3). Comparing GenCSJx$\{10 \mid 100\}$ (models 4 and 5) with GenCSJx1 (model 3), we can confirm the effectiveness of using the more generated text data. Then, comparing NTT+CSJ+GenCSXx$\{1 \mid 10 \mid 100\}$ (models 7, 8, and 9) with NTT+CSJ (model 6), we can confirm that the proposed method successfully improves the widely used n-gram LM interpolation method [9–11].

Table 4: *WERs obtained with GenCSJx1 by changing λ.*

Model	Dev	Eval
GenCSJx1 with $\lambda = 0.0$ [18]	21.8	21.2
GenCSJx1 with $\lambda = 0.3$ (model 3)	**21.6**	**20.7**
GenCSJx1 with $\lambda = 1.0$ [22, 23]	22.0	21.3

Next, we look at the 100-best rescoring results using the four LSTMLMs. Rescoring was performed on the best trigram LM results, i.e. the results obtained with model 9. Comparing the results of models 9 and 10, we can confirm the large effect of using the LSTMLM and performing rescoring. Comparing NTT+CSJ+GenCSJx$\{1 \mid 10 \mid 100\}$ (models 11, 12, and 13) with NTT+CSJ (model 10), we can confirm the steady WER reductions as with the trigram LM results. However, the effect of using the more generated text data is slightly reduced compared with the trigram LM results. We need to further investigate how to effectively utilize a large amount of generated texts in LSTMLM training. Interestingly, a lower PPL model does not necessarily show a lower WER. PPL is calculated using reference transcriptions but N-best rescoring is performed on ASR hypotheses that have ASR errors, i.e. the purpose of N-best rescoring is to find better ASR hypotheses, even though they may have errors [35, 36]. The generated texts are not always completely correct and may resemble errors of better ASR hypotheses. Therefore, an LSTMLM trained using the generated texts may perform robustly in N-best rescoring.

5. Conclusion and future work

We proposed an LM data augmentation method by generating domain-transferred texts using a BLSTM-based word replacing model [18] and an LSTMLM. We experimentally confirmed its effectiveness by performing ASR with a hybrid system using n-gram LMs and N-best rescoring using LSTMLMs. Future work will include the use of a more advanced deep Transformer model [37, 38] for text generation [23] and the evaluation in an end-to-end ASR system [39, 40].

6. References

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

[2] D. Yu and L. Deng, *Automatic speech recognition: A deep learning approach*. Springer-Verlag, London, 2015.

[3] T. Hain, L. Burget, J. Dines, G. Garau, V. Wan, M. Karafiat, J. Vepa, and M. Lincoln, "The AMI system for the transcription of speech in meetings," in *Proc. ICASSP*, 2007, pp. IV357–IV360.

[4] T. Hori, S. Araki, T. Yoshioka, M. Fujimoto, S. Watanabe, T. Oba, A. Ogawa, K. Otsuka, D. Mikami, K. Kinoshita, T. Nakatani, A. Nakamura, and J. Yamato, "Low-latency real-time meeting recognition and understanding using distant microphones and omni-directional camera," *IEEE TASLP*, vol. 20, no. 2, pp. 499–513, Feb. 2012.

[5] J. Barker, S. Watanabe, E. Vincent, and J. Trmal, "The fifth 'CHiME' speech separation and recognition challenge: Dataset, task and baselines," in *Proc. Interspeech*, 2018, pp. 1561–1565.

[6] M. Harper, "IARPA Babel program," https://www.iarpa.gov/index.php/research-programs/babel.

[7] R. P. Lippmann, E. A. Martin, and D. B. Paul, "Multi-style training for robust isolated-word speech recognition," in *Proc. ICASSP*, 1987, pp. 709–712.

[8] D. S. Park, W. Chan, Y. Zhang, C.-C. Chiu, B. Zoph, E. D. Cubuk, and Q. V. Le, "SpecAugment: A simple data augmentation method for automatic speech recognition," in *Proc. Interspeech*, 2019, pp. 2613–2617.

[9] F. Jelinek, *Statistical methods for speech recognition*. The MIT Press, Cambridge, MA, 1998.

[10] A. Stolcke, "SRILM – An extensible language modeling toolkit," in *Proc. ICSLP*, 2002, pp. 901–904.

[11] B.-J. Hsu, "Generalized linear interpolation of language models," in *Proc. ASRU*, 2007, pp. 549–552.

[12] I. Bulyko, M. Ostendorf, M. Siu, T. Ng, A. Stolcke, and Ö. Çetin, "Web resources for language modeling in conversational speech recognition," *ACM Transactions on Speech and Language Processing (TSLP)*, vol. 5, no. 1, Dec. 2007.

[13] A. Gandhe, L. Qin, F. Metze, A. Rudnicky, I. Lane, and M. Eck, "Using web text to improve keyword spotting in speech," in *Proc. ASRU*, 2013, pp. 428–433.

[14] G. Mendels, E. Cooper, V. Soto, J. Hirschberg, M. Gales, K. Knill, A. Ragni, and H. Wang, "Improving speech recognition and keyword search for low resource languages using web data," in *Proc. Interspeech*, 2015, pp. 829–832.

[15] A. Jensson, K. Iwano, and S. Furui, "Development of a speech recognition system for Icelandic using machine translated text," in *Proc. SLTU*, 2008, pp. 18–21.

[16] H. Cucu, A. Buzo, L. Besacier, and C. Burileanu, "SMT-based ASR domain adaptation methods for under-resourced languages: Application to Romanian," *Speech Communication*, vol. 56, pp. 195–212, Jan. 2014.

[17] A. Gorin, R. Lileikytė, G. Huang, L. Lamel, J.-L. Gauvain, and A. Laurent, "Language model data augmentation for keyword spotting in low-resourced training conditions," in *Proc. Interspeech*, 2016, pp. 775–779.

[18] S. Kobayashi, "Contextual augmentation: Data augmentation by words with paradigmatic relations," in *Proc. NAACL-HLT*, 2018, pp. 452–457.

[19] E. Jang, S. Gu, and B. Poole, "Categorical reparameterization with Gumbel-Softmax," in *Proc. ICLR*, 2017.

[20] L. Mangu, E. Brill, and A. Stolcke, "Finding consensus in speech recognition: Word error minimization and other applications of confusion networks," *Computer Speech and Language*, vol. 14, no. 4, pp. 373–400, Oct. 2000.

[21] E. Arisoy, A. Sethy, B. Ramabhadran, and S. Chen, "Bidirectional recurrent neural network language models for automatic speech recognition," in *Proc. ICASSP*, 2015, pp. 5421–5425.

[22] M. Suzuki, N. Itoh, T. Nagano, G. Kurata, and S. Thomas, "Improvements to n-gram language model using text generated from neural language model," in *Proc. ICASSP*, 2019, pp. 7245–7249.

[23] Y. Wang, H. Huang, Z. Liu, Y. Pang, Y. Wang, C. Zhai, and F. Peng, "Improving n-gram language models with pre-trained deep transformer," *arXiv:1911.10235v1 [cs.CL]*.

[24] Z. Xie, "Neural text generation: A practical guide," *arXiv:1711.09534v1 [cs.CL]*.

[25] G. Saon, G. Kurata, T. Sercu, K. Audhkhasi, S. Thomas, D. Dimitriadis, X. Cui, B. Ramabhadran, M. Picheny, L.-L. Lim, B. Roomi, and P. Hall, "English conversational telephone speech recognition by humans and machines," in *Proc. Interspeech*, 2017, pp. 132–136.

[26] W. Xiong, L. Wu, F. Alleva, J. Droppo, X. Huang, and A. Stolcke, "The Microsoft 2017 conversational speech recognition system," in *Proc. ICASSP*, 2018, pp. 5934–5938.

[27] C. Lüscher, E. Beck, K. Irie, M. Kitza, W. Michel, A. Zeyer, R. Schlüter, and H. Ney, "RWTH ASR systems for LibriSpeech: Hybrid vs attention-w/o data augmentation," in *Proc. Interspeech*, 2019, pp. 231–235.

[28] E. Yilmaz, H. van den Heuvel, and D. A. van Leeuwen, "Acoustic and textual data augmentation for improved ASR of code-switching speech," in *Proc. Interspeech*, 2018, pp. 1933–1937.

[29] K. Maekawa, "Corpus of spontaneous Japanese: Its design and evaluation," in *Proc. Workshop on Spontaneous Speech Processing and Recognition (SSPR)*, 2003, pp. 7–12.

[30] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Köpf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "PyTorch: An imperative style, high-performance deep learning library," in *Proc. NeurIPS*, 2019, pp. 8024–8035.

[31] R. Kneser and H. Ney, "Improved backing-off for M-gram language modeling," in *Proc. ICASSP*, 1995, pp. 181–184.

[32] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlíček, Y. Qian, P. S. J. Silovský, G. Stemmer, and K. Veselý, "The Kaldi speech recognition toolkit," in *Proc. ASRU*, 2011.

[33] V. Peddinti, D. Povey, and S. Khudanpur, "A time delay neural network architecture for efficient modeling of long temporal contexts," in *Proc. Interspeech*, 2015, pp. 3214–3218.

[34] C. Allauzen, M. Riley, J. Schalkwyk, W. Skut, and M. Mohri, "OpenFst: A general and efficient weighted finite-state transducer library," in *Proc. CIAA*, 2007, pp. 11–23.

[35] T. Oba, T. Hori, A. Nakamura, and A. Ito, "Round-robin duel discriminative language models," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 20, no. 4, pp. 1244–1255, May 2012.

[36] A. Ogawa, M. Delcroix, S. Karita, and T. Nakatani, "Improved deep duel model for rescoring N-best speech recognition list using backward LSTMLM and ensemble encoders," in *Proc. Interspeech*, 2019, pp. 3900–3904.

[37] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, and Ł. Kaiser, "Attention is all you need," in *Proc. NIPS*, 2017, pp. 5998–6008.

[38] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional Transformers for language understanding," in *Proc. NAACL-HLT*, 2019, pp. 4171–4186.

[39] S. Watanabe, T. Hori, S. Karita, T. Hayashi, J. Nishitoba, Y. Unno, N. E. Y. Soplin, J. Heymann, M. Wiesner, N. Chen, A. Renduchintala, and T. Ochiai, "ESPnet: End-to-end speech processing toolkit," in *Proc. Interspeech*, 2018, pp. 2207–2211.

[40] T. Hori, J. Cho, and S. Watanabe, "End-to-end speech recognition with word-based RNN language models," in *Proc. SLT*, 2012, pp. 113–118.