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

Semi-Supervised Learning with Data Augmentation for End-to-End ASR

Felix Weninger[1], *Franco Mana*[2], *Roberto Gemello*[2], *Jesús Andrés-Ferrer*[3], *Puming Zhan*[1]
[1]Nuance Communications, Inc., Burlington, MA, USA
[2]Nuance Communications, Torino, Italy
[3]Nuance Communications, Valencia, Spain
{felix.weninger,franco.mana,roberto.gemello,jesusandres.ferrer,puming.zhan}@nuance.com

Abstract

In this paper, we apply Semi-Supervised Learning (SSL) along with Data Augmentation (DA) for improving the accuracy of End-to-End ASR. We focus on the consistency regularization principle, which has been successfully applied to image classification tasks, and present sequence-to-sequence (seq2seq) versions of the FixMatch and Noisy Student algorithms. Specifically, we generate the pseudo labels for the unlabeled data on-the-fly with a seq2seq model after perturbing the input features with DA. We also propose soft label variants of both algorithms to cope with pseudo label errors, showing further performance improvements. We conduct SSL experiments on a conversational speech data set (doctor-patient conversations) with 1.9 kh manually transcribed training data, using only 25 % of the original labels (475 h labeled data). In the result, the Noisy Student algorithm with soft labels and consistency regularization achieves 10.4 % word error rate (WER) reduction when adding 475 h of unlabeled data, corresponding to a recovery rate of 92 %. Furthermore, when iteratively adding 950 h more unlabeled data, our best SSL performance is within 5 % WER increase compared to using the full labeled training set (recovery rate: 78 %).
Index Terms: automatic speech recognition, semi-supervised learning, data augmentation, sequence-to-sequence, end-to-end

1. Introduction

End-to-end (E2E) systems have become a focus of ASR research in recent years, due to their ability of integrating all components of an ASR system in a single deep neural network (DNN), which greatly simplifies and unifies the training and decoding process [1–5]. The Sequence-to-Sequence (seq2seq) model with attention is one of the model architectures for E2E ASR systems which has shown state-of-the-art performance [6–10]. However, a general observation is that E2E ASR needs large amounts of training data for achieving state-of-the-art performance, especially when no language model trained with external text data is included in the system. Data Augmentation (DA) and semi-supervised learning (SSL) are two approaches that can be used for improving E2E model performance with limited amounts of manually transcribed training data.

DA perturbs (usually randomly) the input data without altering the corresponding labels. This not only increases the variety of the data, but also serves as implicit regularization to avoid overfitting [11]. It has been successfully used for both conventional [12–14] and E2E ASR systems [15, 16]. In particular, the SpecAugment approach proposed in [17] has shown impressive improvement for seq2seq based E2E ASR models.

SSL (also called semi-supervised training) aims at leveraging unlabeled data for improving ASR model accuracy. In the self-training paradigm for SSL, a seed model trained with limited amount of labeled data is used to generate transcriptions (pseudo labels) for unlabeled data (cf. [18]). This procedure can be iterated on additional unlabeled data [19]. Another possible implementation of SSL is via teacher-student training, where a 'student' model is trained to replicate the outputs of a powerful 'teacher' model on the unlabeled data [20].

The central research question of our paper is how to best integrate SSL with DA for training E2E ASR systems. So far, the use of DA with SSL for E2E ASR has been largely limited to a simple cascade of both, i.e., doing pseudo label generation for the unlabeled data and then applying DA [21–24]. In contrast, for image classification, several algorithms have recently been proposed that use DA for teacher-student training [25] and *consistency regularization* in SSL [26–28]. Consistency regularization stems from the intuition that a small perturbation to an input data sample should not change the output distribution a lot. However, these SSL algorithms were designed only for static classification and need to be modified to support the seq2seq ASR use case.

In this regard, our paper makes the following contributions: First, we modify the Noisy Student [25] and FixMatch [28] algorithms – which have only been applied to image classification so far – for the seq2seq ASR use case. Second, we show performance improvements for both algorithms by using soft labels and consistency training (via SpecAugment and dropout). Finally, we demonstrate additional gains from iterative generation of pseudo labels by exploiting a larger amount of unlabeled data. We show that our proposed methods outperform the simple approach of doing DA after generating pseudo labels.

Relation to prior work: Several studies have recently investigated SSL techniques for E2E ASR, e.g. representation learning [29], the usage of external text data [30], text-to-speech [31], and transcriptions generated by conventional ASR [32]. Regarding self-training for E2E ASR, [22] proposed data filtering and ensemble schemes, generating hard pseudo labels via beam search. In [33], dropout was employed to improve the pseudo label accuracy and confidence measure, due to its well-known model ensembling property. All of these works did not consider the interaction of SSL with DA, which we investigate in our study. Very recently, [34] proposed teacher-student learning with DA for consistency training, without considering the Noisy Student or the FixMatch algorithm or soft labels as in our work.

2. Methods

For our E2E ASR models, we use an encoder-decoder architecture with attention as described in [9], which is similar to Listen-Attend-Spell (LAS) [6]. The ASR task is treated as a seq2seq learning problem: The model M is trained to predict a sequence y_j of symbols (here, we use sub-word units) from a sequence of acoustic features (usually a spectrogram x).

The encoder e creates a hidden representation of the acoustic features. Here, e is implemented as a stack of convolutional (CNN) layers followed by bidirectional Long Short-Term Memory (bLSTM) layers. The decoder is similar to an RNN LM that takes into account a context vector c_j. Bahdanau attention [35] is used to focus c_j on various parts of the encoder output. The

http://dx.doi.org/10.21437/Interspeech.2020-1337

output distribution $p_j = p(y_j|y_1, \ldots, y_{j-1}, x) = M(x, y_{<j})$ for the j-th symbol is computed by

$$
\begin{aligned}
s_j &= f(s_{j-1}, \text{Embedding}(y_{j-1}), c_{j-1}), && (1)\\
c_j &= \text{Attention}(s_j, e(x)), && (2)\\
s'_j &= \text{Dense}(s_j, c_j), && (3)\\
p_j &= g(s'_j), && (4)
\end{aligned}
$$

where g is a softmax layer and f is a stack of LSTM layers.

To perform ASR using the seq2seq model, hypotheses are generated by beam search. The model is conditioned on its previous output y_{j-1} in Eq. (1). The sequence $y_1, \ldots, y_j, \ldots$ of output symbols is produced one at a time, until the end-of-sentence symbol is reached.

2.1. Supervised and Semi-Supervised Training

For *supervised* training, the following cross-entropy (CE) loss is minimized for a mini-batch $\mathcal{B}$ of training examples:

$$\mathcal{L}_\theta(\mathcal{B}) = \frac{1}{|\mathcal{B}|} \sum_{i \in \mathcal{B}, j} \mathcal{L}^{\text{CE}}(y^*_{i,j}, M_\theta(A(x_i), y^*_{i,<j})). \quad (5)$$

Here, $y^*_{i,j}$ is the j-th symbol in the ground truth (GT) transcription of the i-th utterance in $\mathcal{B}$, represented as a one-hot vector, $A(x_i)$ denotes the features of the i-th utterance in $\mathcal{B}$ after data augmentation (cf. Section 2.2), and θ are the model parameters.

In *semi-supervised* training, the loss (5) is split into a labeled and an unlabeled part as $\mathcal{L}_\theta(\mathcal{B}) = (1/|\mathcal{B}|)(\mathcal{L}_\theta(\mathcal{B}^l) + \mathcal{L}_\theta(\mathcal{B}^u))$,

$$\mathcal{L}_\theta(\mathcal{B}^l) = \sum_{i \in \mathcal{B}^l, j} \mathcal{L}^{\text{CE}}(y^*_{i,j}, M_\theta(A(x^l_i), y^*_{i,<j})), \quad (6)$$

$$\mathcal{L}_\theta(\mathcal{B}^u) = \sum_{i \in \mathcal{B}^u, j} \mathbb{1}(\hat{p}_{i,j} \geq C)\, \mathcal{L}^{\text{CE}}(\hat{y}_{i,j}, M_\theta(A(x^u_i), \hat{y}_{i,<j})), \quad (7)$$

where $\mathcal{B}^l$ contains labeled and $\mathcal{B}^u$ contains unlabeled utterances, $\mathcal{B} = \mathcal{B}^l \cup \mathcal{B}^u$. For the unlabeled loss (7), pseudo labels, i.e. pseudo truth (PT) transcriptions $\hat{y}_{i,j}$, need to be obtained (cf. Section 2.3). Denoting the pseudo label confidence for utterance i and token j as $\hat{p}_{i,j}$, confidence-based filtering is included in the loss (7) by setting a confidence threshold $C > 0$.

2.2. Data Augmentation (DA)

In our training framework, DA is modeled by a function A, which also depends on the current state of the random number generator. We parameterize the function A based on SpecAugment (SA) [17] with hyperparameters F_{max}, T_{max}, m_F and m_T. The SA algorithm masks (replaces by zeros) up to F_{max} contiguous frequency bands and up to T_{max} contiguous time frames in the spectrogram x. The starting positions and the actual number of masked rows / columns are sampled from a uniform distribution. This process is repeated m_F times for masking frequencies and m_T times for masking time frames. Due to the combinatorial explosion of possible corruptions applied by A to a single input x, the usage of SA results in a practically infinite amount of training data.

2.3. Pseudo Labeling

We obtain pseudo labels for SSL based on PT transcriptions of the unlabeled data. These are generated 'offline' prior to training using beam search with a seq2seq model. Data augmentation is turned off during this offline PT generation phase. Since the PT generation is done only once, it is reasonable to use a large beam size W to improve the PT quality (here, we set $W = 16$). Furthermore, we apply a heuristic loop filtering technique to the PT utterances similar to the one from [22]. This helps us avoid reinforcing the well-known 'looping' problem, where the seq2seq model keeps repeating the same n-gram.

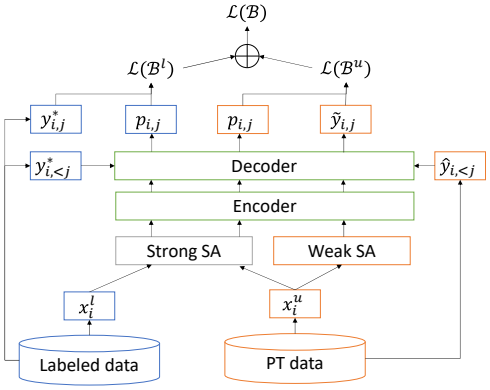

Figure 1: *Sequence-to-sequence FixMatch algorithm.*

Semi-supervised training can be performed by directly using the PT transcriptions as pseudo labels $\hat{y}_{i,j}$ in (7). Alternatively, we can dynamically update the PT transcriptions in the training process as in the FixMatch or Noisy Student approaches.

2.3.1. DA for Consistency Training (FixMatch)

The FixMatch algorithm is a self-training method that was proposed in [28] for image classification. The method implements *consistency training* by applying two kinds of data augmentation to the input x in both the unlabeled loss and the pseudo label generation, thus encouraging the outputs to be consistent for both augmented inputs. Here, we present an extension of FixMatch to seq2seq models. Specifically, we generate pseudo labels $\tilde{y}_{i,j}$ for unlabeled training examples x^u_i as:

$$\tilde{y}_{i,j} = M_\theta(A_w(x^u_i), \hat{y}_{i,<j}), \quad (8)$$

where A_w is a *weak* augmentation function. We then use $\tilde{y}_{i,j}$ in place of the PT label $\hat{y}_{i,j}$ in Eq. (7). Figure 1 depicts our FixMatch algorithm for seq2seq ASR.

As argued by [28], choosing a weak augmentation A_w for pseudo label generation instead of the 'strong' augmentation A in the unlabeled loss ensures reasonable accuracy of the pseudo labels and improves the convergence. Moreover, our experiments show that applying the confidence threshold C leads to gradual annealing of the supervised training signal, as the model becomes more and more confident and hence includes more and more unlabeled data as the training proceeds. Finally, by applying dropout in all forward passes of the model (including the pseudo label generation), we add further regularization to the training.

Besides extending the algorithm to seq2seq, we make two further modifications. First, we also investigate using soft labels, in order to make the training more robust against pseudo label errors in case of ambiguous instances. Second, while [28] applied weak augmentation on labeled data, we found strong augmentation to perform better in practice.

2.3.2. Teacher-Student Training with DA (Noisy Student)

The Noisy Student algorithm [25] is another recent SSL algorithm proposed for image classification, combining teacher-

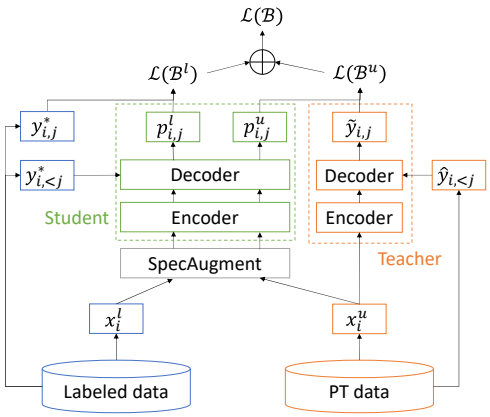

Figure 2: *Sequence-to-sequence Noisy Student algorithm.*

student training with DA. The main difference to FixMatch is that pseudo labels are generated by a pretrained teacher model M_{θ^*} with 'frozen' parameters θ^*, instead of the current model parameters θ:

$$\tilde{y}_{i,j} = M_{\theta^*}(x^u_i, \hat{y}_{i,<j}). \tag{9}$$

In our experiments, the teacher model is of the same topology as the student model. Figure 2 depicts our Noisy Student algorithm for ASR.

Since both hard and soft teacher labels (distillation) were considered in [25], we also explored both for the E2E ASR task. The hard label variant is equivalent to using the PT transcriptions in the unlabeled loss (7), i.e. $\tilde{y}_{i,j} = \hat{y}_{i,j}$. For the soft label variant, we (re-)compute soft labels on-the-fly via a forward pass on the teacher model. This is done to avoid storing full posteriors, which is hard for large label spaces and PT data sets. The on-the-fly label computation also allows us to include dynamic modifications of the input and the model. Specifically, we explore consistency training for the Noisy Student algorithm by (weakly) augmenting the input for pseudo label generation as in FixMatch (Eq. 8), or running the teacher model with dropout included. Our Noisy Student framework also subsumes generating hard labels with consistency training, similar to [34].

2.3.3. Iterative SSL

We also investigate iterative SSL, i.e., several rounds of PT generation by the E2E model via beam search, in two ways. First, we perform several iterations of the Noisy Student algorithm similar to [25]. Second, we also compare to an iterative self-training algorithm with DA inspired by [24], that regenerates PT transcription for each mini-batch based on the current model parameters. However, while [24] used greedy search, we found this to yield insufficient PT quality and thus adopted a lightweight beam search ($W = 4$).

2.4. Decoding

The decoding loss is extended with several terms so as to avoid the bias of seq2seq models to both deletions and insertions. Specifically, we maximize the following score via histogram pruning target synchronous beam search:

$$\log p(y\,|\,x) + \lambda_{\text{cov}} \operatorname{cov}(x, y) + \lambda_{\text{wip}}|y| + \left(\frac{5 + |u|}{5 + 1}\right)^{\lambda_{\text{rlp}}} \tag{10}$$

where $p(y|x)$ is the model probability; $\operatorname{cov}(x, y)$ is a coverage score (see Eq. (11) of [36]) weighted by λ_{cov} which adds a bonus

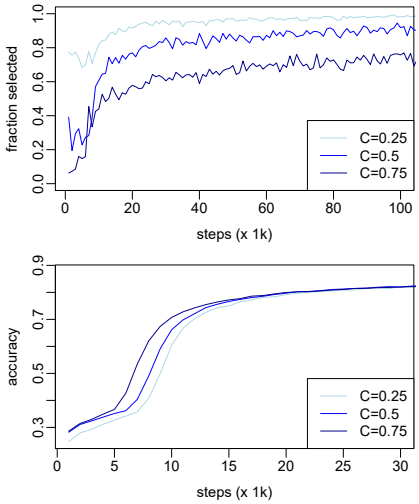

Figure 3: *FixMatch behavior over time for various confidence thresholds C.*

per each encoder states that is attended more than 0.5 while decoding; λ_{wip} is a constant word insertion penalty to balance the average probability reduction per decoded token; and the last term is a root length bonus/penalty to better approximate the length bias [37].

3. Experiments

3.1. Data Set

Our methods are evaluated on a conversational speech transcription task (doctor-patient conversations). All speech data are anonymized field data. Our experiments are based on a corpus of 1.9 k hours manually end-pointed and transcribed speech. We divided the corpus into four parts of equal size, treating only the first one (475 h) as labeled. In a first set of experiments, we treated the second part (475 h) as unlabeled data (ignoring the labels). We performed additional experiments adding the remaining 950 h of data as unlabeled data. In all cases, we measure the word error rate (WER) on a test set of 300 k words (26.8 h).

3.2. E2E Model Configuration

Our E2E models use 80-dimensional log Mel filterbank outputs as input features. The inputs are first passed through a stack of 3 CNN layers, which is parameterized so as to yield a 512-dimensional embedding for each input frame. Subsequently, there is a pyramid bLSTM encoder with 6 layers (512 LSTM units per layer and direction), which performs frame decimation (by a factor of 2) after every other layer, thus reducing the frame rate by a factor of 8. The decoder uses 2 (unidirectional) LSTM layers (1 024 units per layer). The softmax output layer predicts the posterior probabilities for 2 k word piece targets determined on the training set. In total, the model has 66 m parameters.

3.3. Training Parameters

The E2E models are trained with dropout [38] (with probability 0.3), label smoothing for hard labels [39] (with probability 0.9 for the target class), and early stopping (using a validation set held out from the training data) in order to improve generalization. The SpecAugment parameters are set as $F_{\text{max}} = 35$, $T_{\text{max}} = 50$, $m_F = 1$, $m_T = 2$. For the weak augmentation A_w in pseudo label generation, SA is parameterized with $F_{\text{max}} = 5$, $m_F = 1$, $T_{\text{max}} = m_T = 0$ (frequency masking only).

Table 1: *FixMatch results using 475 h labeled data (GT), 475 h unlabeled data (PT) and 200 k training steps.*

C	PT labels	PT noise	Init	WER
Supervised (475h)				
–	–	–	random	16.77
Semi-supervised, FixMatch				
0.5	hard	dropout	random	15.97
0.5	hard	+ SA ($F = 5$)	random	16.13
0.5	hard	+ strong SA	random	16.34
0.5	soft	dropout	random	15.79
0.0	soft	dropout	475 h GT	**15.22**

4. Results

4.1. Supervised Training

First, we performed baseline experiments with supervised training and SA. Using 475 h GT data, we obtain 16.77 % WER (cf. Table 1), which is the baseline for our SSL experiments. Using 1.9 kh GT data, we achieve 13.79 % WER, which is the lower bound on the WER attainable with SSL.

4.2. FixMatch

Figure 3 shows the behavior of the FixMatch algorithm for different confidence thresholds C. It can be seen that the fraction of selected tokens decreases significantly with increasing C, thus effectively training on less data. However, as training progresses, the validation accuracy varies little between different C, which also results in similar WER. Table 1 shows the ASR performance for various FixMatch settings, using a short training schedule of 200 k maximum steps to reduce experimental turnaround time. We observe that adding SA-based consistency training on top of dropout does not give an improvement; however, it is confirmed that using strong augmentation instead of weak augmentation in pseudo label computation degrades the results. The usage of soft labels leads to a slight gain (1.1 % WERR, where WERR is defined as relative WER reduction).

To further improve on the FixMatch performance, we initialized the FixMatch parameters from the model previously trained on 475 h GT data, while also dispensing with the confidence-based selection, since in this case we assume that the PT labels are of high quality from the start. This approach led to a significant improvement (3.6 % WERR), yielding 9.2 % WERR from FixMatch compared to the supervised baseline.

4.3. Noisy Student

Results of the Noisy Student algorithm are shown in Table 2, using a training schedule of 70 epochs (with early stopping). We observe 2.2 % WERR from using consistency regularization compared to the hard label Noisy Student algorithm. Moreover, the usage of soft labels improves the results considerably (2.8 % WERR). The best result (15.02 % WER) is obtained by combining soft labels and consistency training. The best Noisy Student setup performs similar to the best FixMatch setup, and outperforms the iterative self-training similar to [24] by 2 %.

We also assess the WER recovery rate (WRR) of our SSL algorithms as defined in [22], which is the ratio of performance gain achieved by adding unlabeled data vs. the gain from adding the same amount of *labeled* data (in this case, training with 950 h labeled data). The simplest variant, which uses one-shot hard label PT generation (15.60 % WER), achieves 61.6 % WRR. The other SSL methods in Table 2, which all use on-the-fly PT label generation, can improve on this baseline. In particular, our variant using soft labels along with consistency training obtains 92.1 % WRR.

Table 2: *Comparison of FixMatch and Noisy Student methods, using 475 h of labeled (GT) and 475 h of unlabeled data (PT), and 70 training epochs. SA: SpecAugment. Init(ialization): random or from the supervised training on 475 h GT.*

PT labels	PT noise	Init	WER	WRR
Semi-supervised, Noisy Student				
hard	none	random	15.60	61.6
hard	SA ($F = 5$)	random	15.26	79.5
soft	none	random	15.16	84.7
soft	dropout	random	15.10	87.9
soft	SA ($F = 5$)	random	**15.02**	**92.1**
Semi-supervised, FixMatch				
soft	dropout	475 h GT	15.04	91.1
Semi-supervised, iterative self-training [24]				
hard	none	475 h GT	15.34	75.3
Supervised (950 h), oracle performance				
–	–	random	*14.87*	*100.0*

Table 3: *Results with iterative PT generation using a total of 475 h labeled and 1.4 kh unlabeled data.*

PT labels	PT noise	Init	WER	WRR
Semi-supervised, Noisy Student				
hard	none	random	14.88	63.4
soft	none	random	14.52	75.5
soft	SA ($F = 5$)	random	**14.44**	**78.2**
Supervised (1.9 kh), oracle performance				
–	–	random	*13.79*	*100.0*

While we found only minor performance gains from multiple iterations of the Noisy Student algorithm on the *same* unlabeled data, we obtained additional improvements when including additional 950 h PT data, for a total of 1.4 kh PT data. In this case, the PT labels $\hat{y}_{i,j}$ were generated using a model trained on 475 h GT and 475 h PT using the Noisy Student algorithm. Results are shown in Table 3. With soft PT labels and consistency training, we achieve a 4 % improvement over the previous best result.

5. Conclusions

In this paper, we presented a comprehensive study on SSL strategies for end-to-end ASR. We investigated the FixMatch and Noisy Student algorithms for ASR and demonstrated improvements from using soft labels and consistency training. We believe that this is due to their ability to reduce error reinforcement in SSL. In the result, the performance of SSL can approach the one of supervised learning with similar amounts of data.

In future work, we will extend our investigation to different DA schemes beyond SpecAugment (e.g. [15], [16]). We will also look at iteratively including more and more unlabeled data while increasing the model size as in [25].

6. References

[1] A. Graves, "Sequence transduction with recurrent neural networks," in *Proc. of ICML Workshop on Representation Learning*. Edinburgh, UK: IEEE, 2012, pp. 1–9.

[2] D. Bahdanau, J. Chorowski, D. Serdyuk, P. Brakel, and Y. Bengio, "End-to-end attention-based large vocabulary speech recognition," in *Proc. of ICASSP*. Shanghai, China: IEEE, 2016, pp. 4945–4949.

[3] E. Battenberg, J. Chen, R. Child, A. Coates, Y. Gaur, Y. Li, H. Liu, S. Satheesh, A. Sriram, and Z. Zhu, "Exploring neural transducers for end-to-end speech recognition," in *Proc. of ASRU*. Okinawa, Japan: IEEE, 2017, pp. 206–213.

[4] R. Prabhavalkar, K. Rao, T. N. Sainath, B. Li, L. Johnson, and N. Jaitly, "A comparison of sequence-to-sequence models for speech recognition," in *Proc. of INTERSPEECH*. Stockholm, Sweden: ISCA, 2017, pp. 939–943.

[5] J. Li, G. Ye, A. Das, R. Zhao, and Y. Gong, "Advancing acoustic-to-word CTC model," in *Proc. of ICASSP*. Calgary, Canada: IEEE, 2018, pp. 5794–5798.

[6] W. Chan, N. Jaitly, Q. Le, and O. Vinyals, "Listen, attend and spell: A neural network for large vocabulary conversational speech recognition," in *Proc. of ICASSP*. Shanghai, China: IEEE, 2016, pp. 4960–4964.

[7] C.-C. Chiu, T. N. Sainath, Y. Wu, R. Prabhavalkar, P. Nguyen, Z. Chen, A. Kannan, R. J. Weiss, K. Rao, E. Gonina *et al.*, "State-of-the-art speech recognition with sequence-to-sequence models," in *Proc. of ICASSP*. Calgary, Canada: IEEE, 2018, pp. 4774–4778.

[8] A. Zeyer, K. Irie, R. Schlüter, and H. Ney, "Improved training of end-to-end attention models for speech recognition," in *Proc. of INTERSPEECH*. Hyderabad, India: ISCA, 2018, pp. 7–11.

[9] F. Weninger, J. Andrés-Ferrer, X. Li, and P. Zhan, "Listen, Attend, Spell and Adapt: Speaker adapted sequence-to-sequence ASR," in *Proc. of INTERSPEECH*. ISCA, 2019, pp. 3805–3809.

[10] Z. Tüske, G. Saon, K. Audhkhasi, and B. Kingsbury, "Single headed attention based sequence-to-sequence model for state-of-the-art results on Switchboard-300," *arXiv:2001.07263*, 2020.

[11] A. Hernández-García and P. König, "Data augmentation instead of explicit regularization," *arXiv:1806.03852*, 2018.

[12] T. Ko, V. Peddinti, D. Povey, and S. Khudanpur, "Audio augmentation for speech recognition," in *Proc. of INTERSPEECH*. Dresden, Germany: ISCA, 2015, pp. 3586–3589.

[13] X. Cui, V. Goel, and B. Kingsbury, "Data augmentation for deep neural network acoustic modeling," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 23, no. 9, pp. 1469–1477, 2015.

[14] W. Zhou, W. Michel, K. Irie, M. Kitza, R. Schlüter, and H. Ney, "The RWTH ASR system for TED-LIUM Release 2: Improving hybrid HMM with SpecAugment," in *Proc. of ICASSP*. Barcelona, Spain: IEEE, 2020, pp. 7839–7843.

[15] G. Saon, Z. Tüske, K. Audhkhasi, and B. Kingsbury, "Sequence noise injected training for end-to-end speech recognition," in *Proc. of ICASSP*. Brighton, UK: IEEE, 2019, pp. 6261–6265.

[16] T.-S. Nguyen, S. Stüker, J. Niehues, and A. Waibel, "Improving sequence-to-sequence speech recognition training with on-the-fly data augmentation," in *Proc. of ICASSP*. Barcelona, Spain: IEEE, 2020, pp. 7689–7693.

[17] 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," *Proc. of INTERSPEECH*, pp. 2613–2617, 2019.

[18] L. Lamel, J.-L. Gauvain, and G. Adda, "Lightly supervised and unsupervised acoustic model training," *Computer Speech & Language*, vol. 16, no. 1, pp. 115–129, 2002.

[19] B. Khonglah, S. Madikeri, S. Dey, H. Bourlard, P. Motlicek, and J. Billa, "Incremental semi-supervised learning for multi-genre speech recognition," in *Proc. of ICASSP*. Barcelona, Spain: IEEE, 2020, pp. 7419–7423.

[20] M. Gibson, G. Cook, and P. Zhan, "Semi-supervised training strategies for deep neural networks," in *Proc. of ASRU*. Okinawa, Japan: IEEE, 2017, pp. 77–83.

[21] G. Synnaeve, Q. Xu, J. Kahn, E. Grave, T. Likhomanenko, V. Pratap, A. Sriram, V. Liptchinsky, and R. Collobert, "End-to-end ASR: From supervised to semi-supervised learning with modern architectures," *arXiv:1911.08460*, 2019.

[22] J. Kahn, A. Lee, and A. Hannun, "Self-training for end-to-end speech recognition," in *Proc. of ICASSP*. Barcelona, Spain: IEEE, 2020, 7084–7088.

[23] Y. Huang, S. Thomas, M. Suzuki, Z. Tüske, L. Sansone, and M. Picheny, "Semi-supervised training and data augmentation for adaptation of automatic broadcast news captioning systems," in *Proc. of ASRU*. Sentosa, Singapore: IEEE, 2019, pp. 867–874.

[24] Y. Chen, W. Wang, and C. Wang, "Semi-supervised ASR by end-to-end self-training," *arXiv:2001.09128*, 2020.

[25] Q. Xie, E. Hovy, M.-T. Luong, and Q. V. Le, "Self-training with noisy student improves ImageNet classification," in *Proc. of CVPR*. Seattle, WA: IEEE, 2020, to appear.

[26] Q. Xie, Z. Dai, E. Hovy, M.-T. Luong, and Q. V. Le, "Unsupervised data augmentation for consistency training," *arXiv:1904.12848*, 2019.

[27] D. Berthelot, N. Carlini, I. Goodfellow, N. Papernot, A. Oliver, and C. A. Raffel, "MixMatch: A holistic approach to semi-supervised learning," in *Proc. of Advances in Neural Information Processing Systems*. Vancouver, Canada: Curran Associates, Inc., 2019, pp. 5050–5060.

[28] K. Sohn, D. Berthelot, C.-L. Li, Z. Zhang, N. Carlini, E. D. Cubuk, A. Kurakin, H. Zhang, and C. Raffel, "FixMatch: Simplifying semi-supervised learning with consistency and confidence," *arXiv:2001.07685*, 2020.

[29] S. Ling, Y. Liu, J. Salazar, and K. Kirchhoff, "Deep contextualized acoustic representations for semi-supervised speech recognition," in *Proc. of ICASSP*. Barcelona, Spain: IEEE, 2020, 6429–6433.

[30] S. Karita, S. Watanabe, T. Iwata, A. Ogawa, and M. Delcroix, "Semi-supervised end-to-end speech recognition," in *Proc. of Interspeech*. Hyderabad, India: ISCA, 2018, pp. 2–6.

[31] S. Karita, S. Watanabe, T. Iwata, M. Delcroix, A. Ogawa, and T. Nakatani, "Semi-supervised end-to-end speech recognition using text-to-speech and autoencoders," in *Proc. of ICASSP*. Brighton, UK: IEEE, 2019, pp. 6166–6170.

[32] B. Li, T. N. Sainath, R. Pang, and Z. Wu, "Semi-supervised training for end-to-end models via weak distillation," in *Proc. of ICASSP*. Brighton, UK: IEEE, 2019, pp. 2837–2841.

[33] S. Dey, P. Motlicek, T. Bui, and F. Dernoncourt, "Exploiting semi-supervised training through a dropout regularization in end-to-end speech recognition," in *Proc. of INTERSPEECH*. Graz, Austria: ISCA, 2019, pp. 734–738.

[34] R. Masumura, M. Ihori, A. Takashima, T. Moriya, A. Ando, and Y. Shinohara, "Sequence-level consistency training for semi-supervised end-to-end automatic speech recognition," in *Proc. of ICASSP*. Barcelona, Spain: IEEE, 2020, pp. 7054–7058.

[35] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," in *Proc. of ICLR*. San Diego, CA: open publishing, 2015.

[36] J. Chorowski and N. Jaitly, "Towards better decoding and language model integration in sequence to sequence models," in *Proc. of INTERSPEECH*. Stockholm, Sweden: ISCA, 2017, pp. 523–527.

[37] Y. Wu, M. Schuster, Z. Chen, Q. V. Le, M. Norouzi, W. Macherey, M. Krikun, Y. Cao, Q. Gao, K. Macherey *et al.*, "Google's neural machine translation system: Bridging the gap between human and machine translation," *arXiv:1609.08144*, 2016.

[38] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[39] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proc. of CVPR*. Las Vegas, NV: IEEE, 2016, pp. 2818–2826.