EXPLICIT ALIGNMENT OF TEXT AND SPEECH ENCODINGS FOR ATTENTION-BASED END-TO-END SPEECH RECOGNITION

Jennifer Drexler, James Glass

MIT Computer Science and Artificial Intelligence Laboratory
Cambridge, MA, USA
{jdrexler, glass}@csail.mit.edu

ABSTRACT

In this work, we present a novel training procedure for attention-based end-to-end automatic speech recognition. Our goal is to push the encoder network to output only linguistic information, improving generalization performance particularly in low-resource scenarios. We accomplish this with the addition of a text encoder network, which the speech encoder is encouraged to mimic. Our main innovation is the comparison of the *attention-weighted* speech encoder outputs to the outputs of the text encoder - this guarantees two sequences of the same length that can be directly aligned. We show that our training procedure significantly decreases word error rates in all experiments and has the biggest absolute impact in the lowest resource scenarios.

Index Terms— speech recognition, end-to-end, attention, low-resource

1. INTRODUCTION

End-to-end neural network models for automatic speech recognition (ASR) have recently achieved comparable performance to traditional HMM-based models in tasks with large training corpora [1, 2]. However, these models can easily overfit when trained on smaller corpora, making them ill-suited to lower-resource tasks [3, 4].

This is unfortunate, given that low-resource scenarios are exactly where the advantages of end-to-end models are most necessary. For example, end-to-end models have been found to work best with graphemic output, while traditional models work best with access to a pronunciation dictionary [5]. In practice, domains or languages in which only small training corpora are available tend to also be ones in which such dictionaries, as well as the resources to do feature engineering or follow a complicated training procedure, are also limited. If they can be modified to work well with less training data, end-to-end models could enable effective ASR without significant domain expertise.

In this paper, we consider an additional objective function designed to improve the generalization performance of end-to-end neural network models in low-resource scenarios. We argue that the ability to easily incorporate multiple objective functions to train specific model components is a key advantage of neural network models that has been overlooked in the ASR literature, especially in low-resource scenarios.

Our novel objective function makes use of a text encoder network in combination with an attention-based end-to-end ASR system. In addition to end-to-end training of the ASR model, our objective function pushes the ASR encoder to match the output of the text encoder when the two networks are fed paired inputs. In this way, we hope to force the speech encoder to output only linguistic information and not any other information (like speaker and noise characteristics) contained in the speech signal. Our hypothesis is that this will improve the generalization performance of the end-to-end model by limiting spurious correlations between the output transcripts and non-linguistic information in the speech signal, which are most likely to occur in low-resource scenarios.

One key contribution of our work is to handle the length discrepancies between the speech and text encoder outputs by comparing the *attention-weighted* speech encoder outputs to the text encoder outputs. Once trained, the attention mechanism provides an explicit alignment between speech and text that enables this comparison.

Our model training proceeds in several stages. First, we train a baseline ASR model. Then, we train the text encoder to match the ASR encoder as well as it can - by definition, this will capture the linguistic information in the speech encoder outputs but nothing else. Next, we retrain the speech encoder to match the text encoder - removing the non-linguistic information from the encoder outputs. Finally, we retrain the decoder to effectively decode these updated encoder outputs.

We use the Librispeech [6] corpus for experiments, training on the "clean 100" set or a smaller subset - 20 or 50 hours - of it. We show that, by following the procedure outlined above, we can significantly improve performance over the baseline and that the improvements are larger for models trained on less data. We confirm the effectiveness of this procedure for low-resource scenarios with the SI84 training set from the Wall Street Journal corpus [7]. We also explore the

contribution of the different steps in our training procedure with several "early stopping" experiments in which certain stages of training are not trained to convergence.

To conclude, we discuss several paths forward for this work, both towards achieving these gains with a simpler training procedure and towards making use of additional text corpora and untranscribed speech within the framework presented here.

2. RELATED WORK

In this paper, we focus on attention-based end-to-end neural network models for ASR, specifically the listen, attend, and spell (LAS) architecture [8]. Similar models have recently achieved state-of-the-art status in a number of large ASR tasks [1, 2, 9]. These impressive results rely on several modifications to the LAS architecture - including pretraining, subword units, and multi-headed attention - some of which we incorporate here.

In low-resource settings, end-to-end ASR results are more mixed. In [3], a traditional HMM-DNN system outperformed both CTC-based [10] and attention-based [8, 11] end-to-end models on eight low-resource languages, each with 40 hours of training data.

End-to-end ASR has been successful, however, in low-resource settings for which additional unpaired speech and text are available for semi-supervised training. [4] introduced a multi-task learning framework similar to the model in this work, but used adversarial training to create a shared embedding space from unpaired speech and text. In [12] and [13], the authors develop a "speech chain" model created by combining ASR and text-to-speech (TTS) models. They use the ASR model to transcribe additional speech, creating synthetic training data for the TTS system, and similarly use the TTS model to create ASR training data from additional text. This process iteratively improves both ASR and TTS performance.

Hori et al. [14] introduced a modified, and somewhat simplified, version of the speech-chain system by replacing the TTS system with a text-to-encoder (TTE) model, with essentially the same structure as the TTS model used in [12]. Their training procedure had three stages: first, standard end-to-end training of the ASR model; second, training the TTE model; and third, unsupervised cycle-consistency training. The TTE model is trained in the second stage with paired speech and text: it is trained to take input text and produce the same output that the speech encoder produces when given the matching speech utterance. In the third stage, the authors sample several possible text outputs from the ASR model for a given speech input and compute a cycle-consistency loss based on the difference between the output of the speech encoder and the TTE model outputs.

Our model and training procedure share many similarities with [14], but we consider only fully supervised training - we are able to significantly improve our baseline results using only the same data used to train the baseline model. Our text encoder is much simpler than the one used in [14] but still has the potential for semi-supervised training using a similar cycle-consistency loss; we plan to explore this in future work.

Within the traditional ASR framework, there is a thread of research that also bears some similarity to this work: adversarial training for removing non-linguistic factors from speech representations. This research has been focused on neural network acoustic models. [15] uses adversarial training to push the outputs of an intermediate layer of a DNN acoustic model to be speaker-invariant; [16] uses a similar method to induce features that are invariant to noise conditions. In both cases, the authors reported significant improvements in performance. Here, we attempt to generate similar invariance without using explicit speaker or noise condition labels.

3. ASR MODEL ARCHITECTURE

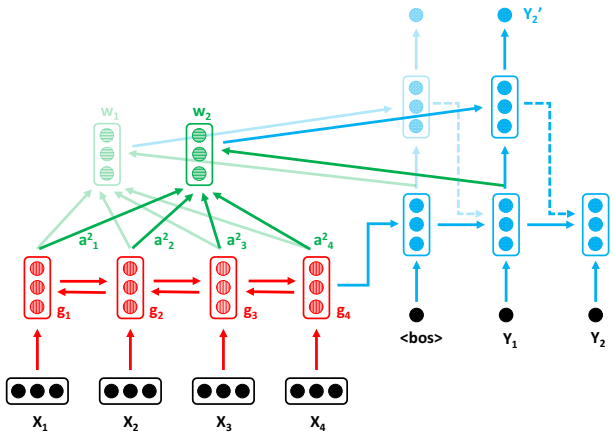

Fig. 1. High-level schematic of the ASR model architecture when using teacher forcing, at time-step $t = 2$. Speech inputs (X) and text labels (Y) shown in black/solid circles. Encoder outputs (g) shown in red/vertical-striped circles. Decoder states and outputs shown in blue/dotted circles. Attention-weighted encoder outputs (w) shown in green/horizontal-striped circles, with attention weights (a) computed as part of the transformation from g to w. Calculations from $t = 1$ rendered in faded colors.

Our ASR model is based on the listen, attend, and spell (LAS) architecture [8]. It is an encoder-decoder model with attention [11]. The architecture is depicted in Figure 1, which shows the inputs to the model, as well as the encoder outputs (in red), attention-weighted encoder-outputs (green), decoder intermediate states and outputs (blue). The neural network components themselves are not shown in the figure; they are encapsulated in the arrows.

The encoder is composed of four bidirectional LSTM layers, three of which downsample their input by a factor of two. The encoder as a whole downsamples the speech input by a factor of eight. This downsampling is left out of Figure 1 for simplicity.

The encoder is combined with an LSTM-based decoder with attention. The decoder itself has two unidirectional LSTM layers and a softmax layer. At each time-step, its input is the previous output from the decoder, starting with a special start symbol as the first input. During ASR training, we use a fixed scheduled sampling rate of 0.1: 10% of the time we randomly sample from the output distribution the decoder to provide the next input, the rest of the time we use the ground truth. The output targets of the decoder are either characters or subword units discovered with a unigram language model, as in [17].

The attention mechanism within the decoder uses input feeding [18], meaning that it compares the encoder outputs to both the current decoder state and the previous state of the attention mechanism. It is an MLP attention mechanism, as described in [11].

The ASR model is trained end-to-end with cross-entropy loss to maximize the log-likelihood of the ground truth output sequence. Our stopping criterion for training is the accuracy of the most likely decoder output on the validation set using teacher forcing. At test time, we use beam search [19] to find the most likely transcript.

4. ALIGNMENT OF TEXT AND SPEECH ENCODINGS

We introduce a text encoder network for our additional objective function. The text encoder architecture is quite simple: it has an embedding layer followed by two bidirectional LSTM layers that do not downsample their input. We use the same subword segmentation for input text as for the ASR output targets.

Our goal is to push the speech and text encoders to output similar encodings when fed matched speech/text pairs. This goal is complicated by the fact that the text and speech inputs, and thus the encoder outputs, are different lengths. A key innovation in this paper is to use the *attention-weighted* speech encoder outputs (w) for this training. For a matched speech/text pair, these will necessarily be the same length as the text encoder outputs when we use teacher forcing in the decoder, because the length of the attention-weighted speech encoder outputs is equal to the number of decoder steps.

We compute the encoding loss, L^{enc}, of the model given speech features X and text Y. $g(X)$ is the output sequence from the speech encoder given input X; it has length N, and $g_n(X)$ denotes the n^{th} element of $g(X)$. $h(Y)$ is the output sequence of the text encoder given input Y; its length is T. We use the MLP attention mechanism described in detail in [11]; its output, $a(X, Y)$, is an N-dimensional score vector at each decoding timestep t, where $\sum_{n=1}^{N} a_n^t(X, Y) = 1$. We compute the attention-weighted speech encoder output, w_t, for timestep t as:

$$w_t(X, Y) = \sum_{n=1}^{N} a_n^t(X, Y) * g_n(X) \quad (1)$$

The variables in Equation 1 (X, g, a, and w) are noted in Figure 1. We use the following equation to compute the encoder loss for the full utterance (X, Y):

$$L^{enc}(X, Y) = \sum_{t=1}^{T} SmoothL1(w_t(X, Y), h_t(Y)) \quad (2)$$

Equation 2 calculates the difference between the attention-weighted speech encodings, w, and the text encodings, $h(Y)$. We use an element-wise smoothed L1 loss that is equivalent to the mean squared-error (MSE) when the absolute difference is less than one and the L1 loss otherwise.

$$SmoothL1(a, b) = \begin{cases} 0.5(a-b)^2, & \text{if } |a-b| < 1 \\ |a-b| - 0.5, & \text{otherwise} \end{cases}$$

This is in contrast to [14], which uses the sum of the L1 loss and the MSE when comparing the speech and text encoder outputs.

5. TRAINING PROCEDURE

For all experiments in this paper, we follow a sequential training procedure with different loss functions depending on the step. The steps of this procedure are outlined in Table 1. Step 1 is baseline ASR training; this step impacts all of the parameters in the ASR model. In Step 2, we train the text encoder to match its outputs to those of the speech encoder; all parameters are fixed except for those of the text encoder. Step 3 uses the same loss as Step 2, but it is now used to train the parameters of the speech encoder, pushing the speech encodings closer to the text encoder outputs. Finally, in Step 4, we again do standard ASR training, but we fix the parameters of the speech encoder and train only the decoder.

For our main results, we train each step to convergence on an ASR validation set before moving on to the next step. Steps 1 and 4 are trained to maximize the accuracy of the validation set using the ASR model. Steps 2 and 3 are trained to minimize the loss between the speech and text encodings of the validation set.

We also present results for a set of experiments in which we explore the impact of stopping one of the training steps early, while still training the rest to convergence. These experiments are designed with two questions in mind. First, whether it is possible to achieve similar performance improvements with fewer overall training iterations, and second, what the requirements are for an effective shared encoding space for text and speech.

Table 1. Training Procedure

Step	Description	Loss Function	Components Trained
1	ASR Training	L^{asr}	speech encoder, attention, decoder
2	Text Encoder Training	L^{enc}	text encoder
3	Speech Encoder Training	L^{enc}	speech encoder
4	ASR Training	L^{asr}	attention, decoder

We considered an alternate training procedure in which we replace steps 1 and 2 with simply training a text autoencoder, followed by step 3, in which we train the parameters of the speech encoder to minimize the loss between the speech and text encoder outputs. We found, however, that this was not a reliable method for training the attention mechanism in the decoder.

6. EXPERIMENTAL DETAILS

In this paper, we experiment with two standard corpora: Librispeech [6] and Wall Street Journal (WSJ) [7]. For Librispeech, we present results for the standard clean_460 training set as a 'topline' and treat the standard clean_100 training set as a low-resource training regime. We also create two lower-resource scenarios with subsets of the clean_100 set: clean_50 and clean_20 which comprise 50% and 20% of the utterances in the clean_100 set, respectively. We use the standard clean validation and test sets for all Librispeech experiments. For WSJ, we use the SI84 training set (also called WSJ0), which contains approximately 15 hours of transcribed speech. We use the standard dev93 set for validation and all WSJ scores are reported on the eval92 set.

We use 80-dimensional log-mel filterbank features, computed using 25ms frames with a 10ms frame-rate for all speech input. Librispeech text was segmented using a unigram wordpiece model with a 500 unit vocabulary with a maximum unit length of four characters. WSJ models are character based.

All models trained with at least 50 hours of speech used 512 units for all encoder layers and 1024 units for all decoder layers. Models trained on the clean_20 set used 128 units for all encoder layers and 256 units for decoder layers - we find that a smaller model produces better results in very low-resource scenarios. For WSJ, we used 320 units in all layers, to match comparable prior work. All word embedding layers in all models had 128 units.

We use Kaldi[1] to generate all speech features. All models were trained with a modified version of OpenNMT[2]. We used SentencePiece[3] for text segmentation.

[1]https://github.com/kaldi-asr/kaldi

[2]https://github.com/OpenNMT/OpenNMT-py/

[3]https://github.com/google/sentencepiece

7. RESULTS AND DISCUSSION

7.1. Main Results

7.1.1. Librispeech

First, we establish a baseline for our low-resource conditions. These results are in the second and third columns of Table 2. We report both character error rate (CER) and word error rate (WER). For comparison, our topline model trained on the clean_460 training set achieves a WER of 12.7% and CER of 5.7%.

Table 2. Comparison between baseline models and models trained with encoding alignment procedure, across different amounts of training data.

	Baseline		Aligned	
Train Set	CER	WER	CER	WER
clean_100	11.2	23.3	9.9	21.3
clean_50	16.0	31.2	14.6	29.2
clean_20	30.4	53.3	27.7	49.7

Our baseline results on the standard 460 and 100 hour training sets are comparable to those reported in [14] (they report 11.1% CER, 25.2% WER on clean_100 and 4.6% CER, 11.8% WER on clean_460) - our ASR encoder has fewer layers with more units each and our decoder has a different output vocabulary, so it is to be expected that the results would be slightly different - in particular, our slightly less powerful model performs better on the smaller dataset but worse on the larger dataset.

Our results on the 50% and 20% subsets illustrate the general problem with low-resource ASR: performance degrades significantly as less and less training data is used.

The last two columns of Table 2 show the results of our encoding alignment procedure. In all cases, we are able to improve on the baseline results, despite using exactly the same training data as the comparable baseline model. The relative improvement in WER is largest on the clean_100 set (8.6%), while the absolute improvement is largest on the smallest training set (3.6%). It is also worth noting that our best result on the clean_100 training set is comparable to the model presented in [14] (CER: 9.4%, WER: 21.5%),

which uses an extra 360 hours of speech (without transcriptions). We are eager to explore in future work whether the gains achieved here are complementary with the use of an unsupervised cycle-consistency loss.

7.1.2. WSJ

Since the Librispeech clean_50 and clean_20 training sets were created for this paper, we have no comparison in the literature for those results. We ran the same experiments on the WSJ0 corpus, to confirm both the competitiveness of our baseline model and the effectiveness of our proposed method. Our baseline model achieves a CER of 15.0% and WER of 36.5%, compared to similar attention-based models reporting a CER of either 17.0% [20] or 15.8% [21]. The training procedure presented here reduces the CER to 13.8% and the WER to 35.0%. This represents a 4.1% relative improvement in WER.

7.2. Early Stopping Results

In this section, we experiment with early stopping of different steps of our training procedure. All experiments in this section use the clean_20 training set.

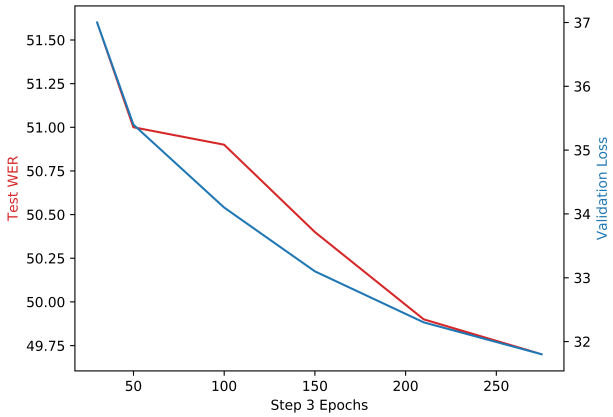

Fig. 2. Validation set loss (blue) and test set WER (red) as a function of the number of Step 3 training epochs completed.

7.2.1. Speech Encoder Training (Step 3)

Stopping Step 3 early is a test of our core hypothesis - we expect that it will negatively impact our results if the speech encoding space is not fully aligned to the text encoding space. To perform this test, we train Steps 1 and 2 to convergence, then save models at several points during Step 3. From each of these models, we train Step 4 to convergence to get a final WER.

As shown in Figure 2, we see a direct relationship between the encoding loss on the validation set and the WER on the test set as training progresses. This confirms our hypothesis that directly matching the speech encoder outputs to a text encoding space will improve generalization performance of the model.

7.2.2. Text Encoder Training (Step 2)

There is no theoretical reason why the text encoding space must be as close to the original speech encoding space as possible, given that we will be re-training the speech encoder to match whatever text encoder space is defined in Step 2. In these experiments, we stop Step 2 training early, then train Steps 3 and 4 to convergence using their respective stopping criteria. The results of these experiments are in Table 3, with models referenced based on how many epochs of Step 2 training were run before moving on to Step 3.

Table 3. Results of encoder alignment models with early stopping of Step 2. After Step 2, all models were trained to convergence on Steps 3 and 4.

Step 2 Epochs	Val L^{enc}	CER	WER
50	86.4	27.6	49.9
100	82.4	27.8	50.0
200	80.8	27.7	49.7

Table 3 shows that the final model performance does not depend strongly on the number of epochs of Step 2 training. The second column, indicating the encoder loss on the validation set at the end of Step 2 training, shows that this training has not converged after 50 or 100 epochs; still, the final results are comparable across all three models.

It is not necessary to train Step 2 to convergence, but that does mean that we will be able to save computation by stopping Step 2 early; it is possible that we are simply trading fewer iterations of Step 2 for more iterations of Step 3 in order to achieve the same final result. As shown in Figure 3, this is not the case: regardless of how many epochs of Step 2 we complete, Step 3 training follows virtually the same path in terms of the validation set loss and the models converge at approximately the same number of Step 3 epochs.

It is also interesting to note the difference in the values of the validation loss after Step 2 (in Table 3) and after Step 3 (in Figure 3). These numbers confirm that the original speech encoder outputs contain much non-linguistic information that the text encoder cannot learn to represent, and that much of this extraneous information is removed from the speech encodings during Step 3 training.

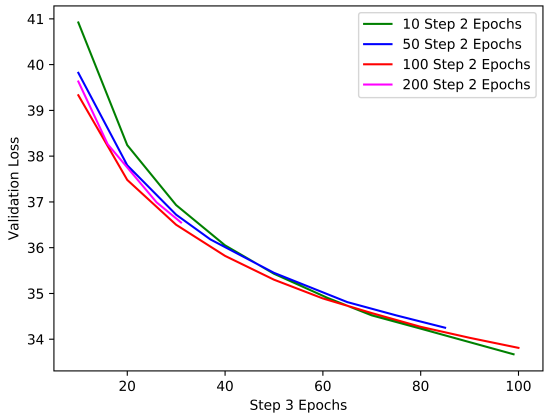

Fig. 3. Validation set loss (blue) as a function of the number of Step 3 training epochs completed. Each line represents a different early stopping point for Step 2. Validation loss at $X = 0$ is the loss at the end of Step 2.

8. CONCLUSIONS

In this paper, we develop a training procedure for attention-based ASR models designed to improve their generalization performance in low-resource settings. Our training strategy includes the addition of a text encoder network to a standard ASR model architecture, and a novel objective function designed to push the encoder component of the ASR model to represent only linguistic information. This is accomplished by encouraging the *attention-weighted* speech encoder outputs to match the outputs of the trained text encoder when the networks are fed paired speech/text inputs.

We experiment with several subsets - 100, 50, and 20 hours - of the Librispeech corpus and the SI84 set of the WSJ corpus, and find that our procedure improves WERs in all cases. On the 100 hour Librispeech set, we achieve comparable improvements to those reported in a related paper that made use of an additional 360 hours of untranscribed speech [14], while using only the same data used to train the baseline model.

Additionally, we show through a series of early-stopping experiments that the second step in our training process does not need to be trained to convergence, which can reduce the overall training time needed.

We look forward to several possible directions for future work using this framework. First, we plan to explore whether some steps of our training procedure can be trained jointly, reducing the complexity of model training. Second, we hope to investigate whether the improvements achieved here are complementary with the cycle-consistency loss proposed in [14]. In addition to the use of untranscribed speech, this model architecture can also take advantage of stand-alone text corpora, using the method described in [4]. Given the significant performance improvements demonstrated here in a fully supervised training scenario, we are confident we can also improve upon previously reported semi-supervised training results.

9. REFERENCES

[1] 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 *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 4774–4778.

[2] A. Zeyer, K. Irie, R. Schlüter, and H. Ney, "Improved training of end-to-end attention models for speech recognition," *arXiv preprint arXiv:1805.03294*, 2018.

[3] A. Rosenberg, K. Audhkhasi, A. Sethy, B. Ramabhadran, and M. Picheny, "End-to-end speech recognition and keyword search on low-resource languages," in *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2017, pp. 5280–5284.

[4] J. Drexler and J. Glass, "Combining end-to-end and adversarial training for low-resource speech recognition," in *Spoken Language Technology Workshop (SLT)*. IEEE, 2018, pp. 361–368.

[5] T. N. Sainath, R. Prabhavalkar, S. Kumar, S. Lee, A. Kannan, D. Rybach, V. Schogol, P. Nguyen, B. Li, Y. Wu *et al.*, "No need for a lexicon? evaluating the value of the pronunciation lexica in end-to-end models," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 5859–5863.

[6] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, "Librispeech: an asr corpus based on public domain audio books," in *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2015, pp. 5206–5210.

[7] LDC. (1994) Wall Street Journal Corpus. [Online]. Available: https://catalog.ldc.upenn.edu/ldc94s13a

[8] W. Chan, N. Jaitly, Q. Le, and O. Vinyals, "Listen, attend and spell: A neural network for large vocabulary conversational speech recognition," in *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2016, pp. 4960–4964.

[9] 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 *Interspeech*, 2017, pp. 939–943.

[10] A. Graves, S. Fernández, F. Gomez, and J. 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.

[11] D. Bahdanau, J. Chorowski, D. Serdyuk, P. Brakel, and Y. Bengio, "End-to-end attention-based large vocabulary speech recognition," in *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2016, pp. 4945–4949.

[12] A. Tjandra, S. Sakti, and S. Nakamura, "Listening while speaking: Speech chain by deep learning," in *Automatic Speech Recognition and Understanding Workshop (ASRU)*. IEEE, 2017, pp. 301–308.

[13] ——, "Machine speech chain with one-shot speaker adaptation," *arXiv preprint arXiv:1803.10525*, 2018.

[14] T. Hori, R. Astudillo, T. Hayashi, Y. Zhang, S. Watanabe, and J. L. Roux, "Cycle-consistency training for end-to-end speech recognition," *arXiv preprint arXiv:1811.01690*, 2018.

[15] Z. Meng, J. Li, Z. Chen, Y. Zhao, V. Mazalov, Y. Gang, and B.-H. Juang, "Speaker-invariant training via adversarial learning," in *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 5969–5973.

[16] D. Serdyuk, K. Audhkhasi, P. Brakel, B. Ramabhadran, S. Thomas, Y. Bengio, and C. F. MILA, "Invariant representations for noisy speech recognition," *arXiv preprint arXiv:1612.01928*, 2016.

[17] T. Kudo, "Subword regularization: Improving neural network translation models with multiple subword candidates," *arXiv preprint arXiv:1804.10959*, 2018.

[18] M.-T. Luong, H. Pham, and C. D. Manning, "Effective approaches to attention-based neural machine translation," *arXiv preprint arXiv:1508.04025*, 2015.

[19] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Advances in neural information processing systems*, 2014, pp. 3104–3112.

[20] S. Kim, T. Hori, and S. Watanabe, "Joint CTC-attention based end-to-end speech recognition using multi-task learning," in *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2017, pp. 4835–4839.

[21] S. Karita, S. Watanabe, T. Iwata, A. Ogawa, and M. Delcroix, "Semi-supervised end-to-end speech recognition." in *Interspeech*, 2018, pp. 2–6.