

# REGULARIZED FORWARD-BACKWARD DECODER FOR ATTENTION MODELS

*Tobias Watzel, Ludwig Kürzinger, Lujun Li, Gerhard Rigoll*

Chair of Human-Machine Communication, Technical University of Munich

## ABSTRACT

Nowadays, attention models are one of the popular candidates for speech recognition. So far, many studies mainly focus on the encoder structure or the attention module to enhance the performance of these models. However, mostly ignore the decoder. In this paper, we propose a novel regularization technique incorporating a second decoder during the training phase. This decoder is optimized on time-reversed target labels beforehand and supports the standard decoder during training by adding knowledge from future context. Since it is only added during training, we are not changing the basic structure of the network or adding complexity during decoding. We evaluate our approach on the smaller TEDLIUMv2 and the larger LibriSpeech dataset, achieving consistent improvements on both of them.

**Index Terms**— Speech recognition, Attention models, Forward-backward decoder, Regularization

## 1. INTRODUCTION

Automatic speech recognition (ASR) systems have increased their performance steadily over the years. The introduction of neural networks (NNs) into the area of speech recognition led to various improvements. Hybrid approaches replaced traditional Gaussian mixture models by learning a function between the input speech features and hidden markov model states in a discriminative fashion. However, these approaches are composed of several independently optimized modules, i.e., an acoustic model, a pronunciation model, and a language model. As they are not optimized jointly, useful information cannot be shared between them. Furthermore, specific knowledge is necessary for each module to retrieve the optimal result.

Recently, sequence-to-sequence (Seq2Seq) models are gaining popularity in the community [1–13] since they fuse all aforementioned modules into a single end-to-end model, which directly outputs characters (chars). Works like [4, 7] have already shown that Seq2Seq models can be superior to hybrid systems [7] if enough data is available. Seq2Seq models can be categorized into approaches based on connectionist temporal classification (CTC) [1, 2], on transducer [11–13] and on attention [3–10].

In CTC, a recurrent neural network (RNN) learns alignments between unlabeled input speech features and a transcript. The basic idea is to assume the conditional independence of the outputs and marginalize over all possible alignments [1]. For ASR, this assumption is not valid, as consecutive outputs are highly correlated. Transducer models relax the conditional independence and add another RNN to learn the dependencies between all previous input speech features and the output [12]. Attention models also combine two RNNs with an additional attention network. One RNN acts as an encoder to transform the input data into a robust feature space. The attention model creates a glimpse given the last hidden layer of the encoder, the previous time-step attention vector and the previous time-step decoder output. The decoder RNN then utilizes the glimpse and the previous decoder output to generate chars [10].

In our work, we propose a novel regularization technique by utilizing an additional decoder to improve attention models. This newly added decoder is optimized on time-reversed labels. Since we primarily focus on improving the training process, we utilize the decoder only during the optimization phase and discard it later in the inference. Thus, the network architecture of a basic attention model is not changed during decoding. A recent study demonstrated that it is beneficial to add a right-to-left (R2L) decoder to a conventional left-to-right (L2R) decoder [14]. The R2L decoder is trained on time-reversed target labels and acts as a regularizer during optimization. Their work focused mainly on the advantage of using the additional information to improve the beam search in decoding. They applied a constant scalar value, which attached a more significant weight on the loss function of the standard L2R decoder. Furthermore, they trained their models on Japanese words whereby label and time-reversed label sequences were equal. Another comparable work has been published in the domain of speech synthesis. In [15], they also utilized a second R2L decoder, combined both losses and added another regularizing function for the L2R and R2L decoder outputs. Similar to [14], they trained only on equal sequence lengths. In the English language, however, byte pair encodings (BPEs) for encoding the target transcripts seem superior [4, 7]. As encoding a time-reversed transcript produces unequal sequence lengths between L2R and R2L decoders, regularization of these sequences is challenging. To the best of our knowledge, an in-depth study on how to solve this

problem and leveraging the newly added decoder during the optimization process has not been done for attention models. Our contributions are to introduce an optimization scheme inspired by [15] for attention models in ASR and utilize the added decoder during the training. Furthermore, we propose two novel regularization terms for equal and unequal output sequence lengths and demonstrate their superiority over conventional attention models.

## 2. PROPOSED METHOD

### 2.1. Attention Model

The standard attentional Seq2Seq model contains three major components: the encoder, the attention module and the decoder. Let  $\mathbf{X} = (\mathbf{x}_1, \dots, \mathbf{x}_t, \dots, \mathbf{x}_T)$  be a given input sequence of  $T$  speech features and let  $\mathbf{y} = (y_1, \dots, y_k, \dots, y_K)$  be the target output sequence of length  $K$ . The encoder transforms the input sequence into a latent space:

$$\begin{aligned} \mathbf{H}^{\text{enc}} &= (\mathbf{h}_1^{\text{enc}}, \dots, \mathbf{h}_t^{\text{enc}}, \dots, \mathbf{h}_T^{\text{enc}}) \\ &= \text{Encoder}(\mathbf{x}_1, \dots, \mathbf{x}_t, \dots, \mathbf{x}_T), \end{aligned} \quad (1)$$

where  $\mathbf{H}^{\text{enc}}$  encodes essential aspects of the input sequence, i.e., characteristics of the speech signal. The resulting hidden encoder states  $\mathbf{H}^{\text{enc}}$  and the hidden decoder state  $\mathbf{h}_{k-1}^{\text{dec}}$  are fed into the attention module to predict proper alignments between the  $t$ -th input and  $k$ -th output sequences:

$$\begin{aligned} \alpha_{k,t} &= \text{Attention}(\mathbf{h}_{k-1}^{\text{dec}}, \mathbf{H}^{\text{enc}}) \\ &= \exp(e_{k,t}) / \sum_{t'=1}^T \exp(e_{k,t'}), \end{aligned} \quad (2)$$

where  $\alpha_k = (\alpha_{k,1}, \dots, \alpha_{k,t})$  are the attention weights and  $e_{k,t}$  is the output of a scoring function:

$$e_{k,t} = \text{Scoring}(\mathbf{h}_{k-1}^{\text{dec}}, \mathbf{h}_t^{\text{enc}}, \alpha_{k-1}). \quad (3)$$

Depending on the task, there are several ways to implement scoring functions. We choose the content-based and location-aware attention from [16] for scoring. Based on the attention weights  $\alpha_k$ , a context vector  $\mathbf{c}_k$  is created to summarize all information in the hidden states of the encoder for the current prediction:

$$\mathbf{c}_k = \sum_t \alpha_{k,t} \mathbf{h}_t^{\text{enc}}. \quad (4)$$

The decoder generates the output distribution using the context vector  $\mathbf{c}_k$  and the decoder hidden state  $\mathbf{h}_{k-1}^{\text{dec}}$ :

$$p(y_k | p_{1:k-1}, \mathbf{X}) \sim \text{Generate}(\mathbf{h}_{k-1}^{\text{dec}}, \mathbf{c}_k), \quad (5)$$

where  $\mathbf{h}_{k-1}^{\text{dec}}$  is a recurrency, usually a long short-term memory (LSTM):

$$\mathbf{h}_k^{\text{dec}} = \text{LSTM}(\mathbf{h}_{k-1}^{\text{dec}}, \mathbf{c}_k, y_{k-1}), \quad (6)$$

with  $y_{k-1}$  being the predicted target label of the previous prediction step. The resulting model is optimized by cross-entropy loss  $\mathcal{L}_{\text{CE}}$ .

### 2.2. Adding a Backward Decoder

For a traditional attention model, the char distribution  $p(\vec{y}_k | p_{1:k-1}, \mathbf{X})$  is generated by a single L2R decoder. This distribution is dependent on the past and thus, has no information about the future context. For this reason, we extend the model by adding a second R2L decoder, which is trained on time-reversed output labels to generate  $p(\overleftarrow{y}_l | p_{L:l+1}, \mathbf{X})$ . The reverse distribution contains beneficial information for the L2R decoder since the decoder has no access to future labels. The R2L decoder contains an individual attention network, which includes a likewise scoring mechanism as the L2R decoder. The decoders learn to create the posterior  $p(\vec{y} | \mathbf{X}, \vec{\theta})$  for the L2R and  $p(\overleftarrow{y} | \mathbf{X}, \overleftarrow{\theta})$  for the R2L case, respectively. Thus,  $\vec{\theta}$  represents the attention and decoder parameters for target labels, which are typically time encoded (e.g., *cat*) and  $\overleftarrow{\theta}$  are the attention and decoder parameter of the time-reversed target labels (e.g., *tac*).

In an ideal case, the posteriors of both decoders should satisfy the following condition:

$$p(\vec{y} | \mathbf{X}, \vec{\theta}) = p(\overleftarrow{y} | \mathbf{X}, \overleftarrow{\theta}), \quad (7)$$

as both networks receive the same amount of information. However, the decoders depend on a different context, i.e., the L2R on past context and the R2L on future context, which results in a similar but not equal training criterion.

### 2.3. Regularization for Equal Sequence Lengths

If we apply chars as target values for training the attention model, we are dealing with equal output sequence lengths since there is no difference between the forward and reverse encoding of a word. Therefore, we extend the loss  $\mathcal{L}_{\text{CE}}$  similar to [15] with a regularization term to retrieve the global loss  $\tilde{\mathcal{L}}$ :

$$\tilde{\mathcal{L}} = \alpha \mathcal{L}_{\text{CE}}(\vec{\theta}) + (1 - \alpha) \mathcal{L}_{\text{CE}}(\overleftarrow{\theta}) + \lambda \Omega(\vec{\theta}, \overleftarrow{\theta}), \quad (8)$$

where  $\alpha$  defines a weighting factor for the losses, and  $\Omega(\vec{\theta}, \overleftarrow{\theta})$  is a regularizer term weighted by  $\lambda$ . We apply the  $L_2$  distance between the decoder outputs  $\vec{y} \in \mathbb{R}^K$  and  $\overleftarrow{y} \in \mathbb{R}^L$  with  $K = L$  as regularization. Thus,  $\Omega(\vec{\theta}, \overleftarrow{\theta})$  is defined it as:

$$\Omega(\vec{\theta}, \overleftarrow{\theta}) = \frac{1}{K} \sum_{k=1}^K \|\vec{y}_k - \overleftarrow{y}_k\|_2. \quad (9)$$

The regularization term forces the network to minimize the distance between outputs of the L2R and R2L decoders. Therefore, the L2R network gets access to outputs that are

based on future context information to utilize its knowledge and increase the overall performance. Note that this kind of regularization is only feasible as we are dealing with equal sequence lengths, which makes it simple to create  $\Omega(\vec{\theta}, \overleftarrow{\theta})$ .

## 2.4. Regularization for Unequal Sequence Lengths

We can extend the approach above by applying BPE units instead of chars. However, in contrast to chars, we face the problem of obtaining unequal sequence lengths  $\vec{y} \in \mathbb{R}^K$  for L2R and  $\overleftarrow{y} \in \mathbb{R}^L$  for R2L decoders with  $K \neq L$ . In fact, we create the same number of BPE units, however, they differ between the L2R and R2L decoders, which results in a difference encoding (e.g., c a t\_ for L2R and t a c\_ for the R2L). Thus, the proposed regularization in Equation 9 is not feasible. We resolve this issue utilizing a differentiable version of the dynamic time warping (DTW) algorithm [17] as a distance measurement between temporal sequences of arbitrary lengths the so-called soft-DTW algorithm. By defining a soft version of the *min* operator with a softening parameter  $\gamma$ :

$$\min^\gamma\{a_1, \dots, a_n\} := \begin{cases} \min_{i \leq n} a_i & \gamma = 0 \\ -\gamma \log \sum_{i=1}^n e^{a_i/\gamma} & \gamma > 0, \end{cases} \quad (10)$$

we can rewrite the soft-DTW loss as a regularization term  $\Omega(\vec{\theta}, \overleftarrow{\theta})$  similar as above:

$$\Omega(\vec{\theta}, \overleftarrow{\theta}) = \min^\gamma\{\langle \mathbf{A}, \Delta(\vec{y}, \overleftarrow{y}) \rangle, \mathbf{A} \in \mathcal{A}_{k,l}\}. \quad (11)$$

Here,  $\langle \cdot, \cdot \rangle$  is the inner product of two matrices,  $\mathbf{A}$  is an alignment matrix of a set  $\mathcal{A}_{k,l} \subset \{0, 1\}^{k,l}$  which are binary matrices that contain paths from  $(1, 1)$  to  $(k, l)$  by only applying  $\downarrow$ ,  $\rightarrow$  and  $\searrow$  moves through this matrix and  $\Delta(\vec{y}, \overleftarrow{y}) := [\delta(\vec{y}_k, \overleftarrow{y}_l)]$  is defined by a distance function  $\delta(\vec{y}_k, \overleftarrow{y}_l)$  (e.g., Euclidean distance). Based on the inner product, we retrieve an alignment cost for all possible alignments between  $\vec{y}$  and  $\overleftarrow{y}$ . Since we force the network to also minimize  $\Omega(\vec{\theta}, \overleftarrow{\theta})$ , it has to learn a good match between the different sequence lengths of the L2R and R2L decoders.

## 3. EXPERIMENTS

### 3.1. Training Details

All our experiments are evaluated on the smaller dataset TEDLIUMv2 [18] and the larger dataset LibriSpeech [19]. TEDLIUMv2 has approximate 200 h of training data, whereas LibriSpeech contains of 960 h of training data.

We preprocess both datasets by extracting 80-dimensional log Mel features and adding the corresponding pitch features, which results in an 83-dimensional feature vector. Furthermore, we apply chars and BPE units as target labels. The chars are directly extracted from the datasets, whereas the BPE units are created by a language-independent sub-word

tokenizer. For all experiments, we select 100 BPE units, which seem sufficient for our approach.

The proposed architecture is created in the ESPnet toolkit [20] and trained by the standard training script for attention models. The encoder is built up by four bidirectional long short-term memory projected (BLSTMP) layers of dimension 1024. Each decoder contains a single LSTM network with 1024 cells and a linear output layer.

We perform a three-stage training scheme inspired by [15]. In the first stage, we train a standard attentional network with a L2R decoder. Then, we apply the pretrained encoder, freeze its weights and train the R2L model. Finally, we combine both networks into one model to receive the final architecture. In all stages, we optimize the network with Adadelta initialized with an  $\epsilon = 10^{-8}$ . If we do not observe any improvement of the accuracy on the validation set, we decay  $\epsilon$  by a factor of 0.01 and increment a patience counter by one. We apply an early stopping of the training if the patient counter exceeds three. The batch-size is set to 30 for all training steps.

Depending on the target labels in the third training stage, i.e., chars or BPE units, we deploy two different techniques to regularize the L2R decoder. For chars, forward sequences  $\vec{y}$  and backward sequences  $\overleftarrow{y}$  have equal lengths. Thus, we add a  $L_2$  regularizer identical to Equation 8 and scale it with  $\lambda = 1$  for the smaller and  $\lambda = 0.1$  for the bigger dataset. On the other hand, for BPE units, we utilize the soft-DTW from Equation 11 as a regularizer since it represents a distance measurement between the unequal sequence lengths  $\vec{y}$  and  $\overleftarrow{y}$ , which we want to minimize. Here, we set  $\gamma = 1$  and scale the regularization with  $\lambda = 10^{-4}$  for both datasets. Besides the added regularizations for chars and BPE units, we regularize the L2R network further by applying  $\alpha = 0.9$  in all the experiments. Thereby, we ensure that the overall training is focused on the L2R decoder network. During decoding, we remove the R2L network since it is only necessary in the training stages. As a result, we are not changing or adding complexity to the final model during decoding.

### 3.2. Benchmark Details

We evaluate our approach on five different setups. In the *Forward* setup, a model is trained with a standard L2R decoder, which is the baseline for all experiments. The second setup is the *Forward* setup, where a model is trained on time-reversed target labels, which results in a R2L decoder. We perform a similar approach in *Backward Fixed*, however, we apply the pretrained encoder from the L2R model and freeze its weights during training. To solely investigate the effect of the R2L decoder as regularization, we define the *Dual Decoder* setup. The model consists of a shared encoder from *Forward* and the pretrained L2R and R2L decoder from the *Forward* and *Backward Fixed* setups. The combined model is trained with  $\alpha = 0.9$  and  $\lambda = 0.0$ . In the last *Dual Decoder Reg* setup,

**Table 1.** Evaluation of our approach on TEDLIUMv2 and LibriSpeech with the resulting WERs (%) for all five setups

Methods	TEDLIUMv2 [18]				LibriSpeech [19]							
	char		BPE		char				BPE			
	dev	test	dev	test	dev-clean	dev-other	test-clean	test-other	dev-clean	dev-other	test-clean	test-other
Forward	16.77	17.32	17.83	18.00	7.69	20.67	7.72	21.63	7.59	20.98	7.67	21.92
Backward	18.12	18.47	18.57	17.99	7.60	20.78	7.54	21.83	7.53	20.94	7.60	21.71
Backward Fixed	23.34	23.77	25.55	25.01	11.39	28.36	11.75	28.53	12.07	28.63	12.39	29.06
Dual Decoder	16.47	17.12	17.70	18.08	7.29	20.99	7.60	22.00	7.46	21.29	7.70	22.01
Dual Decoder Reg	<b>15.68</b>	<b>15.94</b>	<b>16.75</b>	<b>17.42</b>	<b>7.24</b>	<b>19.96</b>	<b>7.02</b>	<b>20.95</b>	<b>7.17</b>	<b>20.01</b>	<b>7.33</b>	<b>20.63</b>

which is similar to the *Dual Decoder* setup, we include the  $L_2$  distance [15] for chars and the soft-DTW loss [17] for BPE units as target labels. Instead of performing the forward and backward beam search as in [14], we only apply a forward beam search deploying the L2R decoder with a beam size of 20.

### 3.3. Results

In Table 1, we present the results of our approach applying chars and BPE units for the TEDLIUMv2 [18] and LibriSpeech [19] datasets.

For the smaller dataset TEDLIUMv2, we observe a clear difference in WERs between the *Forward* and the *Backward* setup. Ideally, the performance of these setups should be equal, as both networks receive the same amount of information. However, we observe an absolute difference of 1% WER for all evaluation sets, except for the test BPE set. One explanation for this variation may be that the *Backward* setup is more complex. Since the dataset contains only around 220h of training data, the number of reverse training samples could not be sufficient. In the bigger dataset LibriSpeech, the first two setups obtain nearly the same WER with only a minor difference. This dataset contains nearly five times the data of the smaller dataset and therefore, the network in the *Backward* setup receives enough reverse training examples. It seems, that the amount of data seems crucial for the R2L decoder to satisfy Equation 7.

In the *Backward Fixed* setup, we can verify the strong dependency of the decoder, relying on the high-level representation of features created by the encoder. Although we do not change the information of the target labels by reversing them, the fixed encoder from the *Forward* setup learned distinct, high-level features, which are based on past context. We observe this by a decline of the WERs in both datasets. Even though, the utilized BLSTMPs in the encoder network receive the complete feature sequence in the input space, they generate high-level features based on past label context, since they do not have access to future labels. As a result, the R2L model applying a fixed encoder from the *Forward* setup is worse compared to the trainable encoder in the R2L model.

In the *Dual Decoder* setup, we follow the idea of [14] to apply the R2L model as a regularizer of the L2R network. Interestingly, the R2L decoder is not able to effectively support the L2R decoder. We recognize only a slight improvement of the WER, which is not consistent in both datasets. Therefore, a simple weighting of the loss during training is not sufficient to enhance the L2R decoder. One reason might be that the L2R decoder receives only implicit information from the R2L decoder by weighting the losses, which is considered not valuable for the optimization of the L2R decoder.

To induce valuable information, we add our proposed regularization terms in the last *Dual Decoder Reg* setup. The overall network is forced to minimize the added regularization terms explicitly. The L2R decoder can directly utilize information of the R2L decoder to improve its predictions. We receive the overall best WER for the last setup. For the TEDLIUMv2 dataset, we recognize an average relative improvement of 7.2% for the char and 4.4% for the BPE units. For the LibriSpeech dataset, we are able to receive an average relative improvement of 4.9% for the char and 5.1% for the BPE units.

Compared to other state-of-the-art approaches, we decided not to include CTC and a language model since their integration into our approach raises several issues as we deal with a shared encoder and unequal sequence lengths among the two decoders.

## 4. CONCLUSION

Our work presents a novel way to integrate a second decoder for attention models during the training phase. The proposed regularization terms support the standard L2R model to utilize future context information from the R2L decoder, which is usually not available during optimization. We solved the issue of regularizing unequal sequence lengths, which arise applying BPE units as target values, by adding a soft version of the DTW algorithm. We outperform conventional attention models independent of the dataset size. Our regularization technique is simple to integrate into a conventional training scheme, does not change the overall complexity of the standard model, and only adds optimization time.

## 5. REFERENCES

- [1] Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber, “Connectionist Temporal Classification: Labelling Unsegmented Sequence Data with Recurrent Neural Networks,” in *Proceedings of the 23rd international conference on Machine learning*. ACM, 2006, pp. 369–376.
- [2] Alex Graves and Navdeep Jaitly, “Towards End-to-End Speech Recognition with Recurrent Neural Networks,” in *International conference on machine learning*, 2014, pp. 1764–1772.
- [3] I Sutskever, O Vinyals, and QV Le, “Sequence to sequence learning with neural networks,” *Advances in NIPS*, 2014.
- [4] Zoltán Tüske, Kartik Audhkhasi, and George Saon, “Advancing sequence-to-sequence based speech recognition,” *Proc. Interspeech 2019*, pp. 3780–3784, 2019.
- [5] Chao Weng, Jia Cui, Guangsen Wang, Jun Wang, Chengzhu Yu, Dan Su, and Dong Yu, “Improving Attention Based Sequence-to-Sequence Models for End-to-End English Conversational Speech Recognition,” in *Interspeech*, 2018, pp. 761–765.
- [6] William Chan, Navdeep Jaitly, Quoc Le, and Oriol Vinyals, “Listen, attend and spell: A neural network for large vocabulary conversational speech recognition,” in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2016, pp. 4960–4964.
- [7] Chung-Cheng Chiu, Tara N Sainath, Yonghui Wu, Rohit Prabhavalkar, Patrick Nguyen, Zhifeng Chen, Anjali Kannan, Ron J Weiss, Kanishka Rao, Ekaterina Gonnina, et al., “State-of-the-art speech recognition with sequence-to-sequence models,” in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 4774–4778.
- [8] Jan Chorowski, Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio, “End-to-end continuous speech recognition using attention-based recurrent nn: First results,” *arXiv preprint arXiv:1412.1602*, 2014.
- [9] Dzmitry Bahdanau, Jan Chorowski, Dmitriy Serdyuk, Philemon Brakel, and Yoshua Bengio, “End-to-end attention-based large vocabulary speech recognition,” in *2016 IEEE international conference on acoustics, speech and signal processing (ICASSP)*. IEEE, 2016, pp. 4945–4949.
- [10] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio, “Neural machine translation by jointly learning to align and translate,” *arXiv preprint arXiv:1409.0473*, 2014.
- [11] Alex Graves, “Sequence Transduction with Recurrent Neural Networks,” *arXiv preprint arXiv:1211.3711*, 2012.
- [12] Alex Graves, Abdel-rahman Mohamed, and Geoffrey Hinton, “Speech recognition with deep recurrent neural networks,” in *2013 IEEE international conference on acoustics, speech and signal processing*. IEEE, 2013, pp. 6645–6649.
- [13] Hasim Sak, Matt Shannon, Kanishka Rao, and Françoise Beaufays, “Recurrent Neural Aligner: An Encoder-Decoder Neural Network Model for Sequence to Sequence Mapping,” in *Interspeech*, 2017, pp. 1298–1302.
- [14] Masato Mimura, Shinsuke Sakai, and Tatsuya Kawahara, “Forward-Backward Attention Decoder,” in *Interspeech*, 2018, pp. 2232–2236.
- [15] Yibin Zheng, Xi Wang, Lei He, Shifeng Pan, Frank K Soong, Zhengqi Wen, and Jianhua Tao, “Forward-Backward Decoding for Regularizing End-to-End TTS,” *arXiv preprint arXiv:1907.09006*, 2019.
- [16] Jan K Chorowski, Dzmitry Bahdanau, Dmitriy Serdyuk, Kyunghyun Cho, and Yoshua Bengio, “Attention-based models for speech recognition,” in *Advances in neural information processing systems*, 2015, pp. 577–585.
- [17] Marco Cuturi and Mathieu Blondel, “Soft-DTW: a differentiable loss function for time-series,” in *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR.org, 2017, pp. 894–903.
- [18] Anthony Rousseau, Paul Deléglise, and Yannick Esteve, “Enhancing the TED-LIUM corpus with selected data for language modeling and more TED talks,” in *LREC*, 2014, pp. 3935–3939.
- [19] Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur, “Librispeech: an ASR corpus based on public domain audio books,” in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2015, pp. 5206–5210.
- [20] Shinji Watanabe, Takaaki Hori, Shigeki Karita, Tomoki Hayashi, Jiro Nishitoba, Yuya Unno, Nelson Enrique Yalta Soplín, Jahn Heymann, Matthew Wiesner, Nanxin Chen, et al., “Espnet: End-to-end speech processing toolkit,” *arXiv preprint arXiv:1804.00015*, 2018.