Effect of Adding Positional Information on Convolutional Neural Networks for End-to-End Speech Recognition

Jinhwan Park, Wonyong Sung

Department of Electrical and Computer Engineering
Seoul National University

{bnoo,wysung}@snu.ac.kr

Abstract

Attention-based models with convolutional encoders enable faster training and inference than recurrent neural network-based ones. However, convolutional models often require a very large receptive field to achieve high recognition accuracy, which not only increases the parameter size but also the computational cost and run-time memory footprint. A convolutional encoder with a short receptive field length can suffer from looping or skipping problems when the input utterance contains the same words as nearby sentences. We believe that this is due to the insufficient receptive field length, and try to remedy this problem by adding positional information to the convolution-based encoder. It is shown that the word error rate (WER) of a convolutional encoder with a short receptive field size can be reduced significantly by augmenting it with positional information. Visualization results are presented to demonstrate the effectiveness of adding positional information. The proposed method improves the accuracy of attention models with a convolutional encoder and achieves a WER of 10.60% on TED-LIUMv2 for an end-to-end speech recognition task.

Index Terms: speech recognition, convolutional networks, positional encoding

1. Introduction

Many automatic speech recognition (ASR) algorithms employ recurrent neural networks (RNNs) because of their ability to recognize sequences [1, 2, 3]. In particular, attention-based models are prevalent for ASR [4], and they usually employ RNNs for the encoder and decoder. However, RNN-based models are very difficult to parallelize, resulting in severe restrictions when implementing them on graphics processing units (GPUs) or embedded devices for low power and high speed. The efficiency of implementation is very low, especially when the batch size is very small.

Recently, non-recurrent structures, such as convolution [5] and self-attention [6], have actively been studied for application to sequential tasks to achieve computational efficiency. Because non-recurrent structures can process multiple input frames at a time, the number of parameter accesses from DRAM can be reduced significantly. In particular, for speech recognition, convolutional neural networks (CNNs) [7, 8] and self-attention networks [9] have been successfully applied to attention-based models and have shown lower word error rate (WER) than RNN-based models. We focus on convolution-based models that require only a limited receptive field size for the input. Note that using a limited-length input for speech recognition is very advantageous for low-latency system design [10]. Although we focused on attention models that are not capable of online inference, the proposed method can be extended to streaming inference when local attentions [11, 12] are applied [13].

Convolutional models for speech recognition often require a large receptive field length to observe a long input context. Note that the receptive field size is determined by the filter length in each layer and the depth of the model. WER increases drastically when the receptive field length is not sufficient. This is mainly due to looping or skipping problems, which are frequently observed when the encoder of the attention model contains similar outputs at different time-steps [14]. Employing a large filter size can help solve this problem, but it demands a large parameter size or computational overhead. Depth-wise convolutions can be used to reduce the parameter size overhead, but they cannot solve the large intermediate memory requirement and the delay problem for the input [15, 16, 17]. We consider that the high error rate of models with small receptive fields is caused by the time-invariant property of convolution. When similar pronunciation is repeated in the input speech, a convolutional encoder also yields very close output values. This property can be helpful in terms of generalization, but it results in unstable attention because the model cannot distinguish similar values at different time steps.

In this study, we analyzed the error pattern when convolutional models only have small receptive field sizes. Then, it is shown that the recognition accuracy of small receptive-field models can be improved by adding the simplest form of positional encoding, which is used in the Transformer architecture [18]. The encoder output is visualized to prove the effectiveness of positional information. The proposed method improves the accuracy of attention models with convolutional encoders, especially when the models have small filter sizes. We achieved 10.60% WER on TED-LIUMv2 using the single end-to-end model.

2. Related Works

The effect of positional information on CNNs for image recognition was recently studied in [19]. The study has shown that a CNN with a large receptive field size inherently learns positional information; the results of recognition rely heavily on positional information. They have shown that a sufficient receptive field size and zero-padding are required for convolutional models to learn positional information.

External positional information has been applied from the early convolutional models for sequential tasks. Trainable positional embedding vectors were added to word embeddings for training convolutional language models [5]. This approach is not suitable for speech recognition, where the length of the inputs is longer and varies much by data. Sinusoidal positional encoding has recently been proposed with Transformer architecture [6]. Sinusoidal positional encoding can be effective even when the input sequence is longer than the ones in the training data.

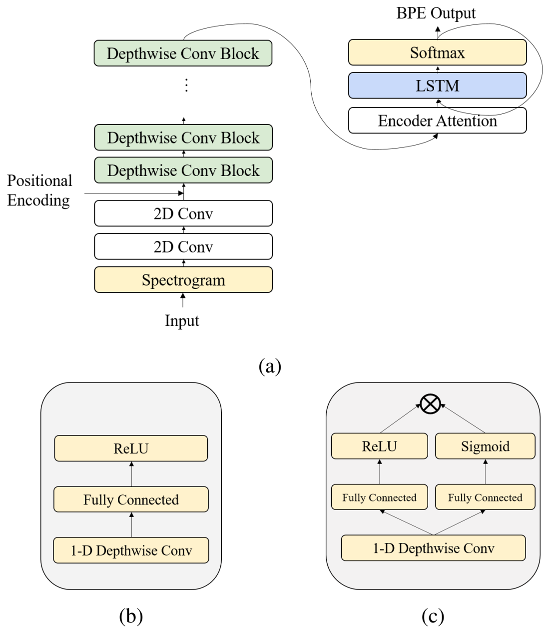


Figure 1: *(a) The attention-based model with 1-D depthwise convolutions and positional information for encoder. (b) A depthwise convolutional block and (c) a block with gating structure.*

Positional information has often been considered to help stabilize training attention weights. Attention feedback [20] and location-based attention [18] use the past attention location history to compute current attention weights. Soft-window pretraining uses auxiliary loss that encourages attention weights to be aligned with the input time steps [15]. Through our experimental results, we verified that applying positional encoding has a similar effect to these methods without modifying the model structure or training processes.

3. Model Description

The attention-based speech recognition model is based on [21], except that 1-D depth-wise convolutional layers are used for the encoder. Each layer is computed as follows:

$$\begin{aligned} x'_{t,k} &= \sum_{i=-(T-1)/2}^{(T-1)/2} W_{i,k} \cdot x_{t+i,k} \\ \mathbf{h}_t &= f(\mathbf{V}\mathbf{x}'_t + \mathbf{b}), \end{aligned} \tag{1}$$

where f is the activation function, and $\mathbf{W} \in \mathbb{R}^{T\times D}, \mathbf{V} \in \mathbb{R}^{D\times D}$ are the trainable variables. T and D denote the width and output dimensions of a convolution, respectively. For the encoder, we applied two 2-D convolutions to the input features in the frequency- and time-axis. The 1-D depth-wise convolutional layers are stacked on top of the 2-D convolutions as shown in Figure 1. We used a residual connection for every two convolutional layers. Layer normalization [22] was applied after residual connections for better convergence.

In the decoder, the attention weights $\alpha_{i,t}$, the energies $e_{i,t}$ for the encoder time-step t, and decoder step i are computed as:

$$\begin{aligned} e_{i,t} &= \mathbf{v}_e^T \cdot \tanh(\mathbf{W}[\mathbf{s}_i, \mathbf{h}_t, \beta_{i,t}]) \\ \boldsymbol{\alpha}_i &= \mathrm{softmax}(\mathbf{e}_i), \end{aligned} \tag{2}$$

where v is a trainable vector, $\mathbf{W}$ is a trainable matrix, $\mathbf{s}_i$ is the current decoder state, and $\mathbf{h}_t$ is the output of the last layer of the encoder. $\beta_{i,t}$ is the attention weight feedback which is defined as:

$$\beta_{i,t} = \sigma(\mathbf{v}_\beta^T \mathbf{h}_t) \sum_{k=1}^{i-1} \alpha_{k,t} \tag{3}$$

The attention context vector is given as:

$$\mathbf{c}_i = \sum_t \alpha_{i,t}\mathbf{h}_t. \tag{4}$$

The decoder is a single-layer long short-term memory [23] (LSTM) that is computed as follows:

$$\mathbf{s}_i = \mathrm{LSTM}(\mathbf{s}_{i-1}, \mathbf{y}_{i-1}, \mathbf{c}_{i-1}). \tag{5}$$

Usually, the positional encoding vector in Transformer architecture [6] is added to the input of the encoder. For Transformer architecture in speech recognition, a linear transformation is often applied to the positional encoding before being added to the input [9]. In our experiments, the best performance was obtained when the positional encoding vector was concatenated to the output of 2D convolutions. We also tried adding or concatenating it to the output of the convolutional encoders, but it made training diverge in the initial stage. We used the positional encoding vector as proposed in [6], which is computed as follows:

$$\begin{aligned} pe_{i,2k} &= \sin(i/10000^{(2k)/Dmodel}) \\ pe_{i,2k+1} &= \cos(i/10000^{(2k+1)/Dmodel}) \end{aligned} \tag{6}$$

4. Experimental Results

Table 1: *TED-LIUM release 2 results of the models with different filter size. Pos. denotes that positional encoding is applied.*

Model	**WER [%]**	
	dev	test
Conv. 15x2048 (T=3)	23.01	18.41
Conv. 15x2048 (T=5)	18.06	15.18
Conv. 15x2048 (T=7)	17.03	14.75
Conv. 15x2048 (T=11)	15.18	12.95
Conv. 15x2048 (T=15)	15.41	13.19
Conv. 15x2048 (T=3) + Pos.	15.75	13.37
Conv. 15x2048 (T=5) + Pos.	15.24	12.58
Conv. 15x2048 (T=7) + Pos.	15.57	13.15
Conv. 15x2048 (T=11) + Pos.	14.78	12.58
Conv. 15x2048 (T=15) + Pos.	14.87	13.14

The experiments were performed using the RETURNN framework [24]. TED-LIUM release 2 [25] was used for training, which contains 200 hours of speech. We followed the data preprocessing pipeline used in [26]. We used a 40-dimensional mel-frequency cepstral coefficient for the input features, which were extracted every 10 ms with a 25 ms window size. Byte-pair encoding (BPE) [27] with a vocabulary size of 1K was used for the output labels. The layer-wise pretraining was only applied to LSTM-based models because it lowers the accuracy when used for convolutional models. The decoder was a single-layer 1000-dimensional LSTM algorithm for all models. The configuration files for the experiments are available online.[1]

[1]`https://github.com/car3936/returnn-exp-jinh`

Transcription (WadeDavis_2003_305.16_327.12)
... depend they have a **curious** language and marriage rule which is called linguistic exogamy you must marry someone who speaks a different language and this is all rooted in the mythological past yet the **curious** thing is in these long houses where there are six or seven languages spoken
Without positional encoding
... depend they have a **curious** things and these long houses where they're six or seven languages spoken
With positional encoding
... depend they have a **curious** language and marriage rule which is called linguistic exotic me you must marry someone who speaks a different language and this is all rooted in mythological past get the **curious** things and these long houses were there six or seven languages spoken

Figure 2: *The original transcript and the decoded results with and without the positional encoding.*

4.1. Effect of receptive field size

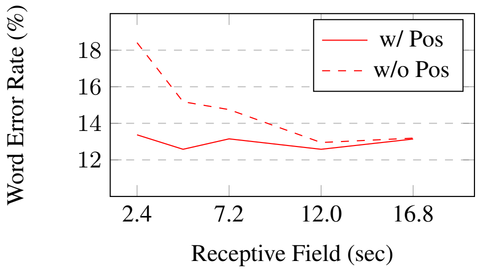


Figure 3: *Test WER on TED-LIUMv2 comparing models with different receptive field size.*

The experimental results of the convolutional models on TED-LIUMv2 are shown in Table 1. A 15x2048 model denotes that 15 SGCN layers with $D = 2048$ are stacked. We trained the 15x2048 convolutional model while changing the filter width of depthwise convolutions. We applied max-pooling with a size of 2 for the initial three convolutional layers. All the models have approximately 84M parameters regardless of the filter width because the number of parameters for depthwise convolution is very small. The results show that positional encoding improved the accuracy of models consistently, especially for those with small filter sizes. Fig. 3 shows the WER of the models with different receptive field sizes. The length of the receptive field was calculated as $(W{-}1) \times 80ms \times \#layers$, where W is the filter width. The WER of the models without positional encoding sharply increased when the receptive field length was reduced. In comparison, when positional encoding was employed, the WER was not significantly affected by the length of the receptive field. This result clearly demonstrates the effect of positional encoding on CNN-based models.

We compared the decoding results of two CNN-based models with and without positional encoding. As shown in Fig. 2, the original transcription possesses two occurrences of 'curious' approximately 30 words apart. The decoding result without positional encoding yields a shortened sentence that skips the words between the two occurrences of 'curious'. However, decoding with positional encoding faithfully shows all the words. Such skipping occurred frequently over the entire test set. Fig. 4 shows a histogram of the number of errors according to the lengths of the transcriptions. We plot the results of the LSTM and the convolutional models with and without positional encoding. The number of errors differed significantly in longer sequences that are more likely to contain repeated words. This

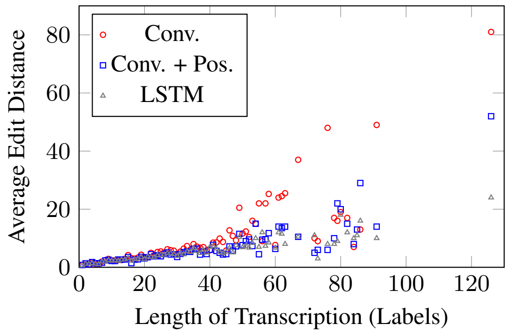
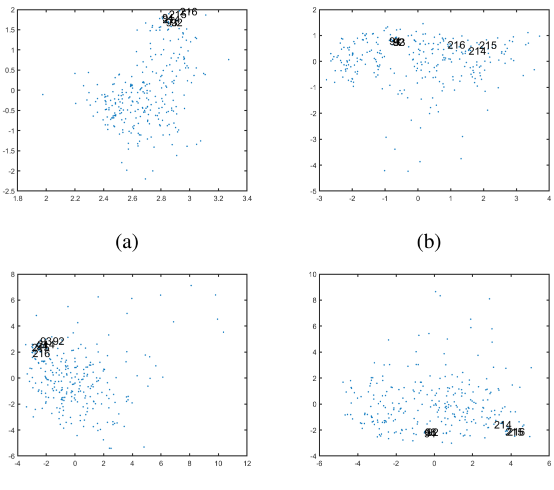


Figure 4: *The average edit distance of test set according to the length of transcription.*



(a) (b)

(c) (d)

Figure 5: *The visualization of encoder output using PCA. The first two principal components are used for visualization. The points of 92-94th and 214-216th steps are indicated with text, which correspond to pronunciation of the word 'curious'. (a) filter width = 3 (b) filter width = 3 with positional encoding. (c) filter width = 5 (d) filter width = 5 with positional encoding.*

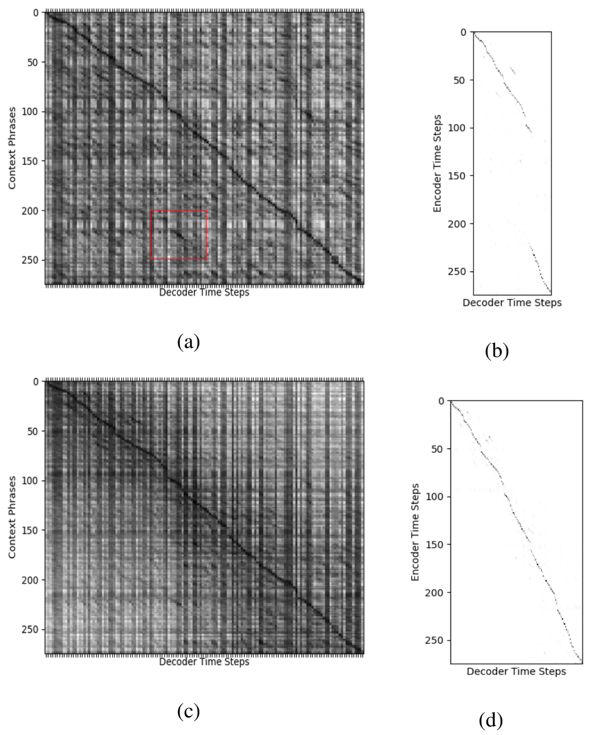


Figure 6: *Attention energy* $\mathbf{e}_i$ ***(left)*** *and weight* $\boldsymbol{\alpha}_i$ ***(right)*** *of models (a) (b) without and (c) (d) with positional encoding are shown. Darker pixels indicate higher values.*

suggests that the convolutional model is vulnerable to skipping problems, and applying positional encoding can alleviate this issue.

4.2. Visualization

We used principal component analysis (PCA) [28] to analyze the effect of positional encoding on the output of the encoder. Fig. 5 shows a visualization of the encoder outputs of the convolutional models. Each point corresponds to a single time-step output of the encoder. We focused on the results of the 92-94th and 214-216th step outputs, which correspond to the first and second occurrences of 'curious', respectively. In Fig. 5 (a) and (c), the encoder output of the convolutional models has a similar component when the input speech contained repeated pronunciation. When positional information was applied, the encoder outputs were located at a distance, as shown in Fig. 5 (b) and (d). This clearly indicates that positional encoding makes the outputs discriminative when similar words are applied.

The attention energy $\mathbf{e}_i$ and the attention weight $\boldsymbol{\alpha}_i$ in Eq. (3) is plotted in Fig. 6. The horizontal and vertical axes correspond to the decoder and encoder time steps, respectively. The label from transcription was given to the decoder every step for plotting the attention energy, while the previous output was used for the attention weight. In Fig. 6 (a), the energy has a high value in the area indicated by the red box. This causes a misalignment of attention in the inference time and results in skipping, as shown in Fig. 6 (b). In Fig. 6 (c), the energy is more concentrated around the diagonal components compared to Fig. 6 (a). This is a desired property when training the attention-based model, since it prevents the model from diverging in the initial stage of training [15].

Table 2: *Experimental results of convolutional models with different sizes. LSTM and Transformer models are shown for comparison. Decoder is a single-layer 1000-dimensional LSTM for all the models.*

Model	**WER [%]**		**Params.**
	dev	test	
Bidir. LSTM 6x1024 [26]	11.7	10.5	161M
Transformer [26]	14.7	12.5	100M
Bidir. LSTM 6x1024	12.65	10.57	161M
Bidir. LSTM 6x1024 + Pos.	17.70	12.35	161M
Unidir. LSTM 6x1536	16.78	14.42	127M
filter width = 3			
Conv. 15x2048	23.01	18.41	84M
Conv. 15x2048 + Pos.	15.75	13.37	84M
Conv. 25x2048 + Pos.	14.51	11.89	127M
Gated Conv. 35x1024 + Pos.	14.94	12.73	80M
Gated Conv. 35x1536 + Pos.	13.02	11.04	229M
filter width = 5			
Gated Conv. 35x2048	14.14	11.16	313M
Gated Conv. 35x2048 + Pos.	12.81	10.60	313M

4.3. Comparison with other models

Table 2 shows the results of the convolutional models with large parameter sizes. The results of LSTM- and self-attention-based models are also shown for comparison. The experimental results show that using positional encoding improves the accuracy of deeper structures. Note that bidirectional LSTM and self-attention models consider the entire context of the input, which is not desirable for deployment. Unidirectional LSTM has higher WERs than convolutional models with a comparable number of parameters.

The proposed method can be applied to other convolutional structures. We tried gated convolution [7], which has been successfully applied to speech recognition tasks. With gated convolution, we achieved a 10.60% WER on the TED-LIUM v2 test set.

5. Concluding Remarks

In this study, we demonstrated that convolutional models with small filter sizes lack the ability to identify positional information, which incurs looping or skipping problem in end-to-end speech recognition. By adding explicit positional encoding, we prevented severe performance degradation of models with small receptive fields. The proposed method did not require any modification to model structures or training algorithms. It also had almost no computational overhead. Since convolutional encoders support fast training and inference, the proposed method is suitable for developing an on-device low-power speech recognition system.

6. Acknowledgements

This work was in part supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIP) (No. 2018R1A2A1A05079504). This work was also partly supported by the Google AI Focused Research Awards Program awarded to Wonyong Sung.

7. References

[1] D. Amodei, S. Ananthanarayanan, R. Anubhai, J. Bai, E. Battenberg, C. Case, J. Casper, B. Catanzaro, Q. Cheng, G. Chen *et al.*, "Deep speech 2: End-to-end speech recognition in english and mandarin," in *International conference on machine learning*, 2016, pp. 173–182.

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

[3] 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," *Proc. Interspeech 2019*, pp. 231–235, 2019.

[4] W. Chan, N. Jaitly, Q. Le, and O. 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.

[5] J. Gehring, M. Auli, D. Grangier, D. Yarats, and Y. N. Dauphin, "Convolutional sequence to sequence learning," in *International Conference on Machine Learning (ICML)*, 2017, pp. 1243–1252.

[6] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in neural information processing systems*, 2017, pp. 5998–6008.

[7] V. Liptchinsky, G. Synnaeve, and R. Collobert, "Letter-based speech recognition with gated ConvNets," *arXiv preprint arXiv:1712.09444*, 2017.

[8] J. Li, V. Lavrukhin, B. Ginsburg, R. Leary, O. Kuchaiev, J. M. Cohen, H. Nguyen, and R. T. Gadde, "Jasper: An end-to-end convolutional neural acoustic model," in *Proceedings of Interspeech*, 2019.

[9] N.-Q. Pham, T.-S. Nguyen, J. Niehues, M. Müller, and A. Waibel, "Very deep self-attention networks for end-to-end speech recognition," *Proc. Interspeech 2019*, pp. 66–70, 2019.

[10] V. Pratap, Q. Xu, J. Kahn, G. Avidov, T. Likhomanenko, A. Hannun, V. Liptchinsky, G. Synnaeve, and R. Collobert, "Scaling up online speech recognition using convnets," 2020.

[11] C.-C. Chiu and C. Raffel, "Monotonic chunkwise attention," *International Conference on Learning Representations (ICLR)*, 2018.

[12] M.-T. Luong, H. Pham, and C. D. Manning, "Effective approaches to attention-based neural machine translation," in *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, 2015, pp. 1412–1421.

[13] K. Kim, K. Lee, D. Gowda, J. Park, S. Kim, S. Jin, Y.-Y. Lee, J. Yeo, D. Kim, S. Jung *et al.*, "Attention based on-device streaming speech recognition with large speech corpus," in *2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*. IEEE, 2019, pp. 956–963.

[14] J. Chorowski and N. Jaitly, "Towards better decoding and language model integration in sequence to sequence models," *Proc. Interspeech 2017*, pp. 523–527, 2017.

[15] A. Hannun, A. Lee, Q. Xu, and R. Collobert, "Sequence-to-sequence speech recognition with time-depth separable convolutions," in *Proceedings of Interspeech*, 2019.

[16] S. Kriman, S. Beliaev, B. Ginsburg, J. Huang, O. Kuchaiev, V. Lavrukhin, R. Leary, J. Li, and Y. Zhang, "Quartznet: Deep automatic speech recognition with 1d time-channel separable convolutions," in *2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2020, pp. 6124–6128.

[17] L. Lee, J. Park, and W. Sung, "Simple gated convnet for small footprint acoustic modeling," in *2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*. IEEE, 2019.

[18] J. K. Chorowski, D. Bahdanau, D. Serdyuk, K. Cho, and Y. Bengio, "Attention-based models for speech recognition," in *Advances in neural information processing systems*, 2015, pp. 577–585.

[19] M. A. Islam, S. Jia, and N. D. Bruce, "How much position information do convolutional neural networks encode?" *International Conference on Learning Representations (ICLR)*, 2020.

[20] Z. Tu, Z. Lu, Y. Liu, X. Liu, and H. Li, "Modeling coverage for neural machine translation," in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2016, pp. 76–85.

[21] A. Zeyer, K. Irie, R. Schlüter, and H. Ney, "Improved training of end-to-end attention models for speech recognition," *Proc. Interspeech 2018*, pp. 7–11, 2018.

[22] J. L. Ba, J. R. Kiros, and G. E. Hinton, "Layer normalization," *arXiv preprint arXiv:1607.06450*, 2016.

[23] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[24] P. Doetsch, A. Zeyer, P. Voigtlaender, I. Kulikov, R. Schlüter, and H. Ney, "Returnn: The rwth extensible training framework for universal recurrent neural networks," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2017, pp. 5345–5349.

[25] A. Rousseau, P. Deléglise, and Y. Esteve, "Enhancing the TED-LIUM corpus with selected data for language modeling and more ted talks." in *LREC*, 2014, pp. 3935–3939.

[26] A. Zeyer, P. Bahar, K. Irie, R. Schlüter, and H. Ney, "A comparison of transformer and LSTM encoder decoder models for ASR," in *IEEE Automatic Speech Recognition and Understanding Workshop, Sentosa, Singapore*, 2019.

[27] R. Sennrich, B. Haddow, and A. Birch, "Neural machine translation of rare words with subword units," in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2016, pp. 1715–1725.

[28] H. Hotelling, "Analysis of a complex of statistical variables into principal components." *Journal of educational psychology*, vol. 24, no. 6, p. 417, 1933.