INTERSPEECH 2021
30 August – 3 September, 2021, Brno, Czechia

An improved wav2vec 2.0 pre-training approach using enhanced local dependency modeling for speech recognition

Qiu-shi Zhu[1], *Jie Zhang*[1], *Ming-hui Wu*[2], *Xin Fang*[1,2], *Li-rong Dai*[1]

[1]NEL-SLIP, University of Science and Technology of China (USTC), Hefei, China
[2]iFlytek Research, iFlytek Co., Ltd., Hefei, China

qszhu@mail.ustc.edu.cn, {mhwu,xinfang}@iflytek.com, {jzhang6,lrdai}@ustc.edu.cn

Abstract

Wav2vec 2.0 is a recently proposed self-supervised pre-training framework for learning speech representation. It utilizes a transformer to learn global contextual representation, which is effective especially in low-resource scenarios. Besides, it was shown that combining convolution neural network and transformer to model both local and global dependencies is beneficial for e.g., automatic speech recognition (ASR), natural language processing (NLP). However, how to model the local and global dependence in pre-training models is still an open question in the speech domain. In this paper, we therefore propose a new transformer encoder for enhancing the local dependency by combining convolution and self-attention modules. The transformer encoder first parallels the convolution and self-attention modules, and then serialized with another convolution module, sandwiched by a pair of feed forward modules. Experimental results show that the pre-trained model using the proposed method can reduce the word error rate (WER) compared to the reproduced wav2vec 2.0 at the cost of slightly increasing the size of training parameters.

Index Terms: Speech recognition, pre-training, wav2vec 2.0, transformer, low-resource, local and global dependence.

1. Introduction

There are nearly 7000 languages in the world [1], while currently only a few of them (e.g., English, Chinese) have sufficient annotation data, and most of them are scarce in data resource, i.e., low resource. For the automatic speech recognition (ASR) of these low-resource languages, it is thus difficult to directly train models with an acceptable performance due to the lack of data resource. The unsupervised or self-supervised pre-training can be leveraged to solve this problem. Unsupervised or self-supervised objects are particularly attractive for learning representations, because they can take advantage of unlabeled data, which is much cheaper and scalable compared to the datasets that require annotation.

It was shown in [2] that the unsupervised representation learning is effective in case the labeled data is scarce. In order to make high-level speech information more accessible, speech representation learning [3–8] is dedicated to modeling a transformation from the surface features, e.g., waveforms, spectrograms, to downstream tasks. Through feature-based speech representation extraction or fine-tuning as a part of the downstream model, the learned model can then be applied to the subsequent ASR. Recently, many unsupervised learning approaches have been proposed for speech representation. For example, an autoregressive predictive coding (APC) method was proposed to reconstruct the future frames conditioned on the past frames [5]. The deep contextualized acoustic representation can be learned using the features that are constructed from the masked input speech frames [9–13]. The contrastive predictive coding (CPC) [2] and wav2vec [6] extracts the representation from data by utilizing the next step prediction and performs different ASR tasks. It was shown in [14, 15] that bi-directional and modified CPC transfers well across domains and languages. The vq-wav2vec approach discretizes the input speech to a quantized latent space [7]. The wav2vec 2.0 model masks the input speech in the latent space and solves a contrastive task defined over a quantization of the latent representations [8].

Among various speech representation learning frameworks, wav2vec 2.0 obtains the best performance [8], e.g., by only using 10 minutes of transcribed speech, it achieves a word error rate (WER) of 5.7/10.1 on the clean/noisy test sets of LibriSpeech [16]. In [17], it was further extended to a multilingual setting and the phoneme error rate can also be reduced significantly. However, the wav2vec 2.0 model is based on a transformer [18], which is effective for modeling the long-term global context, but less capable of extracting fine-grained local feature patterns [19]. As acoustic events often happen in the short-term sense, both long-term global context and fine-grained local dependency are necessary. For this, in [20] a contextNet method was proposed, which uses the convolution and squeeze-and-excitation modules [21] to capture the long-term context. In [19], a conformer was proposed by combining the convolution and transformer to model both local and global dependencies of speech sequences, and can thus improve the ASR performance. The lite transformer splits the input into a self-attention module and a convolution module, and concatenates the corresponding outputs [22]. Indeed, the convolution captures the local context and the attention reveals the global context, respectively. It was shown that the lite transformer can obtain a performance gain for machine translation in mobile applications [22]. Furthermore, the convBERT model [23] is based on the use of a span-based dynamic convolution, which is involved to replace redundant self-attention heads and directly model local dependencies in the BERT [24] model. Compared to BERT, convBERT can thus improve the performance in natural language processing (NLP) tasks, and more importantly requires less pre-training computations.

In general, there are two methods that can enhance the local dependency modeling in the transformer model: 1) convolution-augmented transformer [19], where the self-attention layer is followed by a convolutional layer, sandwiched between a pair of feed forward modules; and 2) local attention-augmented transformer [22], where the self-attention and convolutional layers

This work is supported by the Fundamental Research Funds for the Central Universities (WK2100000016), National Key R&D Program of China (2017YFB1002202) and the Leading Plan of CAS (XDC08010200).

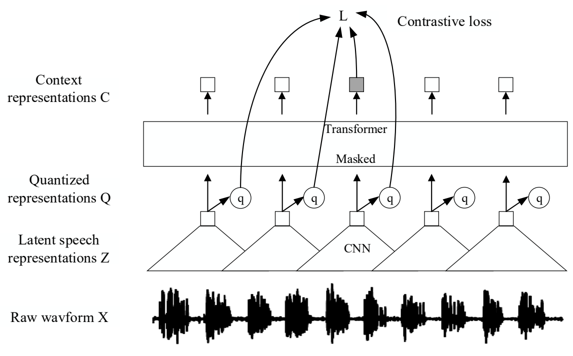
L
Contrastive loss
Context representations C
Transformer
Masked
Quantized representations Q
q
Latent speech representations Z
CNN
Raw wavform X

Figure 1: *An illustration of the wav2vec 2.0 model.*

are in parallel. Due to the fact that convolution captures the local context and attention reflects the global context to some extent, it was shown that enhancing local dependency enables a performance gain for many tasks beyond speech, which, however, is still an open research topic in the context of ASR.

In this work, we therefore investigate whether enhancing local dependency modeling can improve the pre-training efficiency and the transfer performance of wav2vec 2.0. We pre-train ASR models on the unlabeled 960h LibriSpeech data and follow the Libri-light [25] limited resource supervised training sets of 100/10/1 hours. For the local dependency enhancement, we propose four encoders: 1) conformer encoder (similar to the conformer in [19]); 2) transformer encoder 1 (similar to the lite transformer in [22]); 3) two combinations of the convolution-augmented and local attention-augmented transformers (i.e., transformer encoder 2 and transformer encoder 3). By increasing the local dependency, the average WER is decreased by 16.1% relatively on the standard LibriSpeech test set compared to the reproduced wav2vec 2.0 model at the cost of adding 5M parameters. In addition, the pre-training phase converges faster and the pre-trained models can be more easily transferred to the subsequent ASR task with fewer fine-tune epochs. For example, the proposed model only requires 200 epochs to reach the best performance of wav2vec 2.0.

2. Methodology

2.1. Wav2vec 2.0

In order to guide the reader, we briefly review the wav2vec 2.0 model in this section, which is shown in Figure 1, consisting of a CNN-based feature encoder $f : X \mapsto Z$ and a transformer encoder $g : Z \mapsto C$. Specifically, the feature encoder downsamples the input raw waveform X to the latent speech representation Z. The transformer encoder then models the contextualized representation C and captures high-level content from the input Z. The output of the feature encoder is discretized to q_t with a quantization module $Z \mapsto Q$ as targets in the contrastive objective. The quantization module first maps the latent speech representation Z to logits $\mathbf{l} \in \mathbb{R}^{G\times V}$, given G codebooks with V entries. The Gumbel Softmax operation [26] is then used to select discrete codebook entries in a fully differentiable way. As a result, for a given frame Z_t, we can select one entry from each codebook and concatenate the resulting vectors $e_1, ..., e_G$ and apply a linear transformation to obtain $\mathbf{q}_t$. The loss function can therefore be given by

$$L = L_m + \alpha L_d + \beta L_f, \tag{1}$$

where

$$L_m = -\log \frac{\exp(\text{sim}(\mathbf{c}_t, \mathbf{q}_t)/\kappa)}{\sum_{\tilde{\mathbf{q}}\sim\mathbf{Q}_t} \exp(\text{sim}(\mathbf{c}_t, \tilde{\mathbf{q}})/\kappa)}, \tag{2}$$

$$L_d = \frac{1}{GV} \sum_{g=1}^{G} \sum_{v=1}^{V} \overline{p}_{g,v} \log \overline{p}_{g,v}, \tag{3}$$

$$\overline{p}_{g,v} = \frac{\exp(\overline{l}_{g,v} + n_v)/\tau}{\sum_{k=1}^{V} \exp(\overline{l}_{g,k} + n_k)/\tau}. \tag{4}$$

It is clear that the total loss function is the weighted summation over three terms depending on the parameters α and β. In (2), L_m is the contrastive loss, which enables the model distinguishable between the true quantized latent speech representation $\mathbf{q}_t$ and a set of $K+1$ quantized candidate representations $\tilde{\mathbf{q}} \in \mathbf{Q}_t$. The quantized candidate representation $\tilde{\mathbf{q}}$ contains $\mathbf{q}_t$ and K distractors, and the latter are uniformly sampled from other masked time steps of the same utterance. In (1), the diversity loss L_d aims to increase the use of quantized codebook representation, and L_f is an ℓ_2 penalty over the outputs of the feature encoder. In (2), sim stands for the cosine similarity between two vectors and κ is a temperature. In (3), $\overline{p}_{g,v}$ represents the probability of choosing the v-th codebook entry for group g across a batch of utterances, where τ is a temperature. In (4), $\overline{l}_{g,v}$ stands for the average logits $\mathbf{l}$ across utterances in a batch. For more details on the wav2vec 2.0 model, we refer to [8].

2.2. The proposed enhanced local dependency encoder

Based on the relationship between the convolution module and the self-attention module, we propose four local dependency encoders in this section, including the conformer encoder, transformer encoder 1, transformer encoder 2, and transformer encoder 3, which are shown in Figure 2. The differences among these encoders are marked with black dashed blocks in the figure. The conformer encoder cascades the self-attention module and the convolutional module, sandwiched by a pair of half-step feed forward modules. The two feed forward modules are independent in the original conformer [19]. Slightly different from that, in order to compare with wav2vec 2.0, the parameters in two feed forward modules are shared in the proposed encoders, i.e., without additional parameters. The proposed transformer encoder 1 parallels the self-attention module and the convolution module. The left and right branches capture local and global contexts, respectively, and their outputs are simply summed. This is different from the lite transformer in [22], where instead of feeding the whole input to both branches, it is split into two parts along the channel dimension, which will be mixed by the following feed forward module.

In order to further enhance the local dependency modeling, in transformer encoder 2, the convolution module and self-attention module are first paralleled, and the corresponding outputs are summed and then input to another convolution module. In transformer encoder 3, the self-attention module is followed by a convolution module, and then paralleled with another convolution module. Note that in the proposed transformer encoders 2 and 3, two convolution modules are independent with different parameters. For all encoders, the convolution module follow the structure as in [19]. The convolution module contains a pointwise convolution and a gated linear (GLU), followed by a 1-dimensional (1-D) depthwise convolution layer, which is followed by a batchnorm and then a swish activation layer. In the proposed transformer encoders 2 and 3, since there are two convolution modules, in order to keep the amount of the mod-

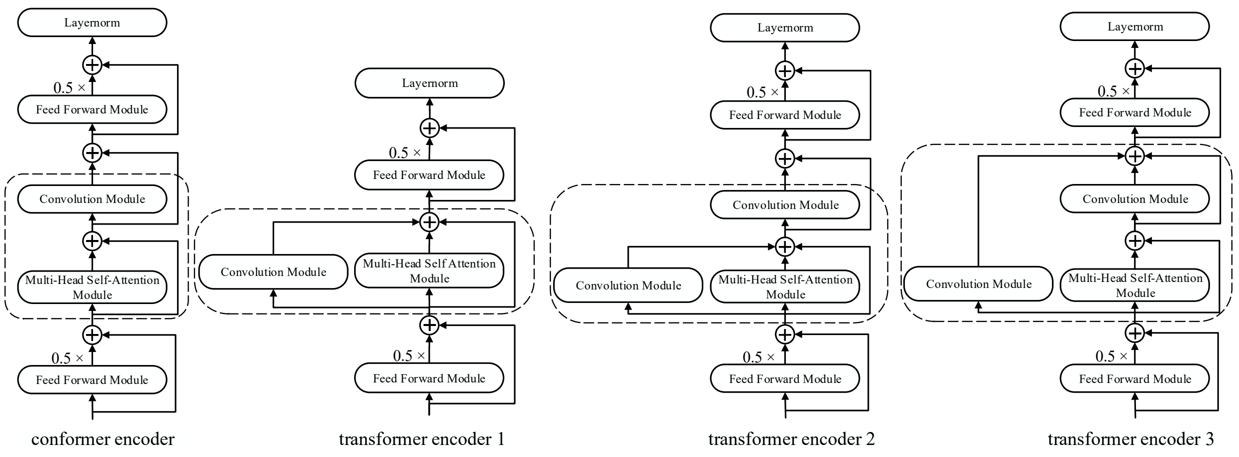

Figure 2: *The proposed enhanced local dependency encoders with different combinations of the self-attention and convolution modules.*

el parameters unchanged, the dimension of the 1-D depthwise convolution layer in the convolution module is halved.

3. Experimental Setup

3.1. Data

In this work, all experiments are conducted on the LibriSpeech corpus without transcriptions containing 100 hours (LS-100) or 960 hours (LS-960) audio as unlabeled data. In detail, LS-100 contains train-clean-100 subset, and LS-960 contains train-clean-100, train-clean-360 and train-other-500 subsets. The labeled data contains train-clean-100, train-10h (10 hours labeled), train-1h (1 hour labeled) subsets from LibriSpeech and Libri-light corpora. In case of using LS-100 speech data as unlabeled data for pre-training, we fine-tune on 10 hours of labeled data to compare the performance under different structures. In case of using LS-960 speech data as unlabeled data for pre-training, we fine-tune on 1 hour, 10 hours, and 100 hours of labeled data, respectively, and the results are compared with wav2vec 2.0 on the standard LibriSpeech dev-clean/other and test-clean/other sets.

3.2. Pre-training

In this work, the ASR models are implemented using the fairseq toolkit [27]. The feature encoder contains seven blocks, where each block has 512 temporal convolution channels with strides (5,2,2,2,2,2,2) and kernel widths (10,3,3,3,3,2,2). Thus, the interval between two sequential samples in the feature encoder output Z is around 20 ms and the receptive audio field is 25 ms.

The models contain 12 transformer encoder blocks with a dimension of 512, a feed forward module with a dimension of 2048 and 8 attention heads when pre-training on LS-100. In case of pre-training on LS-960, the model dimension is 768, the dimension of the feed forward module is 3072, and 8 attention heads are exploited. The depthwise convolution kernel size of the convolution module is 32, and the depthwise convolution dimension is set to be 256. The size of total parameters of the convolution module are 5M. For masking, we sample at a probability of p = 0.065 at all time steps and mask the subsequent M = 10 time steps. The pre-training process is optimized with Adam [28]. During the first 8% of the updates, the learning rate warms up to 5×10^{-4} and then decays linearly. We set G = 2 and V = 320 for the quantization module and each entry with a size of 128. The temperature κ is set to be 0.1 and τ is annealed from 2 to 0.5 by a factor of 0.999995 over iterations. For the contrastive loss, α and β are set to be 0.1 and 10, respectively. We use K = 100 distractors and the total number of pre-training updates is 400k, which keeps the same for wav2vec 2.0.

3.3. Fine-tuning

After pre-training, we add a randomly initialized linear projection layer on the top of the transformer encoder and then fine-tune the learned representation on the labeled data. The representation is classfied into 30 categories, which contain 29 character tokens and a word boundary token. The models are optimized by minimizing a CTC loss [29]. We utilize three random seeds to fine-tune the pre-trained models on all subsets without any language models, and calculate the average WER.

For the original wav2vec 2.0, the feature encoder is not trained during fine-tuning. Only the output classification layer is trained during the first 10k updates, and then the transformer encoder is updated. Slightly different from that, we only fix the feature encoder when fine-tuning, and the transformer encoder and randomly initialized layers are trained altogether. Other configurations keep the same as in [8].

4. Results

4.1. Pre-training on the 100 hours unlabeled data

As the pre-training of wav2vec 2.0 is very time-consuming, we perform pre-training on the 100 hours unlabeled data and fine-tune on the 10 hours labeled data. The results of the considered enhanced local dependency encoders are presented in Table 1. The conformer encoder and transformer encoder 1 obtain a similar performance, and achieve a performance improvement compared to the baseline. Compared to the conformer encoder, the proposed transformer encoders 2 and 3 can further improve the performance, and the transformer encoder 2 obtains the best. For comparison brevity, we will omit the proposed transformer encoders 1 and 3 in the sequel.

4.2. Pre-training on the 960 hours unlabeled data

In order to observe the effect of the data amount on the pre-training performance, we further pre-train on the 960 hours un-

Table 1: *The WERs using the LibriSpeech test-clean/test-other sets with pre-training on the 100 hours unlabeled data and fine-tuning on the 10 hours labeled data.*

model	model size	test	
		clean	**other**
10h labeled			
baseline	45M	20.4	37.9
conformer encoder	50M	19.3	36.3
transformer encoder 1	50M	19.2	36.2
transformer encoder 2	50M	17.0	33.4
transformer encoder 3	50M	17.8	34.5

Table 2: *The WERs using the LibriSpeech test/dev sets with pre-training on the 960 hours unlabeled data and fine-tuning on different labeled dataset.*

model	model size	test		dev	
		clean	**other**	**clean**	**other**
1h labeled					
wav2vec 2.0 [8]	95M	24.5	29.7	24.1	29.6
baseline (reproduced)	95M	19.3	26.6	18.9	26.2
conformer encoder	100M	18.7	24.0	17.4	23.4
transformer encoder 2	100M	16.7	22.2	16.5	22.0
10h labeled					
wav2vec 2.0 [8]	95M	11.1	17.6	10.9	17.4
baseline (reproduced)	95M	9.9	17.6	9.8	17.4
conformer encoder	100M	9.0	15.5	8.9	15.3
transformer encoder 2	100M	8.4	14.6	8.4	14.4
100h labeled					
wav2vec 2.0 [8]	95M	6.1	13.3	6.1	13.5
baseline (reproduced)	95M	5.6	13.0	5.5	13.5
conformer encoder	100M	5.0	11.4	4.9	11.7
transformer encoder 2	100M	4.7	10.5	4.6	10.7

labeled data. The WERs of the baseline, conformer encoder, and the proposed transformer encoder 2 are shown in Table 2. The baseline model is reproduced following the configuration in the original wav2vec 2.0 model. From Table 2, we find that the conformer encoder (the transformer encoder 2) obtain a performance improvement of around 10% (16%) relatively compared to the baseline. This implies that enhancing local dependency in wav2vec 2.0 can improve the ASR performance. It is worth noting that for the reproduction of wav2vec 2.0, we fine-tune on the labeled data, and the transformer encoder and the randomly initialized output layer are jointly trained, which improves the performance of the original wav2vec 2.0 model as shown in [8].

As the pre-trained model is obtained every 100 epochs and then fine-tuned on the 100 hours labeled data, we further show the WER in terms of pre-training epochs in Figure 3. Clearly, the models using the enhanced local dependency encoders (e.g., conformer encoder, transformer encoder 2) converge faster than the baseline at the pre-training stage, and can achieve the optimal performance of wav2vec 2.0 after 200 epochs. This reveals that enhancing the local dependency can improve the pre-training efficiency without sacrificing the ASR accuracy.

In order to quantify the capability of being transferred to downstream ASR tasks for the pre-trained models with enhanced local dependency, we utilize the conicity metric proposed in [30] to analyze the divergence in the learned representations of different time frames. To do so, we first calculate the alignment to mean (ATM) for each vector $\mathbf{v}_i$ con-

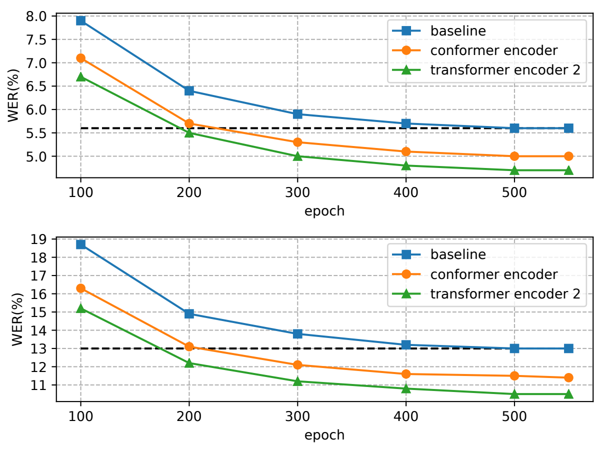

Figure 3: *The WERs obtained using the test-clean set (top) and the test-other set (bottom) in terms of epochs.*

Table 3: *The average conicity in the LibriSpeech dev-clean set.*

model	dev-clean conicity	
	before fine-tuning	**after fine-tuning**
baseline (reproduced)	0.9052	0.5418
conformer encoder	0.6594	0.5063
transformer encoder 2	0.6921	0.5313

tained in the vector set $\mathbf{V} = \{\mathbf{v}_1, ..., \mathbf{v}_m\}$ as $\text{ATM}(\mathbf{v}_i, \mathbf{V}) = \cos(\mathbf{v}_i, \frac{1}{m}\sum_{i=1}^{m}\mathbf{v}_j)$. Then, we calculate the mean of the ATMs for $\mathbf{v}_i \in \mathbf{V}$ as conicity, i.e., $\text{conicity}(\mathbf{V}) = \frac{1}{m}\sum_{i=1}^{m}\text{ATM}(\mathbf{v}_i, \mathbf{V})$. It is clear that the higher the conicity, the more closely the vectors in $\mathbf{V}$ are aligned with respect to the mean. Table 3 shows the average conicity of the pre-trained models using the Librispeech dev-clean set before and after fine-tuning. We can see that the representation learned by the enhanced local dependency encoders (conformer encoder, transformer encoder 2) have a much lower conicity before fine-tuning compared to the baseline, indicating that the proposed enhanced local dependency encoders can capture more fine-grained information. Due to the fact that the fine-grained information helps models quickly capture the ASR-related information, we can therefore conclude that the pre-trained models using the proposed enhanced local dependency encoders are more effective to be transferred to downstream ASR tasks.

5. Conclusions

In this paper, we investigated whether enhancing local dependency can improve the pre-training efficiency and the transfer capability of wav2vec 2.0. For this, we proposed four enhanced local dependency encoders, which depends on the combination of the convolution and self-attention modules. It was shown that the proposed transformer encoder 2, which parallelizes a convolution module and a self-attention mudule followed by another convolution module, achieves the best performance among different combinations. The proposed method can improve the pre-training efficiency and transfer performance of wav2vec 2.0 at the cost of adding a few more parameters. In addition, it was shown that the proposed pre-trained models can be more effectively transferred to downstream ASR tasks.

6. References

[1] M. P. Lewis, G. F. Simon, and C. D. Fennig, "Ethnologue: languages of the world, nineteenth edition," *http://www. ethnologue. com/*, 2016.

[2] A. van den Oord, Y. Li, and O. Vinyals, "Representation learning with contrastive predictive coding," *arXiv preprint arXiv:1807.03748*, 2018.

[3] J. Chorowski, R. J. Weiss, S. Bengio, and A. van den Oord, "Unsupervised speech representation learning using wavenet autoencoders," *IEEE/ACM Tran. Audio, Speech, Language Process.*, vol. 27, no. 12, pp. 2041–2053, 2019.

[4] Y.-A. Chung and J. Glass, "Speech2vec: A sequence-to-sequence framework for learning word embeddings from speech," in *Proc. Interspeech*, 2018, pp. 811–815.

[5] Y.-A. Chung, W.-N. Hsu, H. Tang, and J. Glass, "An unsupervised autoregressive model for speech representation learning," in *Proc. Interspeech*, 2019, pp. 146–150.

[6] S. Schneider, A. Baevski, R. Collobert, and M. Auli, "Wav2vec: unsupervised pre-training for speech recognition," in *Proc. Interspeech*, 2019, pp. 3465–3469.

[7] A. Baevski, S. Schneider, and M. Auli, "vq-wav2vec: Self-supervised learning of discrete speech representations," *arXiv preprint arXiv:1910.05453*, 2019.

[8] A. Baevski, Y. Zhou, A. Mohamed, and M. Auli, "Wav2vec 2.0: A framework for self-supervised learning of speech representations," in *Advances in Neural Information Processing Systems*, vol. 33. Curran Associates, Inc., 2020, pp. 12 449–12 460.

[9] A. T. Liu, S.-W. Yang, P.-H. Chi, P.-C. Hsu, and H.-Y. Lee, "Mockingjay: Unsupervised speech representation learning with deep bidirectional transformer encoders," in *Proc. IEEE Int. Conf. Acoustics, Speech and Signal Processing (ICASSP)*, 2020, pp. 6419–6423.

[10] S. Ling, Y. Liu, J. Salazar, and K. Kirchhoff, "Deep contextualized acoustic representations for semi-supervised speech recognition," in *Proc. IEEE Int. Conf. Acoustics, Speech and Signal Processing (ICASSP)*, 2020, pp. 6429–6433.

[11] A. T. Liu, S.-W. Li, and H.-y. Lee, "Tera: Self-supervised learning of transformer encoder representation for speech," *arXiv preprint arXiv:2007.06028*, 2020.

[12] W. Wang, Q. Tang, and K. Livescu, "Unsupervised pre-training of bidirectional speech encoders via masked reconstruction," in *Proc. IEEE Int. Conf. Acoustics, Speech and Signal Processing (ICASSP)*, 2020, pp. 6889–6893.

[13] D. Jiang, X. Lei, W. Li, N. Luo, Y. Hu, W. Zou, and X. Li, "Improving transformer-based speech recognition using unsupervised pre-training," *arXiv preprint arXiv:1910.09932*, 2019.

[14] K. Kawakami, L. Wang, C. Dyer, P. Blunsom, and A. van den Oord, "Learning robust and multilingual speech representations," in *Proc. Int. Conf. Empirical Methods in Natural Language Processing: Findings*, 2020, pp. 1182–1192.

[15] M. Rivière, A. Joulin, P.-E. Mazaré, and E. Dupoux, "Unsupervised pretraining transfers well across languages," in *Proc. IEEE Int. Conf. Acoustics, Speech and Signal Processing (ICASSP)*, 2020, pp. 7414–7418.

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

[17] A. Conneau, A. Baevski, R. Collobert, A. Mohamed, and M. Auli, "Unsupervised cross-lingual representation learning for speech recognition," *arXiv preprint arXiv:2006.13979*, 2020.

[18] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," *arXiv preprint arXiv:1706.03762*, 2017.

[19] A. Gulati, J. Qin, C.-C. Chiu, N. Parmar, Y. Zhang, J. Yu, W. Han, S. Wang, Z. Zhang, Y. Wu, and R. Pang, "Conformer: convolution-augmented transformer for speech recognition," in *Proc. Interspeech*, 2020, pp. 5036–5040.

[20] W. Han, Z. Zhang, Y. Zhang, J. Yu, C.-C. Chiu, J. Qin, A. Gulati, R. Pang, and Y. Wu, "ContextNet: improving convolutional neural networks for automatic speech recognition with global context," in *Proc. Interspeech*, 2020, pp. 3610–3614.

[21] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 7132–7141.

[22] Z. Wu, Z. Liu, J. Lin, Y. Lin, and S. Han, "Lite transformer with long-short range attention," *arXiv preprint arXiv:2004.11886*, 2020.

[23] Z. Jiang, W. Yu, D. Zhou, Y. Chen, J. Feng, and S. Yan, "Convbert: Improving bert with span-based dynamic convolution," *arXiv preprint arXiv:2008.02496*, 2020.

[24] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[25] J. Kahn, M. Rivière, W. Zheng, E. Kharitonov, Q. Xu, P.-E. Mazaré, J. Karadayi, V. Liptchinsky, R. Collobert, C. Fuegen *et al.*, "Libri-light: A benchmark for asr with limited or no supervision," in *Proc. IEEE Int. Conf. Acoustics, Speech and Signal Processing (ICASSP)*, 2020, pp. 7669–7673.

[26] E. Jang, S. Gu, and B. Poole, "Categorical reparameterization with gumbel-softmax," *arXiv preprint arXiv:1611.01144*, 2016.

[27] M. Ott, S. Edunov, A. Baevski, A. Fan, S. Gross, N. Ng, D. Grangier, and M. Auli, "Fairseq: A fast, extensible toolkit for sequence modeling," in *Proc. Conf. the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, 2019, pp. 48–53.

[28] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[29] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, "Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks," in *Proc. Int. Conf. Machine Learning (ICML)*, 2006, pp. 369–376.

[30] A. Sharma, P. Talukdar *et al.*, "Towards understanding the geometry of knowledge graph embeddings," in *Proc. the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2018, pp. 122–131.