



W2V2-Light: A Lightweight Version of Wav2vec 2.0 for Automatic Speech Recognition

Dong-Hyun Kim*, Jae-Hong Lee*, Ji-Hwan Mo, Joon-Hyuk Chang

Department of Electronics Engineering
Hanyang University, Seoul, Republic of Korea

{dlstjtd1477, ljh931jh, mojihwan, jchang}@hanyang.ac.kr

Abstract

Wav2vec 2.0 (W2V2) has shown remarkable speech recognition performance by pre-training only with unlabeled data and fine-tuning with a small amount of labeled data. However, the practical application of W2V2 is hindered by hardware memory limitations, as it contains 317 million parameters. To address this issue, we propose W2V2-Light, a lightweight version of W2V2. We introduce two simple sharing methods to reduce the memory consumption as well as the computational costs of W2V2. Compared to W2V2, our model has 91% lesser parameters and a speedup of 1.31 times with minor degradation in downstream task performance. Moreover, by quantifying the stability of representations, we provide an empirical insight into why our model is capable of maintaining competitive performance despite the significant reduction in memory

Index Terms: Automatic speech recognition, semi-supervised learning, representation learning, parameter sharing, attention alignment

1. Introduction

Existing automatic speech recognition (ASR) systems require a large amount of transcribed speech data to attain good performance. However, while speech-only data can easily be acquired, obtaining paired transcriptions is often demanding. In order to use massive untranscribed speech data in ASR systems, a lot of research has been conducted, such as semi-supervised learning (SSL) models [1, 2]. In particular, representation learning methods have been in the spotlight recently, due to its impressive performance on downstream tasks. [3, 4]. Especially, Wav2vec 2.0 (W2V2) has shown state-of-the-art performance on speech recognition tasks [5, 6]. This model encodes raw audio via convolution module, generates context representations through Transformer module, and outputs transcriptions from the classifier. In this way, W2V2 achieved remarkable performance on downstream tasks with only limited labeled data [5].

Nonetheless, hardware memory limitations impede the application of W2V2 as it contains 317 million parameters, much larger than widely used ASR models [7, 8]. There have been numerous studies to reduce the size of a deep neural network, such as knowledge distillation and network pruning [9, 10]. Unfortunately, these methods yield unsatisfactory reduction in model size and noticeable degradation in performance, when applied to W2V2 [11]. The methods also do not consider the structural characteristics of W2V2 that the Transformer module occupies 95% (302M) of the model size. On the contrary, we focus on compressing the size of Transformer module in W2V2.

In recent years, several studies have been proposed to reduce the memory usage of Transformer modules in the fields

of natural language processing (NLP) [12, 13]. In particular, Transformer parameter sharing method has received much attention due to its outstanding performance and ease of implementation [14–16]. However, to the best of our knowledge, there have been no studies applying the method to W2V2.

In this study, we present W2V2-Light that integrates two simple sharing schemes to reduce the memory consumption and computational costs respectively. Specifically, we introduce a layer-wise parameter sharing (LPS) method to reduce the memory consumption of our model mainly occupied by the Transformer module. We also propose an attention alignment sharing (AAS) technique to increase the training and inference speed. Compared to W2V2, our W2V2-Light model achieves 91% fewer parameters with a slight degradation of absolute 0.3%/2.5% WER on LibriSpeech clean/other validation sets [17]. The additional introduction of AAS results in a speedup of 1.31 times along with further degradation of absolute 0.2%/1.4% WER. Moreover, we provide an empirical insight into why our model is capable of preserving impressive performance by quantifying the stability of representations.

2. Methods

In order to lighten W2V2, we employ two straightforward sharing methods. First, we reduce the memory consumption of the model by our LPS scheme (Section 2.3). Second, we lower the computational cost via the proposed AAS strategy (Section 2.4). We incorporate the aforementioned methods to build W2V2-Light model. Our entire model architecture is illustrated in Fig. 1.

2.1. Baseline model

As our baseline model, we use W2V2 which is composed of a multi-layer convolutional audio encoder, a quantizer, and Transformer. The convolutional audio encoder $f : \mathcal{X} \mapsto \mathcal{Z}$ takes a raw waveform \mathcal{X} as input and outputs a latent speech representation $\mathbf{z} = [\mathbf{z}_1, \dots, \mathbf{z}_T]^T$, where T is the length of input sequence. The latent representation is then fed to both quantizer module $g : \mathcal{Z} \mapsto \mathcal{Q}$ and Transformer module $h : \mathcal{Z} \mapsto \mathcal{C}$ simultaneously to generate a quantized representation $\mathbf{q} = [\mathbf{q}_1, \dots, \mathbf{q}_T]^T$ and a context representation $\mathbf{c} = [\mathbf{c}_1, \dots, \mathbf{c}_T]^T$ [5]. The quantizer module g chooses one of the V entries $e \in \mathbb{R}^{V \times d/G}$ from each of G codebooks using Gumbel softmax [18] and concatenates the vectors e_1, \dots, e_G to obtain $\mathbf{q} \in \mathbb{R}^f$ through a linear transformation $\mathbb{R}^d \mapsto \mathbb{R}^f$.

2.2. Loss function

W2V2 is pre-trained in an unsupervised manner using contrastive loss L_m , augmented by a codebook diversity loss L_d .

*Equal Contributions

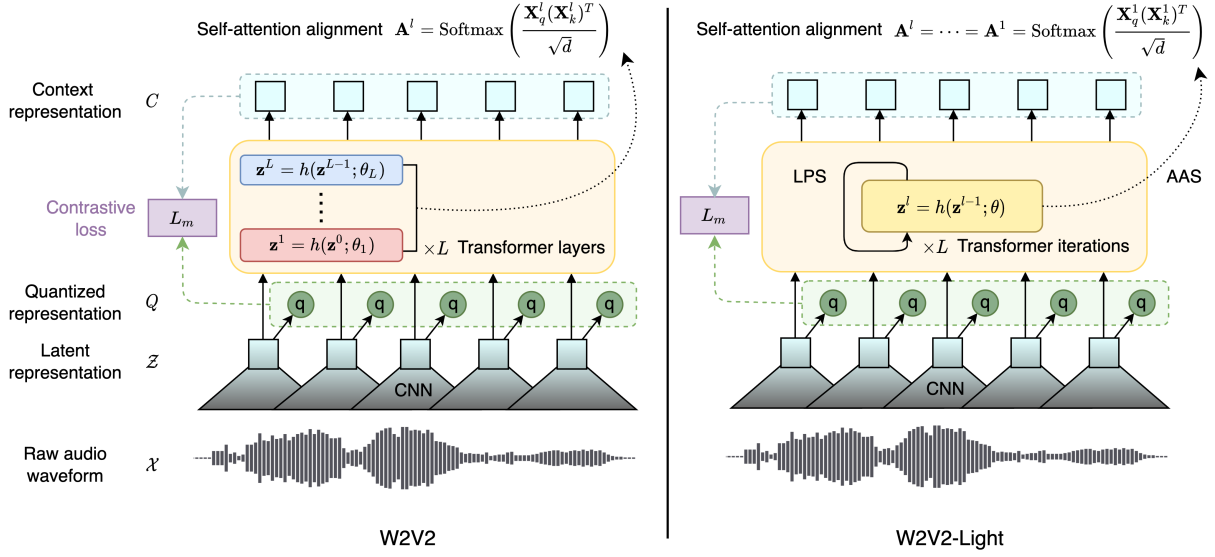


Figure 1: Architectures of W2V2 model (left) and W2V2-Light model (right)

$$\mathcal{L} = \mathcal{L}_m + \alpha \mathcal{L}_d$$

where α is a hyperparameter. The contrastive loss L_m is designed to identify a quantized latent representation \mathbf{q}_t from quantized vectors $\tilde{\mathbf{q}}$ in \mathcal{Q}_t to predict a context representation \mathbf{c}_t , as follows:

$$\mathcal{L}_m = -\log \frac{\exp(\text{sim}(\mathbf{c}_t, \mathbf{q}_t)/\kappa)}{\sum_{\tilde{\mathbf{q}} \in \mathcal{Q}_t} \exp(\text{sim}(\mathbf{c}_t, \tilde{\mathbf{q}})/\kappa)}$$

where κ is a temperature hyperparameter that affects the sharpness of loss function and $\text{sim}(\mathbf{c}_t, \mathbf{q}_t)$ is cosine similarity between \mathbf{c}_t and \mathbf{q}_t .

$$\text{sim}(\mathbf{a}, \mathbf{b}) = \mathbf{a}^T \mathbf{b} / \|\mathbf{a}\| \|\mathbf{b}\|$$

The diversity loss L_d induces the model to maximize the entropy of the averaged softmax distribution for each codebook \bar{p}_g , resulting in the equal use of codebook entries.

$$\mathcal{L}_d = \frac{1}{GV} \sum_{g=1}^G -H(\bar{p}_g) = \frac{1}{GV} \sum_{g=1}^G \sum_{v=1}^V \bar{p}_{g,v} \log \bar{p}_{g,v},$$

where G is the number of codebooks and V is the number of codewords in each codebook.

2.3. Layer-wise Parameter Sharing

In our proposed LPS method, the same parameters are shared throughout the entire Transformer layers. Basically, a multi-layer Transformer in W2V2 obtains a latent representation by $\mathbf{z}^l = h(\mathbf{z}^{l-1}; \theta^l)$, where L is the depth of Transformer, and $l = \{1, \dots, L\}$ is the layer index. Also, $h(\cdot)$ denotes the Transformer layer and θ^l stands for the l -th parameter set. The first Transformer layer input \mathbf{z}^0 is a masked representation in which the mask is applied to the latent representation \mathbf{z} obtained from the convolutional audio encoder.

Our model, on the other hand, calculates the representation using LPS as follows:

$$\mathbf{z}^l = h(\mathbf{z}^{l-1}; \theta), \text{ where } \theta^L = \dots = \theta^1 = \theta,$$

where we gain two advantages: model size reduction and stability of the representation. Our LPS technique reduces the Transformer module size by a factor of L as the same parameters are shared all over the Transformer layers. It also enables to obtain stable representations that lead to better performance on downstream tasks (See Section 4 for more details).

2.4. Attention Alignment Sharing

W2V2 computes self-attention by the matrix multiplication of an attention alignment \mathbf{A} and a value $\mathbf{X}_v \in \mathbb{R}^{T \times d}$, where d is the dimension of representations [21, 22]. The attention alignment at l -th layer \mathbf{A}^l is calculated by the scaled dot-product of the query $\mathbf{X}_q^l \in \mathbb{R}^{T \times d}$ and the key $\mathbf{X}_k^l \in \mathbb{R}^{T \times d}$, as follows:

$$\mathbf{A}^l = \text{Softmax} \left(\frac{\mathbf{X}_q^l (\mathbf{X}_k^l)^\top}{\sqrt{d}} \right). \quad (1)$$

Note that queries, keys, and values are $\mathbf{X}_q^l = \mathbf{W}_q^\top \mathbf{z}_{l-1}$, $\mathbf{X}_k^l = \mathbf{W}_k^\top \mathbf{z}_{l-1}$, $\mathbf{X}_v^l = \mathbf{W}_v^\top \mathbf{z}_{l-1}$ respectively, where $\mathbf{W}_q, \mathbf{W}_k, \mathbf{W}_v \in \mathbb{R}^{d \times d}$ are linear projection matrices.

The fatal drawback of the attention mechanism is that the computational cost of the softmax attention, Eq. (1), scales with quadratic time complexity $O(T^2)$ [23, 24]. To mitigate this problem, we introduce an AAS strategy that pre-computes the attention alignment in the very first Transformer layer and reuses it throughout the whole Transformer layers, as follows:

$$\mathbf{A}^L = \dots = \mathbf{A}^1 = \text{Softmax} \left(\frac{\mathbf{X}_q^1 (\mathbf{X}_k^1)^\top}{\sqrt{d}} \right).$$

Here, the proposed AAS method possibly undermines each layer's diversity of attention alignments. Nevertheless, our model maintains comparable performance on downstream tasks with respect to other SSL models, as we describe in Section 3.2.

Table 1: WER comparison of our models and prior works on the LibriSpeech dev-clean/other and test-clean/other sets. All W2V2-based models are pre-trained on unlabeled LibriSpeech 960h and fine-tuned on labeled train-clean-100 (clean 100h subset of LibriSpeech). IPL and Noisy student were trained on LibriSpeech 100-860 (unlabeled 860h and labeled train-clean-100).

Model	Parameters	Speedup	LM	dev		test	
				clean	other	clean	other
IPL [19]	322M	-	4-gram+Transformer	5.0	8.0	5.6	9.0
Noisy student [20]	*540M	-	LSTM	3.9	8.8	4.2	8.6
W2V2	317M	1.00	4-gram	2.5	6.6	3.2	7.2
			Transformer	2.4	5.5	2.7	6.1
W2V2-Light (LPS)	28M	1.02x	4-gram	3.2	9.5	3.9	9.9
			Transformer	2.7	8.0	3.1	8.4
W2V2-Light (LPS+AAS)	28M	1.31x	4-gram	3.8	12.4	4.5	12.5
			Transformer	2.9	9.4	3.2	9.9

* Note that we approximated the number of Noisy student model parameters referring to [8]

3. Experiments

3.1. Experimental Setup

3.1.1. Dataset

To compare the proposed model with the existing W2V2 and other SSL models, our experiments followed W2V2 settings as much as possible. We pre-trained our model with unlabeled LibriSpeech (960h), and fine-tuned the model on labeled train-clean-100 subset (100h) of LibriSpeech. We evaluated our methods on speech recognition tasks using the standard LibriSpeech dev-clean/other and test-clean/other sets.

3.1.2. Pre-training

The multi-layer convolutional audio encoder in our model has 7 blocks, each with 512 channels, (5,2,2,2,2,2,2) strides, and (10,3,3,3,3,2,2) kernel sizes. The parameter shared 24-layer Transformer is comprised of self-attention layers with 1024 hidden units, feed-forward networks with 4096 dimensions, and 16 attention heads. We used the Adam optimizer [25] with a learning rate scheduled to increase to the maximum of $3e-3$ for the first 20% of updates and then linearly decays. We set total number of updates to 60k with max tokens of 1600k, and averaged gradients for every 32 batches to update our models. Each W2V2-Light (LPS) and W2V2-Light (LPS+AAS) was trained on 4 RTX 3090 GPUs for 15 and 11 days respectively.

3.1.3. Fine-tuning

To evaluate our pre-trained model on speech recognition tasks, we fine-tuned our model on labeled train-clean-100 subset of LibriSpeech. We first added a classification layer on the top of our pre-trained model to generate transcriptions. Subsequently, we trained our model using connectionist temporal classification loss [26]. We adopted the Adam optimizer with tri-stage learning rate scheduler, in which the learning rate warms up for the first 10% of fine-tuning phase, remains for 40%, and then

gradually decreases. To update the model parameters, an averaged gradient on 5 batches was employed. For the initial 10k updates, we only trained the classification layer and updated the Transformer after that. The convolutional audio encoder was not trained in fine-tuning step.

3.1.4. Decoding

For decoding, we considered two types of language models (LMs), 4-gram and Transformer respectively [27]. We decoded our model with pre-trained LMs provided by fairseq [28]. We used beam search decoder with the same decoding parameters described in [5].

3.2. Downstream Task Evaluation

We compared the performance and the number of parameters of our model with W2V2 and conventional SSL models to verify the efficiency of our methods. According to Table 1, our LPS method reduced 91% of the number of parameters from 317M to 28M, which is 11 times smaller than the iterative pseudo-labeling (IPL) model and 19 times smaller than the Noisy student model. On the other hand, WERs on each dataset increased: dev-clean (0.3%), dev-other (2.5%), test-clean (0.4%), and test-other (2.3%). Nevertheless, our model remained superior in comparison with the aforementioned existing SSL algorithms. The additional introduction of our AAS mechanism further degraded performances, but it increased both training and inference speed by 1.31 times.

3.3. LPS performance comparison according to the number of layers

To intensively investigate the influence of the number of layers L on model performance, we conducted experiments to analyze only the LPS-applied W2V2-Light model. Fig. 2 shows that WERs on dev/test sets decrease along with the increase in the number of layers. WERs drastically decline up to the 8-th

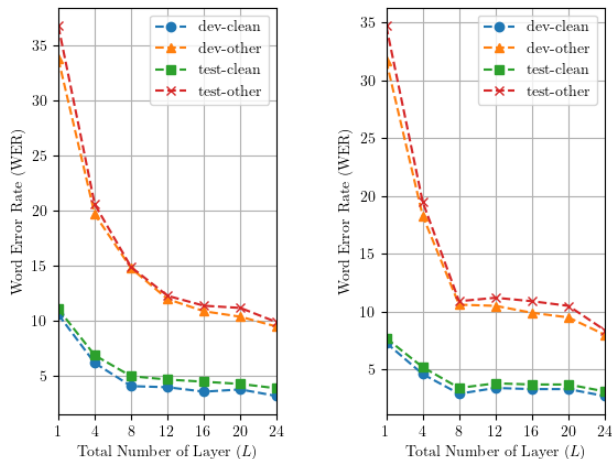


Figure 2: WERs on LibriSpeech dev-clean/other and test-clean/other sets according to the number of layers in W2V2-Light with 4-gram LM (left), with Transformer LM (right).

layer, fluctuate, then drop again after the 20-th layer. This phenomenon is especially noticeable in dev-other and test-other. From these results, we can deduce that our W2V2-Light model with a larger number of layers is more robust for speech recognition tasks in noisy environments. The most striking advantage of our model is that the size of the model remains the same regardless of the number of layers. We leave constructing large-layer W2V2-Light to future work.

4. Discussion

Although our model saves 10 times the memory compared to W2V2, the degradation in speech recognition performance is relatively small. In this section, we provide an empirical explanation for this phenomenon by quantifying the stability of representations.

Stability of representation. Transformer layers in our model, as shown in Fig. 3, produce comparatively stable representations rather than oscillating. It implies that each Transformer layer fulfills the same purpose of obtaining more contextualized representations. This argument is consistent with that of deep equilibrium models (DEQ) [14], which utilized the iterative structure of Transformer in the field of NLP. DEQ leveraged a representation corresponding to the equilibrium point no longer affected by the Transformer layer and achieved remarkable downstream task performance. Based on this, it can be inferred that the stability of representations is an important factor in determining the performance on downstream tasks.

Quantification of stability. We quantified the similarity between input and output representations from the layers of each of the original W2V2 model and our model in the same manner with ALBERT [15]. For the similarity functions, we adopted L2 distance and cosine similarity. As illustrated in Fig. 3, both W2V2 model and our model generate stable representations up to the intermediate layers. Contrarily, when it reaches the upper layers, the representations rapidly variate. In the case of W2V2, the L2 distance remains relatively stable, but the cosine similarity decreases sharply. It implies that while the absolute values of the input and output representations are not significantly different, the positions of the emphasized elements in the

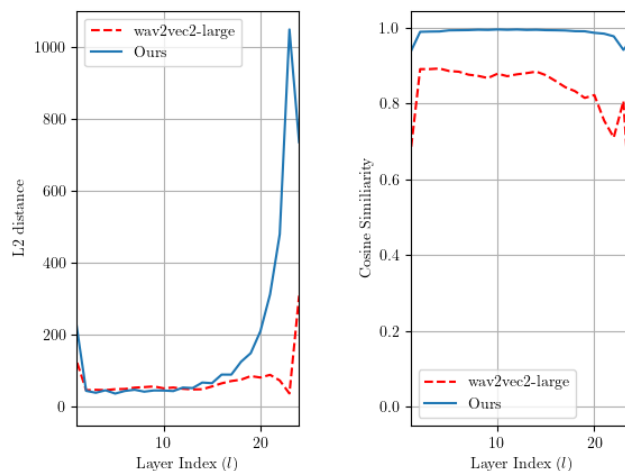


Figure 3: Comparison between input and output representations of each layer using L2 distance and cosine similarity. The total number of layers is 24.

representations differ tremendously. As for our model, on the other hand, it is vice-versa. At first glance, owing to the rapid increment in L2 distance in the upper layers, our model seems to lag behind W2V2 on downstream tasks. As a matter of fact, the representations in the intermediate layers are more linguistically informative than those in subsequent layers, which rather hinder the training [29]. It is because the W2V2 model is similar to the autoencoder in that it carries out the task of predicting masked representations. In other words, our model supplies adequate representations for performing downstream tasks since it accommodates stable representations in the intermediate Transformer layers.

5. Conclusion

In this paper, we have presented a lightweight version of Wav2vec 2.0, W2V2-Light. We shared Transformer parameters and attention alignments to reduce the memory consumption and computational cost simultaneously. We verified that our model could achieve superior performances against existing SSL models even with 91% fewer parameters and 1.31 times speedup. In addition, we investigated the validity of our methods in terms of the stability of representations. Our methods can be applied not only to W2V2, but also to other ASR models that contain the Transformer module. In future work, we plan to share parameters in sub-networks of the Transformer module, taking into account the nature of the representations at each model layer as examined in [29, 30]. Increasing the stability of representations at layers with similar properties may further develop our model.

6. Acknowledgements

This work was supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No.2021-0-00456, Development of Ultra-high Speech Quality Technology for Remote Multi-speaker Conference System)

7. References

- [1] A. v. d. Oord, Y. Li, and O. Vinyals, “Representation learning with contrastive predictive coding,” *arXiv:1807.03748*, 2018.
- [2] Q. Xu, T. Likhomanenko, J. Kahn, A. Hannun, G. Synnaeve, and R. Collobert, “Iterative pseudo-labeling for speech recognition,” *arXiv:2005.09267*, 2020.
- [3] S. Schneider, A. Baevski, R. Collobert, and M. Auli, “Wav2vec: Unsupervised pre-training for speech recognition,” *arXiv:1904.05862*, 2019.
- [4] A. Baevski, S. Schneider, and M. Auli, “VQ-wav2vec: Self-supervised learning of discrete speech representations,” *arXiv:1910.05453*, 2019.
- [5] A. Baevski, Y. Zhou, A. Mohamed, and M. Auli, “Wav2vec 2.0: A framework for self-supervised learning of speech representations,” in *Proc. Advances in Neural Information Processing Systems (NeurIPS)*, 2020.
- [6] Q. Xu, A. Baevski, T. Likhomanenko, P. Tomasello, A. Conneau, R. Collobert, G. Synnaeve, and M. Auli, “Self-training and pre-training are complementary for speech recognition,” *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 3030–3034, 2021.
- [7] D. Bahdanau, J. Chorowski, D. Serdyuk, P. Brakel, and Y. Bengio, “End-to-end attention-based large vocabulary speech recognition,” *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 4945–4949, 2016.
- [8] K. Irie, R. Prabhavalkar, A. Kannan, A. Bruguier, D. Rybach, and P. Nguyen, “On the choice of modeling unit for sequence-to-sequence speech recognition,” in *Proc. INTERSPEECH*, 2019, pp. 3800–3804.
- [9] G. Hinton, O. Vinyals, and J. Dean, “Distilling the knowledge in a neural network,” *arXiv:1503.02531*, 2015.
- [10] Y. LeCun, J. Denker, and S. Solla, “Optimal brain damage,” in *Proc. Advances in Neural Information Processing Systems (NeurIPS)*, 1989.
- [11] Z. Peng, A. Budhkar, I. Tuil, J. Levy, P. Sobhani, R. Cohen, and J. Nassour, “Shrinking bigfoot: Reducing Wav2vec 2.0 footprint,” *arXiv:2103.15760*, 2021.
- [12] J. Frankle and M. Carbin, “The lottery ticket hypothesis: Finding sparse, trainable neural networks,” *arXiv:1803.03635*, 2018.
- [13] M. Dehghani, S. Gouws, O. Vinyals, J. Uszkoreit, and L. Kaiser, “Universal transformers,” *arXiv:1807.03819*, 2018.
- [14] S. Bai, J. Z. Kolter, and V. Koltun, “Deep equilibrium models,” in *Proc. Advances in Neural Information Processing Systems (NeurIPS)*, 2019.
- [15] Z. Lan, M. Chen, S. Goodman, K. Gimpel, P. Sharma, and R. Soricut, “ALBERT: A lite BERT for self-supervised learning of language representations,” *arXiv:1909.11942*, 2020.
- [16] T. Xiao, Y. Li, J. Zhu, Z. Yu, and T. Liu, “Sharing attention weights for fast transformer,” *arXiv:1906.11024*, 2019.
- [17] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, “Librispeech: An ASR corpus based on public domain audio books,” *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 5206–5210, 2015.
- [18] E. Jang, S. S. Gu, and B. Poole, “Categorical reparameterization with gumbel-softmax,” *arXiv:1611.01144*, 2016.
- [19] Q. Xu, T. Likhomanenko, J. Kahn, A. Y. Hannun, G. Synnaeve, and R. Collobert, “Iterative pseudo-labeling for speech recognition,” *arXiv:2005.09267*, 2020.
- [20] D. S. Park, Y. Zhang, Y. Jia, W. Han, C.-C. Chiu, B. Li, Y. Wu, and Q. V. Le, “Improved noisy student training for automatic speech recognition,” in *Proc. INTERSPEECH*, 2020, pp. 2817–2821.
- [21] A. Vaswani, N. M. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, “Attention is all you need,” in *Proc. Advances in Neural Information Processing Systems (NeurIPS)*, 2017.
- [22] L. Dong, S. Xu, and B. Xu, “Speech-transformer: A no-recurrence sequence-to-sequence model for speech recognition,” *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 5884–5888, 2018.
- [23] A. Katharopoulos, A. Vyas, N. Pappas, and F. Fleuret, “Transformers are rnns: Fast autoregressive transformers with linear attention,” in *Proc. of International Conference on Machine Learning (ICML)*, 2020.
- [24] S. Wang, B. Z. Li, M. Khabsa, H. Fang, and H. Ma, “Linformer: Self-attention with linear complexity,” *arXiv:2006.04768*, 2020.
- [25] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” *arXiv:1412.6980*, 2014.
- [26] A. Graves, S. Fernández, F. J. Gomez, and J. Schmidhuber, “Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks,” in *Proc. of International Conference on Machine Learning (ICML)*, 2006.
- [27] A. Baevski and M. Auli, “Adaptive input representations for neural language modeling,” *arXiv:1809.10853*, 2018.
- [28] 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,” *arXiv:1904.01038*, 2019.
- [29] A. Pasad, J.-C. Chou, and K. Livescu, “Layer-wise analysis of a self-supervised speech representation model,” *Proc. IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, 2021.
- [30] L. Pepino, P. E. Riera, and L. Ferrer, “Emotion recognition from speech using Wav2vec 2.0 embeddings,” in *Proc. INTERSPEECH*, 2021, pp. 3400–3404.