OPTIMIZE WAV2VEC2S ARCHITECTURE FOR SMALL TRAINING SET THROUGH ANALYZING ITS PRE-TRAINED MODELS ATTENTION PATTERN

Liu Chen, Meysam Asgari

Oregon Health & Science University
Department of Pediatrics
Portland, Oregon, USA

Hiroko H Dodge

Oregon Health & Science University
Department of Neurology
Portland, Oregon, USA

ABSTRACT

Transformer-based automatic speech recognition (ASR) systems have shown their success in the presence of large datasets. But, in medical research, we have to create ASR for the non-typical population, i.e. pre-school children with speech disorders, with small training dataset. To increase training efficiency on small datasets, we optimize the architecture of Wav2Vec 2.0, a variation of Transformer, through analyzing its pre-trained model's block-level attention pattern. We show that block-level patterns can serve as an indicator for narrowing down the optimization direction. To ensure the reproducibility of our experiments, we leverage *Librispeech-100-clean* as training data to simulate the limited data condition. We leverage two techniques, local attention mechanism and cross-block parameter sharing, with counter-intuitive configurations. Our optimized architecture outperforms the vanilla architecture about 1.8% absolute word error rate (WER) on *dev-clean* and 1.4% on *test-clean*.

Index Terms— Transformer, automatic speech recognition, attention pattern, self-supervise learning, architecture optimization

1. INTRODUCTION

Transformers [1], a popular attention-based deep neural network (DNN) architecture in recent years, unleashes the possible model size to a whole new level [2]. A transformer contains multiple identical blocks. Each block contains multiple DNN layers including a multi-head self-attention (MSA) layer. With enough training data, deep transformers outperforms its relatively-shallower siblings [2, 3, 4] on mainstream tasks. These variations are designed to fully utilize a large dataset. These pre-trained models can serve the general population well. But, in medical research, we are facing special target populations, such as non-typical developed children or vocal injured seniors. Their vocal characteristics are different from the general population [5, 6, 7]. Moreover, collecting thousands of hours of data for them is also financially challenging. We are focusing on increasing Transformer-based ASR's training performance under a limited data condition so that the medical field can gain more benefit from speech recognition researches' achievements. We leverage a mature *model interpretation technique* to analyze a pre-trained acoustic model, Wav2vec 2.0 [4], and optimize its architecture based on the analysing results.

Recent studies on the attention pattern of BERT [8] indicate that we can gain valuable information by analysing the general attention patterns. Inspired by this research, we hypothesize that blocks' general attention patterns are input-irrelevant and the patterns of models, which are trained on large datasets, are close to the optimal pattern. We introduce inductive bias to the Transformer encouraging it to learn these patterns when the training daw ta is limited. We choose Wav2Vec 2.0 [4] as our research object.

Through summarizing block-level attention patterns of a pre-trained Wav2Vec 2.0 model, which has 12 attention blocks, we take two techniques to optimize its architecture: local attention and parameter sharing. Firstly, we apply local attention to the top 11 blocks and global attention to the bottom block. Secondly, we let the top 11 blocks share the same set of parameters while leaving the bottom block having its own parameters. We experimentally validate that each modification improves the training efficiency. The architecture with both modifications outperforms the vanilla architecture about 1.8% absolute word error rate (WER) on *dev-clean* and 1.4% on *test-clean*. Our modifications are counter-intuitive. Prior researches treat all attention blocks equally. For example, adopt one type of attention mechanism [9] and sharing parameter among all blocks [10]. Unlike them, we consider the bottom block different from the rest based on their block-level patterns. Our experiments show that this difference is the key to the improvement of training efficiency.

2. BACKGROUND

2.1. Wav2Vec2

Our target architecture is Wav2Vec 2.0 introduced by Baevski et al.[4]. Figure 1 shows its architecture in detail. It contains three main components: a convolutional feature encoder, a quantization block, and a Transformer. The feature en-

978-1-6654-0540-9/22/$31.00 ©2022 IEEE

coder extracts latent representation from raw audio input. The quantization module discretizes latent representation to a finite set of quantized representations. The Transformer [1], which consists of N global multi-head self-attention blocks (GMSAB), transforms the latent speech representation into content representations.

This is a two-step training. In the pretraining step, the model randomly masks some frames of the latent speech representation and the transformer is responsible to reconstruct these masked parts. The contrastive loss [4] evaluates the similarity between the reconstructed representation and the matched quantized representation. The objective function encourages the Transformer to reconstruct masked parts accurately. In the fine-tuning step, the pre-trained model is fine-tuned with labeled data and adopts Connectionist Temporal Classification (CTC) [11] as the objective function.

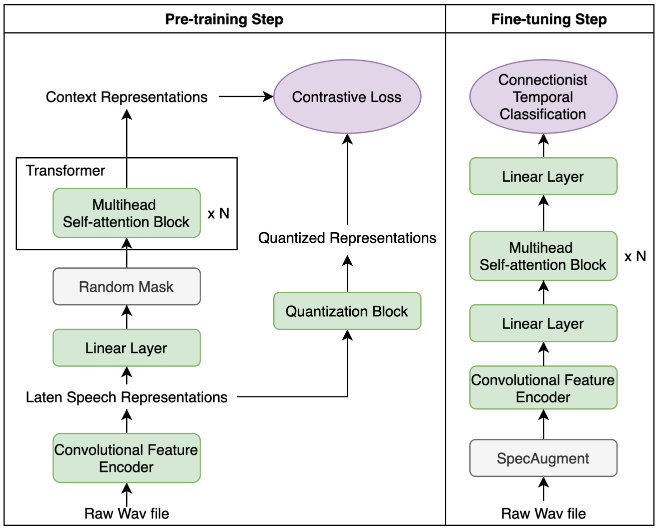
Pre-training Step
Fine-tuning Step
Context Representations
Contrastive Loss
Transformer
Multihead Self-attention Block
x N
Quantized Representations
Random Mask
Quantization Block
Linear Layer
Laten Speech Representations
Convolutional Feature Encoder
Raw Wav file
Connectionist Temporal Classification
Linear Layer
Multihead Self-attention Block
x N
Linear Layer
Convolutional Feature Encoder
SpecAugment
Raw Wav file

Fig. 1. It shows the model architecture of Wav2Vec 2.0 and its training process. We use green to indicate there are learnable weights in these subnetworks and adopt gray to mark processing steps. And a purple ellipse represents loss functions.

2.2. Attention mechanism

Vaswani et al. [1] proposed the global multi-head self-attention mechanism (GMSAM) to draw dependencies between input and output sequences. It is the key component of Transformer [1]. An attention head extracts information from a representation subspace. With multiple heads, the mechanism is able to gain information from multiple different subspaces. We define input X as a sequence of frames, $[x_i, \ldots, x_t, \ldots, x_T]$. X, Q_h, K_h and V_h are the same length. Following shows the detail of GMSAM:

$$\begin{aligned}
Q_h &= Project_h^Q(X) \\
K_h &= Project_h^K(X) \\
V_h &= Project_h^V(X) \\
alpha_h &= softmax(Q_h K_h^T / sqrt(d_k)) \\
head_h &= alpha_h * V_h \\
O &= Project^O([head_1, \ldots, head_h, \ldots, head_H])
\end{aligned}$$

where h is the hth attention head and d_k is the embedding dimension of K_h. *Project* functions are linear layers and H is the total number of heads predefined by users. The o_t is basically a weighted average over the entire V, in GMSAM. The local multi-head self-attention block (LMSAB) [12, 9, 13] constrain o_t to be $[v_{i-(w-1)/2}, \ldots, v_{i+(w-1)/2}]$, where w is known as window size. One advantage of attention mechanisms is their built-in interpretability that researchers can visualize alpha for model evaluation [8] and layer functionality interpretation [14]. There are two common strategies on visualizing a MSAM: visualize the attention per head and present the mean attention over all heads. Kovaleva, Olga, et al. [8] leverages both strategies to analyze the attention pattern of a pre-trained NLP Transformer model named BERT [15]. Kovaleva, et al. [8] categorizes all heatmaps that collected from BERT[15] into four pattern categories:

- Vertical: mainly corresponds to strong attention on special symbols, i.e. symbols stand for the starting of a sentence or serve as sentence delimiters.
- Diagonal: the model learned to attend to local context by itself.
- Vertical+Diagonal: a combination pattern of Vertical and Diagonal.
- Heterogeneous: representing all the rest of patterns

We utilize these categories to describe block-level patterns.

2.3. Cross-blocks parameter sharing

An intuitive interpretation of cross-layers parameter sharing is a recurrent neural network (RNN) considering layer depth as its timestep [16]. This type of network requires a method to decide the network depth. ALBERT [10] sets a fixed network depth. Universal transformer leverages adaptive computation time (ACT) [17] to dynamically decide the depth for each sample and demonstrates its effectiveness on text understanding and generation. Bai et al. (2019) [18] propose a method named Broyden iterations for the same purpose. A common characteristic is that all attention blocks in their networks share the same parameter.

3. ATTENTION VISUALIZATION AND ANALYSIS

Our analyzing object is a pre-trained Wav2Vec 2.0 model[1] from Fairseq [19]. This model has 12 transformer blocks

[1] Wav2Vec 2.0 Base from the pre-training step.

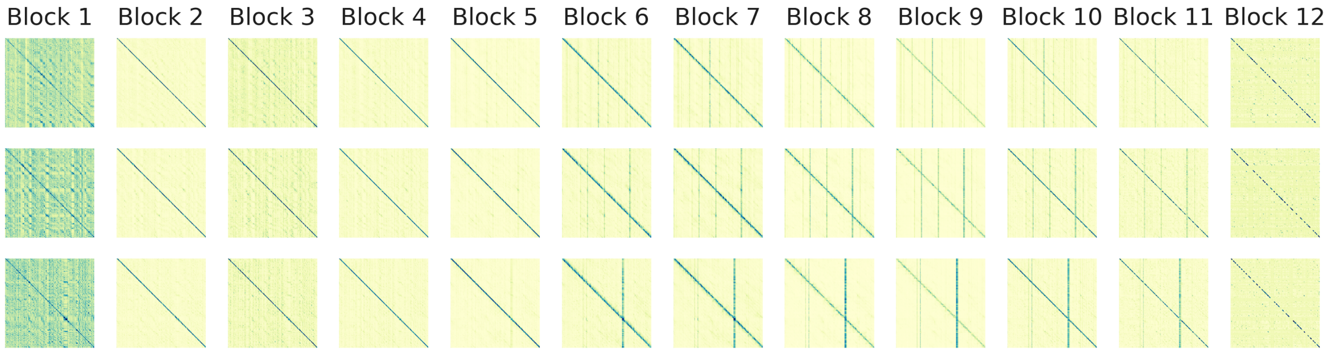

Fig. 2. We sampled three audio recordings from *dev-clean* and plot every block's attention heatmap. A heatmap's x-axis is K's timestep and y-axis is Q's. The duration of these recordings are different. But, in order to summarize block-level attention patterns, we display all heatmaps with the same figure size. We randomly select multiple samples to emphasize that the vertical pattern is a universal situation on certain blocks. *Block 1* is the bottom block and *Block 12* is the top one.

and is trained on *Librispeech-960*. We leverage mean attention visualization to analyze the attention patterns and identify abnormal ones. Unlike Kovaleva, et al. [8], which focuses on input-level patterns, we devote our effort on summarizing block-level patterns. Figure 2 is the attention of randomly selected recordings from Lirbispeech's validation set. We consider that each transformer block's pattern is unique and adopt its block id to represent its pattern. We categorize all 12 patterns into three categories:

- Heterogeneous pattern: *Block 1.*
- Diagonal pattern: *Block 2, 3, 4, 5 and 12.*
- Vertical+Diagonal: *Block 6,7,8,9,10 and 11.*

We categorize *Block 1* as a heterogeneous pattern and assume it is normal since we do not find any sign of abnormality. While we can see a clear diagonal line in all samples' *Block 1*, we also observe an almost uniform attention over the whole sequence. Similar observation on low-level transformer blocks has also been reported in Dai, Zihang, et al. [14]. Thus, we assume this pattern is acceptable. Categorizing *Block 2,3,4,5 and 12* as diagonal patterns is intuitive. A speech frame is strongly related to its neighbors. Therefore, the diagonal patterns are probably normal. Categorizing the remaining blocks as vertical+diagonal is straightforward because, except diagonal lines, there are also a lot of vertical lines in all sampled recordings' heatmap.

3.1. Abnormal pattern

We suspect that *vertical pattern* should not appear in the model of Wav2Vec 2.0. There are two reasons: Firstly, standing from the perspective of matrix operation, this pattern indicates the magnitude of those vectors in K are large. Since both $Projector^Q$ and $Projector^K$ take the same input, $Projector^K$ is oversensitive to these frames and it could be a sign of overfitting. Secondly, according to Kovaleva, et al. [8], vertical patterns strongly correlate to special tokens, such as delimiter, which are designed to modify input sentences in exchange to minor architecture modification across tasks after pre-training step [20] in NLP. Wav2Vec 2.0 does not modify input with any special symbol. Thus, vertical patterns are abnormal and we should avoid it. To achieve this purpose, we utilize the local multi-head self-attention block (LMSAB) to constrain the attention to local region. In other words, we force the attention pattern to be always diagonal.

3.2. Parameter sharing

The pattern of *Block 1* different from the rest blocks. Its attention over the whole sequence triggers us to question the parameter sharing strategy used in early researches that we introduced in Section 2.3. We hypothesis that blocks that belong to the same pattern category are easy to share parameter. Thus, we think *Block 1* should have its own parameter instead of sharing with other blocks due to the pattern difference. For this purpose, we adopt the same method as ALBERT [10] and fix the network depth to be 12 which is same as the pre-trained model's depth. The difference is that, while all blocks in ALBERT [10] share parameters, only top 11 blocks share parameters in our case.

4. EXPERIMENT SETUP

4.1. Dataset

We assume 100 hours of transcribed recordings is an achievable data size for ASR training. Thus, we adopt the *train-clean-100* set from Librispeech corpus [21] as training data for both pre-training step and fine-tuning step. We evaluate all models on the standard Librispeech dev and test sets: *dev-clean* and *test-clean*.

	WER[%]		
Model Name	dev-clean	test-clean	Param Size
G_B1-12	7.62	8.33	95M
L_B1-12	7.67	8.45	95M
L_B6-11	7.16	7.90	95M
L_B2-12	6.76	7.49	95M

Table 1. The model name is formed as [domain attention type]_B[block ID range] where the attention type can only be either LMSAB (*L*) or GMSAB (*G*). The block ID range indicates the blocks that leverage the domain attention type. We always apply GMSAB to unspecified blocks.

4.2. Training

For both pre-training and fine-tuning steps, we mainly follow configurations from Fairseq [19], in which the transformer contains 12 GMSAB. We leverage two Nvidia RTX 3090 GPUs and simulate parallel training on 8 GPUs through setting the update frequency to be 4. We set the maximum token size to be 1.3m per GPU. The equivalent total batch size is 47 audio recordings. In the pre-training step, we train our model for 220k steps. In the fine-tuning step, the total training iteration is 20k steps.

4.3. Decoding and evaluation

We leverage beam search with a language model (LM). We adopt a 4-gram language model from Openslr for all acoustic models. The beam size is 1500. We adopt WER to evaluate a model's training efficiency. A model with lower WER means it is more efficient with limited training data.

5. EXPERIMENTS AND RESULTS

5.1. Local attention

To evaluate the benefit of avoiding vertical patterns. We trained four models: the first one's all 12 blocks are GMSAB. The second one's all 12 blcoks are LMSAB. The third only apply LMSAB to blocks categorized as vertical+diagonal, which is *Block 6,7,8,9,10 and 11*, and the rest blocks are GMSAB. And, the fourth one's top 11 blocks are LMSAB and and the bottom block is GMSAB. We name these models as *G_B1-12*, *L_B1-12*, *L_B6-11* and *L_B2-12*, respectively. We set all LMSABs' window size to be 61.

Table 1 shows the experiment results. The WER difference between *L_B1-12* and *G_B1-12* shows that applying LMSAB to all blocks harms the training efficiency. However, *L_B6-11* outperforms *G_B1-12*. This support that the diagonal pattern harms the performance. *L_B2-12* outperforms most models in Table 1. This confirms our assumption that applying LMSAB to top 11 blocks increase the training efficiency.

	WER[%]		
Model Name	dev-clean	test-clean	Param Size
G_BS1-12	8.14	8.82	17M
G_BS1-11	7.96	8.82	24M
G_BS2-12	7.43	8.15	24M
L_BS2-12	5.87	6.97	24M

Table 2. We leverage similar naming rule as Table 1, except the second part start as *BS* which stands for blocks[*B*] that share[*S*] parameters. The last column is the model size.

5.2. Parameter sharing

We evaluate three parameter sharing configurations using GMSAB only: First, all 12 blocks share a GMSAB. Second, top 11 blocks share a GMSAB and *Block 1* has a separate GMSAB. And third, bottom 11 blocks share a GMSAB and *Block 12* has its own GMSAB. They are named as G_BS1-12, G_BS2-12 and G_BS1-11, respectively. The parameter size of G_BS1-12 is the smallest. The only difference between G_BS2-12 and G_BS1-11 is the blocks that share parameters.

The Table 2 shows that, while G_BS1-12 perform worse than G_B1-12 in Table 1, G_BS2-12 outperforms G_B1-12. This indicates that the configuration of parameter sharing is an essential factor. Arbitrarily sharing parameters among all attention blocks may not unleash this technique's full potential. Moreover, G_BS2-12 outperforms G_BS1-11. This indicates that, instead of parameter size, the sharing configuration is the key.

5.3. Combining both techniques

We optimize the architecture by combining both modification. We adopt LMSAB to top 11 blocks and these blocks share the same parameter. We apply GMSAB to *Block 1*. This model is named as L_BS2-12. The result in Table 2 shows that L_BS2-12 achieves the best performance over models.

6. CONCLUSIONS

Through analyzing the block-level attention pattern, we optimize the transformer's architecture by apply local attention and cross-block parameter sharing on top 11 blocks. Our optimized architecture is more efficient on small dataset than the vanilla Wav2Vec 2.0 [4]. We show that sharing parameters among blocks with similar patterns is more effective than minimizing the parameter size.

7. ACKNOWLEDGEMENTS

This work was supported by Oregon Roybal Center for Aging and Technology Pilot Grant (P30 AG024978) and the I-CONECT project (WWW.i-conect.org) funded by NIH R01-AG051628 and R01-AG056102

8. REFERENCES

[1] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, L ukasz Kaiser, and Illia Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds. 2017, vol. 30, Curran Associates, Inc.

[2] Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al., "Language models are few-shot learners," *arXiv preprint arXiv:2005.14165*, 2020.

[3] Anmol Gulati, James Qin, Chung-Cheng Chiu, Niki Parmar, Yu Zhang, Jiahui Yu, Wei Han, Shibo Wang, Zhengdong Zhang, Yonghui Wu, et al., "Conformer: Convolution-augmented transformer for speech recognition," *arXiv preprint arXiv:2005.08100*, 2020.

[4] Alexei Baevski, Henry Zhou, Abdelrahman Mohamed, and Michael Auli, "wav2vec 2.0: A framework for self-supervised learning of speech representations," *arXiv preprint arXiv:2006.11477*, 2020.

[5] Jonathan D Rodgers, Kris Tjaden, Lynda Feenaughty, Bianca Weinstock-Guttman, and Ralph HB Benedict, "Influence of cognitive function on speech and articulation rate in multiple sclerosis," *Journal of the International Neuropsychological Society: JINS*, vol. 19, no. 2, pp. 173, 2013.

[6] Benjamin G Schultz, Venkata S Aditya Tarigoppula, Gustavo Noffs, Sandra Rojas, Anneke van der Walt, David B Grayden, and Adam P Vogel, "Automatic speech recognition in neurodegenerative disease," *International Journal of Speech Technology*, pp. 1–9, 2021.

[7] Ravichander Vipperla, Steve Renals, and Joe Frankel, "Ageing voices: The effect of changes in voice parameters on asr performance," *EURASIP Journal on Audio, Speech, and Music Processing*, vol. 2010, pp. 1–10, 2010.

[8] Olga Kovaleva, Alexey Romanov, Anna Rogers, and Anna Rumshisky, "Revealing the dark secrets of bert," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 2019, pp. 4365–4374.

[9] Aurko Roy*, Mohammad Taghi Saffar*, David Grangier, and Ashish Vaswani, "Efficient content-based sparse attention with routing transformers," 2020.

[10] Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut, "Albert: A lite bert for self-supervised learning of language representations," *arXiv preprint arXiv:1909.11942*, 2019.

[11] 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*, 2006, pp. 369–376.

[12] Jack Rae and Ali Razavi, "Do transformers need deep long-range memory?," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, Online, July 2020, Association for Computational Linguistics.

[13] Iz Beltagy, Matthew E. Peters, and Arman Cohan, "Longformer: The long-document transformer," 2020.

[14] Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc V Le, and Ruslan Salakhutdinov, "Transformer-xl: Attentive language models beyond a fixed-length context," *arXiv preprint arXiv:1901.02860*, 2019.

[15] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[16] Mostafa Dehghani, Stephan Gouws, Oriol Vinyals, Jakob Uszkoreit, and Lukasz Kaiser, "Universal transformers," *arXiv preprint arXiv:1807.03819*, 2018.

[17] Alex Graves, "Adaptive computation time for recurrent neural networks," *arXiv preprint arXiv:1603.08983*, 2016.

[18] Shaojie Bai, J Zico Kolter, and Vladlen Koltun, "Deep equilibrium models," *arXiv preprint arXiv:1909.01377*, 2019.

[19] Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli, "fairseq: A fast, extensible toolkit for sequence modeling," *arXiv preprint arXiv:1904.01038*, 2019.

[20] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever, "Improving language understanding by generative pre-training," .

[21] 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.