UNDERSTANDING THE ROLE OF SELF ATTENTION IN SPEECH RECOGNITION

Anonymous authors
Paper under double-blind review

ABSTRACT

Transformer-based neural networks have achieved great success in automatic speech recognition (ASR). The key to this achievement is a self-attention (SA) mechanism equipped in the Transformer architecture. While the role and behavior of SA have been actively studied in natural language processing (NLP), SA is not much understood in ASR. Considering the unique characteristics of ASR, it is clear that a separate analysis is necessary. In this paper, we discover interesting properties of SA in Transformer-based ASR encoders by analyzing attention maps, phoneme relationships, and hidden layer representations. Based on the analysis, we propose that the role of attention is broadly categorized into two types: **phonetic** and **localization**. We show that (1) there are heavy redundancies in SA through attention heads and (2) phonetic attention contributes to the performance by standardizing similar phonetic features within the utterance. Grounded by our findings, we present two techniques for practical use. The first method is to modify the network to reuse an attention map for multiple layers, which speeds up the inference by 1.96 times for a 30-second utterance without performance loss. The second method is to merge multiple short utterances before inference as if they were pronounced together, which leads to a 8% reduction of relative word error rate in the LibriSpeech *test-other* dataset.

1 INTRODUCTION

Understanding the exact role of SA would provide intuitions in improving the performance of ASR and designing efficient models.

2 BACKGROUND

2.1 SELF-ATTENTION AND ASR ENCODER

SA is usually contained as a module inside the ASR encoder consisting of stacked Transformer layers.

We briefly review the SA[1] computation procedure. Consider a sequence of d-dimensional column vectors $X = \{x_1, x_2, ...x_T\}$ as input. Each vector corresponds to each frame in speech where total number of T frames are included. The input feature vector is projected to query (Q), key (K), and value (V) of h-th attention head as follows:

$$q_{h,i} = W_h^Q x_i, \quad k_{h,i} = W_h^K x_i, \quad v_{h,i} = W_h^V x_i \quad \left(W_h^Q, W_h^K, W_h^V \in \mathbb{R}^{d_h \times d}\right) \tag{1}$$

$W^{Q,K,V}$ indicates projection matrices for Q, K, V, respectively. $d_h = d/H$ indicates the dimension of each attention head where H is the number of attention heads. The attention map (A), which represents how much frames attend to each other, is computed by scaled dot-product operation and the following softmax. Note that each attention head computes different attention map A_i. A single element of the attention map ($A_{h,ij}$) means how much j-th frame contributes to i-th frame. The output $O = \{o_1, o_2, ...o_T\}$ is then computed as below:

[1] In this paper, SA is equivalently used as multi-head SA.

Table 1: Comparison of the differences between NLP and ASR. Each block compares the input, required context, and the target for training.

NLP	**ASR**
Inputs are discrete (sub-)word tokens. Linguistic (syntactic, semantic) relationships Same tokens are rarely included in a sentence.	Inputs are STFT features extracted with overlap. Phonetic relationships Same phonemes appear very often in a single utterance. Same phonemes can be pronounced differently.
Global (long-range) context is essential. (ex: BERT, QA, Summarization)	Prediction heavily depends on local context.
Target is fixed: (1) Which token should be output (2) Where token should be	Multiple target alignments (CTC, RNNT): (1) Same token may occupy different #frames (2) Same frame may output multiple tokens

$$A_{h,i} = \underset{j}{\text{Softmax}}\left(\frac{q_{h,i}k_{h,j}^T}{\sqrt{d_h}}\right), \quad o_i = W^O \underset{h}{\text{Concat}}\left(\sum_j A_{h,ij} v_{h,j}\right) \quad \left(W^O \in \mathbb{R}^{d\times d}\right) \tag{2}$$

SA layer contains $4d^2$ parameters[2]. As shown in equations, SA requires quadratic memory and computation complexity $O(T^2)$, in exchange for the ability to access any location in the sequence.

2.2 Previous Works on Analyzing Self-Attention

NLP The role of SA has been extensively studied in NLP domain (Kovaleva et al., 2019; Voita et al., 2019; Michel et al., 2019; Rogers et al., 2020; Clark et al., 2019; Tenney et al., 2019).

Although Transformer-based models used in NLP and ASR are common in the architecture, there are huge differences between two domains. Table 1 summarizes the key differences. The valuable findings from NLP cannot be directly applied to ASR. We emphasize that a separate analysis on SA is necessary for ASR, considering the unique characteristics of ASR.

ASR Recently, there have been several studies to understand how SA contributes to the speech domain (Zhang et al., 2021a; Yang et al., 2020; Zhang et al., 2021b).

3 Understanding Self-Attention in ASR

3.1 Setup

We train and evaluate the model on the LibriSpeech-960 (Panayotov et al., 2015) dataset. We extract the 80-dimensional log-Mel filterbank feature from a 25ms window with a stride of 10ms. We use 128 sub-word tokens as vocabulary, built on SentencePiece (Kudo & Richardson, 2018) library using the byte-pair encoding (Sennrich et al., 2016).

We use Conformer-M(medium) (Gulati et al., 2020) as the baseline ASR encoder, trained with CTC (Graves et al., 2006) loss. We train models with the AdamW (Loshchilov & Hutter, 2018) optimizer with a weight decay of 1e-5. We use 4x A100(40GB) GPUs for the experiments. Please refer to Appendix X for the model configuration, training details, and the discussion on why we choose the Conformer as our baseline.

3.2 Attention Map to Attention Map

We start by understanding the characteristics of attention maps. Our analysis utilizes the fact that the attention map A is a probability distribution of length T, which implies that the sum of A_i (i-th row) is always set to 1. All evaluations are performed on LibriSpeech *test-clean* dataset containing 2,620 utterances.

[2]If a SA layer adopts relative position mechanism as Transformer-XL (Dai et al., 2019), there are additional d^2 parameters for positional encoding projection. Conformer in our experiments follows this design.

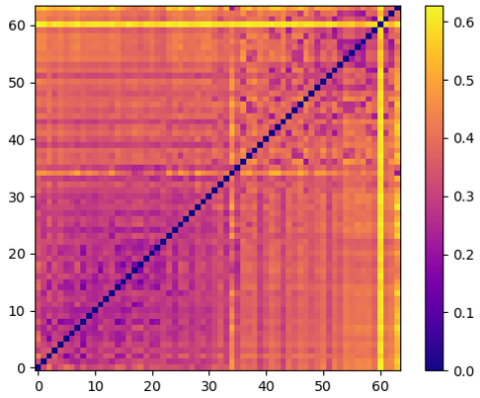


Figure 1: Jensen-Shannon divergence between attention maps. Each row corresponds to 64 attention maps (4 heads x 16 layers) in order. For example, (7, 34) represents the divergence between 4th head of 2nd layer and 3rd head of 9th layer. Smaller divergence implies two attention maps are more likely. This figure shows that lower SA layers attend to similar positions.

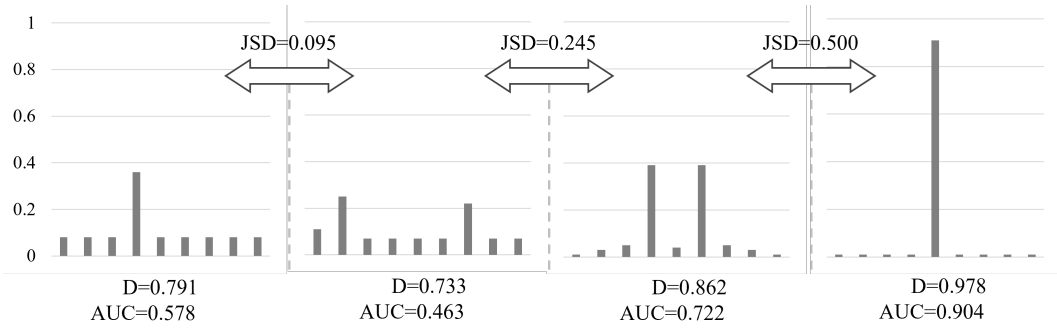


Figure 2: Illustrative example on the Jensen-Shannon divergence, diagonality (D), and AUC of attention span curve.

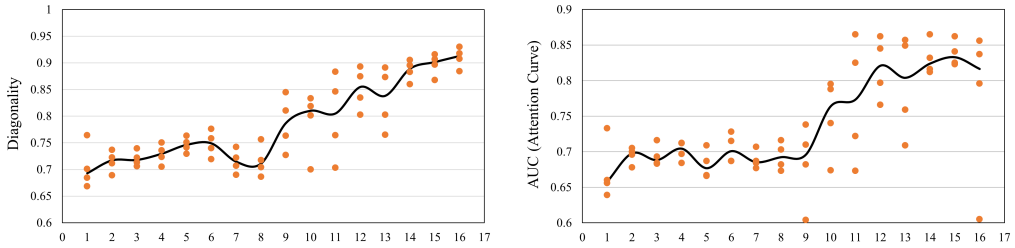
To study the similarity between multiple attention heads, we calculate the Jensen-Shannon divergence (JSD). Figure 1 visualizes the JSD of every attention head pair as a matrix. Surprisingly, we find that the attention heads from the lower layers comprise a large block of small JSD values. This implies that these heads may (1) perform a similar role and (2) carry highly redundant information. For the upper layers, some heads are similar to each other while the others are more related to the heads in lower layers.

To find out the common characteristics, we examine the diagonality of attention heads, following previous works (Zhang et al., 2021b; Yang et al., 2020). Diagonality of the attention map A_h is calculated as below. Diagonality can be interpreted as the negative normalized average attention distance (span-length), averaged through every data sample. We normalize the distance $|i-j|$ by T because input lengths vary.

$$D_h = 1 - \frac{1}{T}\sum_{i=1}^{T}\sum_{j=1}^{T} A_{h,ij} \cdot \frac{|i-j|}{T} \tag{3}$$

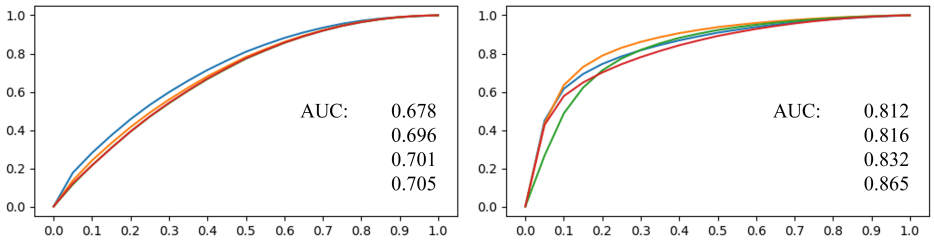
Figure 3(a) shows the diagonality of attention heads. Diagonality tends to increase from lower to upper layers, which means attention heads in upper layers focus more on the local context. Note that this trend is in line with the result from Zhang et al. (2021b). On the other hand, the diagonality of attention heads in lower layers is much smaller and concentrated. In connection with the above observation on JSD, we suggest that lower layers have more scattered attention patterns (smaller diagonality) by attending resembling locations (smaller JSD).

However, the diagonality does not provide information about how evenly attention is distributed. We introduce *attention span curve* to better visualize the uniformness of the attention map. The attention span curve is defined as the sum of attention probabilities within a limited distance. The curve can be also interpreted as a cumulative probability over (relative) distance, computed as below. We further utilize the area-under-curve (AUC) value as a metric.



(a) Diagonality (b) AUC of attention span curve

Figure 3: Plot on (a) diagonality and (b) AUC of attention span curve of attention heads. Each point corresponds to a single head and the y-axis indicates the layer index. Trend line (black) connects the median of values within a layer.



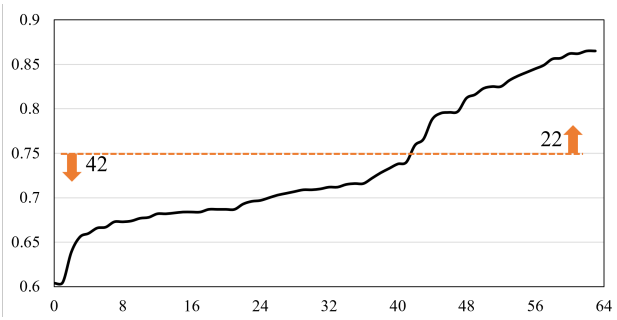
(a) Uniformly distributed attention heads (b) Diagonally concentrated attention heads

Figure 4: Examples of typical attention span curves. (a) and (b) illustrates curves of four attention heads in layers 2 and 14. x-axis and y-axis depict the relative distance and accumulated attention probability, respectively. AUC values are also presented.

$$S_h(r) = \frac{1}{T}\sum_{i=1}^{T}\sum_{j=1}^{T} A_{h,ij} \cdot \mathbb{I}[|i-j| < rT] \quad (r \in [0,1]) \tag{4}$$

Figure 3(b) shows the AUC on attention span curve of attention heads. To help understand the concept, we also show two typical examples of the curve in Figure 4. Diagonality and AUC follow similar tendencies but show slightly different results, especially when diagonality is large. Appendix X shows an example of when these two metrics disagree.

We distinguish attention heads into two patterns, uniformly distributed and diagonally concentrated, based on where the value of the attention span curve reaches the threshold (0.8). If the relative distance at probability 0.8 exceeds 0.45, we categorize the specific head as former. In contrast, if relative distance at probability 0.8 falls behind 0.3, we categorize the head as latter. Within every 64 attention heads, 41 and 18 heads are categorized as uniformly distributed and diagonally con-



#Layer	0	1	2	3
< 0.75	4	4	4	4
#Layer	4	5	6	7
< 0.75	4	4	4	4
#Layer	8	9	10	11
< 0.75	4	2	2	0
#Layer	12	13	14	15
< 0.75	1	0	0	1

Figure 5: Sorted distribution of AUC. Over 64 heads, 42 heads belong to the AUC value under 0.75. Right table indicates the number of attention heads of AUC value under 0.75.

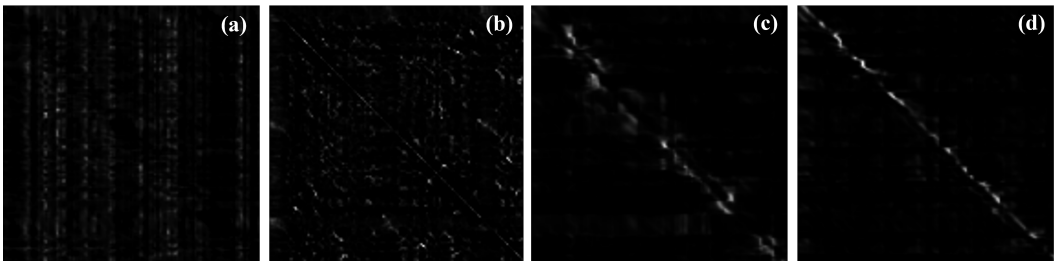


Figure 6: Examples of attention maps. These maps are extracted from different layers but from one utterance. Selected attention head from (a) layer4 (b) layer7 (c) layer12 (d) layer14

centrated, respectively. Only 5 heads are not clearly distinguished. We connect these two patterns to two roles of SA, **phonetic** and **localization**. We explain the reason for the naming in the next Section.

3.3 Attention Map to Phoneme

To further analyze the behavior of SA, we exploit the phoneme information to find out the relationship between SA and phonemes. For the LibriSpeech dataset, we use the frame-level phoneme alignments that are obtained from Montreal Forced Aligner (McAuliffe et al., 2017). Table 2 lists all phoneme classes. We exclude the "silence" phoneme for the analysis. Please visit Appendix X for details on phoneme pre-processing and silence removal.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
AA	AE	AW	AY	AH	EH	ER	EY	IY	IH	O	UH	UW	L	R	M	N	NG
18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35
B	D	DH	G	K	P	T	F	CH	SH	TH	S	Z	V	JH	W	Y	HH

Table 2: Phoneme order for the analysis. Phonemes are extracted from the LibriSpeech lexicon and collapsed into 36 classes. Phonemes are reordered according to their phonological properties.

We convert the attention between two frames as the attention between two phonemes. Specifically, we convert an attention probability A_{ij} to P_{pq} where i-th frame and j-th frame correspond to phoneme p and q, respectively. Therefore, we can calculate the relationship between phonemes per head. We introduce *phoneme attention relationship* (PAR) to understand how SA processes phonological information. PAR shows how much each phoneme attends to the other on average. If two phoneme classes p and q are different, we can simply transport the probability from A_{ij} to P_{pq}. On the other hand, if two frames are within the same phoneme class, we apply a special treatment: we do not include consecutive frames of the same class p in the P_{pp} calculation. We want to separate attending similar frames that are nearby from attending far ones. We focus on the latter because it much better represents the role of SA. PAR is computed as:

$$P_{h,pq} = \frac{T}{|C_p| \cdot |C_q|} \sum_{i \in C_p} \sum_{j \in C_q} A_{h,ij} \quad (p \neq q) \tag{5}$$

$$P_{h,pp} = \frac{T}{|C_p|} \sum_{i \in C_p} \frac{1}{|C_p| - |C_{j,pp}|} \sum_{j \in C_p - C_{j,pp}} A_{h,ij} \tag{6}$$

C_p, C_q indicates the number of frames that corresponds to phoneme class p and q. $C_{j,pp}$ means the number of frames that satisfies two conditions: belong to the same class as j-th frame and all frames between itself and j-th frame also belong to the same class. The idea of phoneme relationship analysis is first introduced in Yang et al. (2020) as phoneme relation map (PRM), but our PAR is different from PRM in two ways; first, PRM do not distinguish consecutive frames of the same class, and second, PRM do not normalize by the number of phonemes of the class inside the utterance.

Figure 7 illustrates how PAR changes through layers. For the lower layers, we observe the diagonal line stands out even though we ignored the consecutive frames of the same phoneme class in calcu-

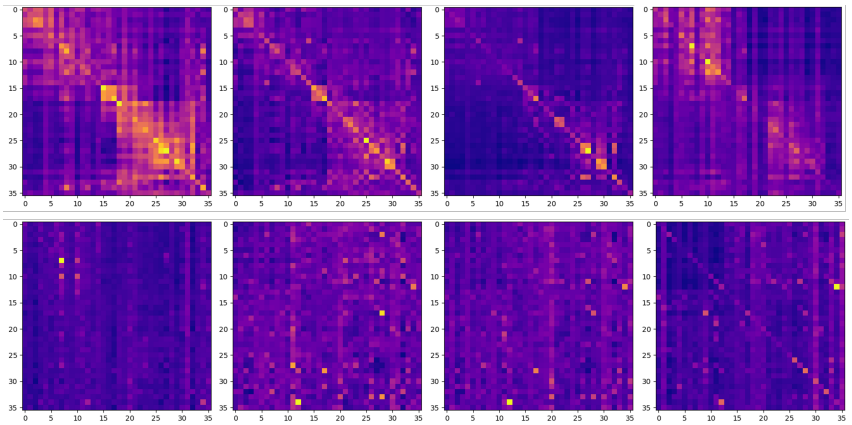
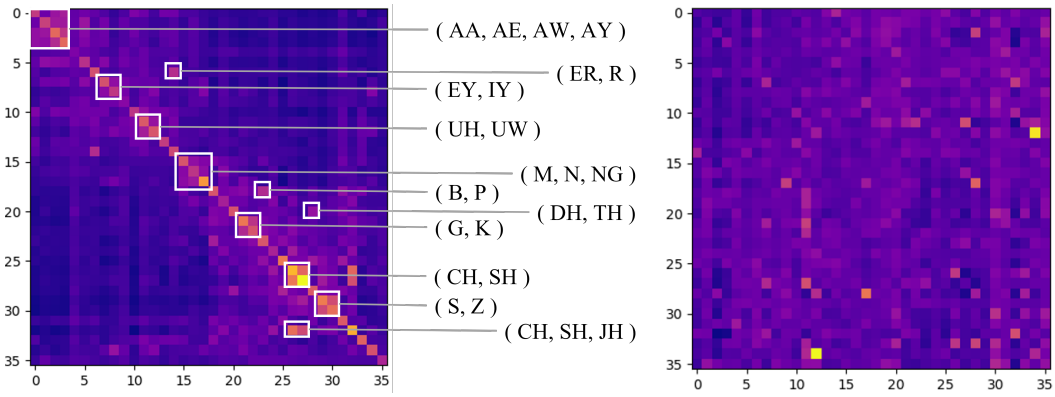


Figure 7: Phoneme attention relationships of self-attention layers. Averaged phoneme attention relationships of even-numbered layers (2nd, 4th, ... 16th) are visualized. Brighter (yellow) values indicate stronger attention between phonemes. Lower layers show a high correlation between similar phonetic features, but these relationships are weakened in upper layers.



(a) Lower half of layers (b) Upper half of layers

Figure 8: Averaged phoneme attention relationship of the lower half (1st, ... 9th) and upper half (10th, ... 16th) layers. Elements that stand out are highlighted. Phonemes with similar phonological properties tend to attend to each other.

lating the PAR. Several blocks of large values also appear through layers, where each layer shows slightly different patterns. In contrast, for the upper layers, there is not much noticeable phoneme relationship because SA in the upper layers tends to focus on local neighbors, represented as large diagonality and AUC, regardless of the content. We suggest that the attention heads in ASR either exploit content-based global information or focus on position-based local information. In speech domain, this can be viewed as **phonetic** role and **localization** role.

For a better interpretation of the phonetic role, we show the average PAR of the lower half and upper half layers in Figure 8. Interestingly, we discover well-known phonological characteristics in the lower half of layers. For example, labial (B, P), velar (G, K), and alveolar (S, Z) consonants highly attend to each other. Nasal phonemes (M, N, NG) also show a high correlation. We notice that not all heads of the phonetic role show diagonal-like PAR. We found some heads are specialized to detect multiple phonemes regardless of the input, generating the PAR with vertical lines. In short, the phonetic role would give attention to (1) phonologically similar frames or (2) frames that contain specialized phonetic features. See Appendix X for examples of two different types of phonetic role. We emphasize that the architecture that is the most suitable for the phonetic behavior is SA, which enables focusing on multiple frames regardless of their location at the same time.

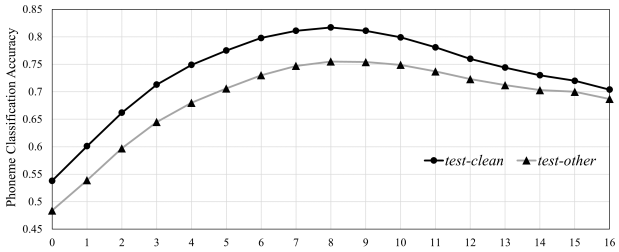


Figure 9: Phoneme classification accuracy for different hidden layers on LibriSpeech *test* datasets. x-axis depicts the layer index. Zeroth entry implies that the classification network is trained on feature vectors obtained before the first SA layer.

Table 3: Model configuration for layer-wise attention map reusing. By reducing the number of unique attention maps, both training and inference speed can be greatly improved. The impact becomes more significant for longer sequences. The front convolutional sub-sampling is not included.

Config.	#Param (M)	Latency(ms) by sequence length (approx. sec)				Training cost(h)
		128 (5.1s)	256 (10.2s)	512 (20.5s)	768 (30.7s)	
1×16	25.45	1.427 (x1.00)	3.743 (x1.00)	11.105 (x1.00)	22.315 (x1.00)	430.0
2×8	24.92	1.245 (x1.15)	2.975 (x1.26)	7.917 (x1.40)	15.054 (x1.48)	337.5
4×4	24.66	1.138 (x1.25)	2.561 (x1.46)	6.288 (x1.77)	11.380 (x1.96)	288.4
8×2	24.52	1.083 (x1.32)	2.350 (x1.59)	5.471 (x2.03)	9.552 (x2.34)	268.8

We hypothesize that this phonetic behavior of SA contributes to the performance by *feature standardization*. We describe this standardization process as "collecting similar information through the sequence and converting each frame more likely to other similar ones". This process takes effect on both sides; similar frames will influence each other to become more similar. We suggest that this homogeneity of similar frames is a desirable characteristic for ASR, because the output text should not be affected by the variance of the frames, such as pronunciation, gender of the speaker, dialects, noise, and speed of the utterance.

3.4 Hidden Layer Representations to Phoneme

We evaluate the effect of feature standardization by phoneme classification. If the features are standardized through SA by phonetic attention heads, we expect the accuracy to be increased because similar phoneme frames would accomplish similar hidden layer representations. Remind that the raw input feature can greatly differ depending on the characteristics of the speaker even if they are of the same phoneme class. For phoneme classification, we extract the hidden layer representations (output of layers) from the baseline model and train a softmax classifier with a single fully-connected layer. Input is 256-dimensional vector and output contains total 37 output classes (36 phonemes + "silence"). We choose the simplest architecture as a classifier to more directly correlate the phoneme accuracy and information in layer representations. See Appendix X for the details on training the classifier and the resulting confusion matrix.

Figure 9 shows the phoneme classification accuracy from different layers. The accuracy of *test-clean* on layer 0 (before SA) is only 53.8%, but it consistently increases to 81.7% at layer 8 and gradually decreases to the end. The result is in line with our hypothesis; for the lower layers, the phonetic attention heads perform the feature standardization, which leads to more distinguishable hidden layer representations. The accuracy decreases for the upper layers because their attention heads mostly perform the localization to aggregate local context and convert phoneme-level information to token-level information for ASR.

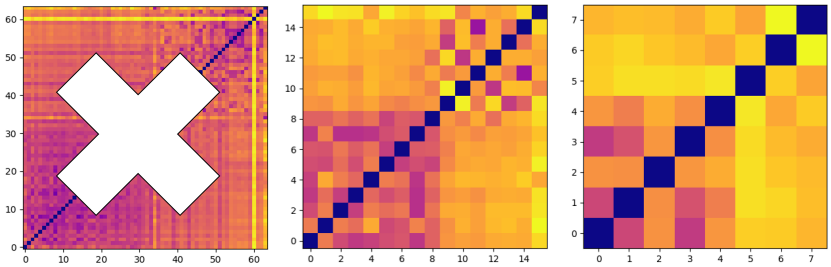
4 Efficient Self-Attention for ASR

4.1 Layer-wise Attention Map Reusing

We propose *layer-wise attention map reusing*, a method to design efficient model by exploiting the redundancy between attention heads. We reuse the computed attention map from the previous layer and reduce the number of all-to-all computations. More specifically, we reuse attention map of l-th

Table 4: Word error rate (%) for models with different attention map reusing configurations. No external language model is used. "H8" indicates that the number of attention heads in the self-attention layer is doubled from 4 to 8.

Configuration	#Heads	*dev-clean*	*dev-other*	*test-clean*	*test-other*
1×16 (Baseline)	64	3.11	8.32	3.24	8.40
2×8	32	?	?	?	?
4×4	16	3.01	8.18	3.29	8.24
$4(\mathtt{H8}) \times 4$	32	3.08	8.11	3.22	8.05
8×2	8	3.30	8.84	3.55	8.74
$8(\mathtt{H8}) \times 2$	16	3.17	8.53	3.37	8.45
$4 + 4 + 8$	12	?	?	?	?
$8 + 4 + 4$	12	?	?	?	?
$4(\mathtt{H8}) + 4(\mathtt{H8}) + 8$	20	?	?	?	?



(a) 2×8 (b) 4×4 (c) 8×2

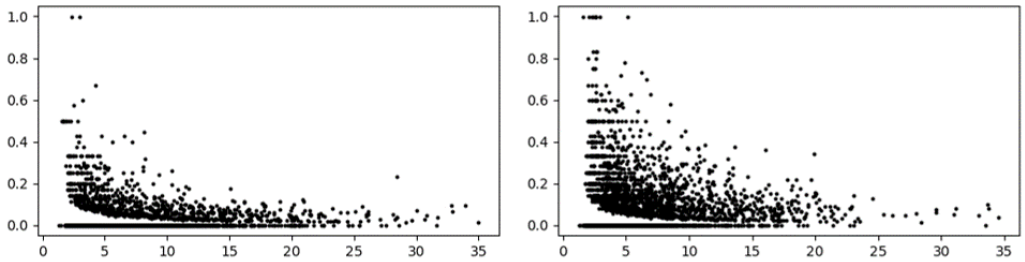
Figure 10: Jensen-Shannon divergence for different attention map reusing configurations.

SA layer to $(l+1), (l+2), ...(l+M-1)$-th consecutive SA layers. If a single attention map is shared through M layers, the computation burden of SA can be reduced by M times. During training, the reused attention map receives gradients from M layers. Note that this layer-wise reusing is easy to implement and fully supported by modern accelerator hardware. For SA layers that receive the pre-computed attention map, query and key are also not needed and can be removed. We double the output dimension of V ($W_h^V \in \mathbb{R}^{2d_h \times d}, W^O \in \mathbb{R}^{d \times 2d}$) to compensate the parameter size for those layers. Unless specified, we maintain the number of attention heads.

Table 3 compares different reusing configurations. Configuration $X \times Y$ means X layers are grouped to share the same attention map and total Y groups exist. Therefore, there are HXY unique attention heads for each model. For example, 8×2 accelerates the inference by 2.34x times (for 30-second utterance) and training by 1.60x times, as a result of 1/8 $O(T^2)$ operations. Inference speed is evaluated on a single RTX-Titan(24GB) GPU and training cost is measured in GPU-hours on A100(40GB) GPUs. For each sequence length, we measure the inference time with the largest possible batch size which does not cause memory overflow.

For each configuration, We train the model from scratch with the same training setting. Table 4 shows the result on LibriSpeech *dev* and *test* datasets. 4×4 achieves comparable or better performance to the baseline. Especially, the improvement on *other* (noisy) datasets is noticeable. 8×2 shows slightly worsen performance. We think this loss is caused by insufficient number of heads and lack of hierarchy. Furthermore, we also tested a variant that contains twice more attention heads (H8).These variants show consistently better performance than their counterparts with almost identical inference speed and exactly equal number of parameters.

We visualize the JSD of different configurations in Figure 10 as Section 3.2. We verify that the patterns resemble to the baseline model, which implies that the models learn the similar role during training. We mention that the uniform split of M may be sub-optimal, and we left the exploration of reusing pattern to future work.



(a) WER on *test-clean* (b) WER on *test-other*

Figure 11: Word error rate of LibriSpeech *test* datasets. The x-axis depicts the length of the utterance in seconds. Error rate tends to be higher for shorter inputs in both datasets.

Table 5: Word error rate(%) for the concatenated inference. Concat-`X` indicates that input features are concatenated before forward propagation to satisfy at least `X` seconds long. `I` indicates the utterance is simply duplicated.

	Original	Concat-`15I`	Concat-`10`	Concat-`15`	Concat-`20`
test-clean	3.24(+14.2%)	3.70	3.22(-0.1%)	3.21 (-0.1%)	**3.17** (-2.2%)
test-other	8.40(+17.3%)	9.85	7.87(-6.3%)	7.76 (-7.6%)	**7.73** (-8.0%)

4.2 Concatenated Inference

Based on our hypothesis on the feature standardization, longer input would benefit from various pronunciations from similar phonetic features. Figure 11 shows the relationship between the sequence length and WER. Clearly, shorter utterance suffers from larger error rate, which can be partially explained by our observations.

We propose *concatenated inference*.

Table 5 shows the improvement from the concatenated inference.

Figure X shows the same WER plot as above. (appendix)

5 Related Work

Efficient Attention Computation

Attention Map Reusing in NLP

Efficient ASR Encoder

Attention Head Regularization

Self-Supervised Speech Representation (Baevski et al., 2020; 2021; Hsu et al., 2021) https://github.com/s3prl/s3prl

6 Conclusion

References

Alexei Baevski, Yuhao Zhou, Abdelrahman Mohamed, and Michael Auli. wav2vec 2.0: A framework for self-supervised learning of speech representations. *Advances in Neural Information Processing Systems*, 33, 2020.

Alexei Baevski, Wei-Ning Hsu, Alexis Conneau, and Michael Auli. Unsupervised speech recognition. *arXiv preprint arXiv:2105.11084*, 2021.

Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D Manning. What does bert look at? an analysis of bert's attention. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pp. 276–286, 2019.

Zihang Dai, Zhilin Yang, Yiming Yang, Jaime G Carbonell, Quoc Viet Le, and Ruslan Salakhutdinov. Transformer-xl: Attentive language models beyond a fixed-length context. In *ACL (1)*, 2019.

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*, pp. 369–376, 2006.

Anmol Gulati, James Qin, Chung-Cheng Chiu, Niki Parmar, Yu Zhang, Jiahui Yu, Wei Han, Shibo Wang, Zhengdong Zhang, Yonghui Wu, and Ruoming Pang. Conformer: Convolution-augmented transformer for speech recognition. In *Proc. Interspeech 2020*, pp. 5036–5040, 2020.

Wei-Ning Hsu, Benjamin Bolte, Yao-Hung Hubert Tsai, Kushal Lakhotia, Ruslan Salakhutdinov, and Abdelrahman Mohamed. Hubert: Self-supervised speech representation learning by masked prediction of hidden units. *arXiv preprint arXiv:2106.07447*, 2021.

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)*, pp. 4365–4374, 2019.

Taku Kudo and John Richardson. Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pp. 66–71, 2018.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2018.

Michael McAuliffe, Michaela Socolof, Sarah Mihuc, Michael Wagner, and Morgan Sonderegger. Montreal forced aligner: Trainable text-speech alignment using kaldi. In *Interspeech*, volume 2017, pp. 498–502, 2017.

Paul Michel, Omer Levy, and Graham Neubig. Are sixteen heads really better than one? *Advances in Neural Information Processing Systems*, 32:14014–14024, 2019.

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)*, pp. 5206–5210. IEEE, 2015.

Anna Rogers, Olga Kovaleva, and Anna Rumshisky. A primer in bertology: What we know about how bert works. *Transactions of the Association for Computational Linguistics*, 8:842–866, 2020.

Rico Sennrich, Barry Haddow, and Alexandra 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)*, pp. 1715–1725, 2016.

Ian Tenney, Dipanjan Das, and Ellie Pavlick. Bert rediscovers the classical nlp pipeline. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 4593–4601, 2019.

Elena Voita, David Talbot, Fedor Moiseev, Rico Sennrich, and Ivan Titov. Analyzing multi-head self-attention: Specialized heads do the heavy lifting, the rest can be pruned. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 5797–5808, 2019.

Shuwen Yang, Andy T. Liu, and Hung yi Lee. Understanding self-attention of self-supervised audio transformers. In *Proc. Interspeech 2020*, pp. 3785–3789, 2020. doi: 10.21437/Interspeech.2020-2231.

Shucong Zhang, Erfan Loweimi, Peter Bell, and Steve Renals. Stochastic attention head removal: A simple and effective method for improving transformer based asr models. In *Proc. Interspeech 2021*, 2021a.

Shucong Zhang, Erfan Loweimi, Peter Bell, and Steve Renals. On the usefulness of self-attention for automatic speech recognition with transformers. In *2021 IEEE Spoken Language Technology Workshop (SLT)*, pp. 89–96. IEEE, 2021b.

A APPENDIX

You may include other additional sections here.