MLP-BASED ARCHITECTURE WITH VARIABLE LENGTH INPUT FOR AUTOMATIC SPEECH RECOGNITION

Anonymous authors
Paper under double-blind review

ABSTRACT

We propose multi-layer perceptron (MLP)-based architectures suitable for variable length input. Recently, several such architectures that do not rely on self-attention have been proposed for image classification. They achieve performance competitive with that of transformer-based architectures, albeit with a simpler structure and low computational cost. They split an image into patches and mix information by applying MLPs within and across patches alternately. Due to the use of MLPs, one such model can only be used for inputs of a fixed, pre-defined size. However, many types of data are naturally variable in length, for example acoustic signals. We propose three approaches to extend MLP-based architectures for use with sequences of arbitrary length. In all of them, we start by splitting the signal into contiguous tokens of fixed size (equivalent to patches in images). Naturally, the number of tokens is variable. The two first approaches use a gating mechanism that mixes local information across tokens in a shift-invariant and length-agnostic way. One uses a depthwise convolution to derive the gate values, while the other rely on shifting tokens. The final approach explores non-gated mixing using a circular convolution applied in the Fourier domain. We evaluate the proposed architectures on an automatic speech recognition task with the Librispeech and Tedlium2 corpora. Compared to Transformer, our proposed architecture reduces the WER by $1.9/3.4\,\%$ on Librispeech test-clean/test-other set, and $1.8/1.6\,\%$ on Tedlium2 dev/test set, using only $75.3\,\%$ of the parameters.

1 INTRODUCTION

Self-attention, the well-known building block of Transformer (Vaswani et al., 2017), appeared in natural language processing (NLP) where it caused a breakthrough. Soon enough, it propagated to the fields of computer vision and automatic speech recognition (ASR). In particular, recent end-to-end ASR systems based on self-attention architecture, e.g., Transformer (Karita et al., 2019a) and Conformer (Gulati et al., 2020), provide state-of-the-art performance.

Recently, several architectures entirely based on MLP have been proposed in the area of computer vision. MLP-based architectures have a simple structure, and achieve performance competitive with that of Transformer-based architectures, despite having less parameters and a lower computational complexity. They split an image into patches and reshape it into a $(\text{channels} \times \text{patches})$ matrix used as input. An illustration of the process, and its analog for variable length sequences, is shown in Figure 1. MLP-based architecture such as MLP-Mixer (Tolstikhin et al., 2021) and gMLP (Liu et al., 2021) consist of MLP across the channel dimension and MLP across the patch dimension (also referred to as spatial dimension in computer vision tasks). MLP across the channel dimension mixes information between channels like a typical feed-forward network(FFN). MLP across the patches dimension mixes information between patches. All these different works demonstrate that this process capture sufficient information, and that self-attention is not always necessary. The small model size and low computational cost of the MLP-based architecture are also useful for ASR. However, MLP across the token dimension requires fixed size inputs, and sequence length of acoustic data is generally variable.

In this paper, we propose three approaches that can handle variable length inputs. First, the input sequences are broken down into contiguous chunks that we call tokens (analog of patches in images). See Figure 1 for an illustration. Building on previous approaches, we propose three new token

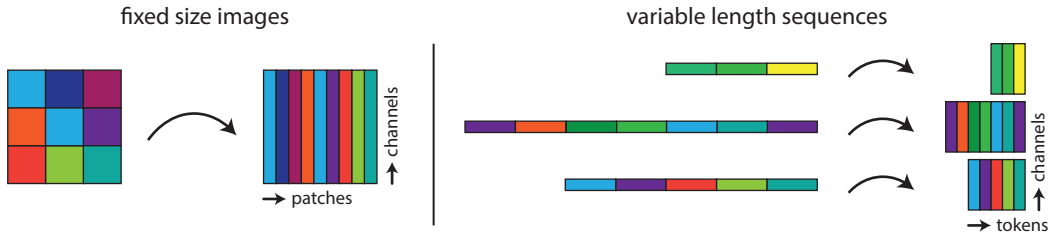

Figure 1: Illustration of the difference between patching for fixed-size images (left) and variable length sequences (right). We rename patches as *tokens* to adapt to the semantics of sequences.

mixing units. Two of them rely on gating, where the input is split in two parts, one is transformed and then multiplied with other. In the first kind of unit, convolutional-MLP (C-MLP), we use a depthwise convolution to transform the input. This applies an MLP-like operation locally along the token dimension and makes the network shift invariant. The second approach, time-shift MLP (T-MLP), concatenates shifted parts of the input. The third approach is the non-gating one. In this case, we apply a simple depthwise convolution, thus mixing tokens with their local neighbors. However, we make use of the fast Fourier transform (FFT) to do so efficiently. This approach is termed Fourier MLP (F-MLP). We applied these MLP-base methods to the connectionist temporal classification (CTC) (Graves et al., 2006) ASR model and evaluated on two datasets, Librispeech (Panayotov et al., 2015) and Tedlium2 (Rousseau et al., 2014). We found different trade-offs between accuracy in terms of word error rate (WER), number of parameters, and inference speed. However, when matching the number of parameters, the proposed MLP architectures consistently outperform self-attention based models, decreasing the WER by as much as $3\,\%$.

Related works. We summarize the recently proposed MLP-based architectures. MLP-Mixer (Tolstikhin et al., 2021) consists of channel-mixing MLPs and patch-mixing MLPs. It alternately applies these MLPs and mixes information between channels and information between patches. gMLP (Liu et al., 2021) also consists of MLPs across the channel and patch dimensions. To more effectively mix patches, it proposes a spatial gating unit (SGU), a module containing MLP across the patch dimension and used to modulate the output of the unit. S^2-MLP (Yu et al., 2021a) and S^2-MLPv2 (Yu et al., 2021b) mix information across the patch dimension by shift operation along the width dimension and height dimension called spatial-shift operation. GFNet (Rao et al., 2021) efficiently applies learnable filters in the frequency domain. It performs a 2D Fourier transform of the input image, multiplies by the filters, and transform back to image domain. Then, MLP is applied across the channels. In the area of NLP, FNet (Lee-Thorp et al., 2021) has been proposed. It applies a 2D Fourier transform to the input sequence, retains the real part, and follows with MLP across channels.

Contributions. Our contributions in this work are as follows. (i) We propose three MLP-based architectures suitable for sequences of arbitrary lengths. (ii) We evaluate our architecture for ASR on two different datasets. We show that the proposed C-MLP architecture outperforms the celebrated self-attention, both in accuracy and inference speed. To the best of our knowledge, this is the first time that MLP-based architecture is applied to ASR.

2 Conventional MLP-based architecture

MLP-based architecture consists of two sub-types of MLP. One, usually a feed forward network, mixes information between channels. The other mixes information between tokens. The long range dependencies The latter allows to model long-range dependencies just like self-attention would.

2.1 Architecture for channel mixing

One structure of the MLP-based model is that token mixing operation is first performed and then channel mixing operation follows. MLP-Mixer and GFNet have this structure. MLP-Mixer first applies the MLP across the token dimension and then applies the MLP across the channel dimension. GFNet first applies a module to mix information in token dimension and then perform the MLP across the channel dimension.

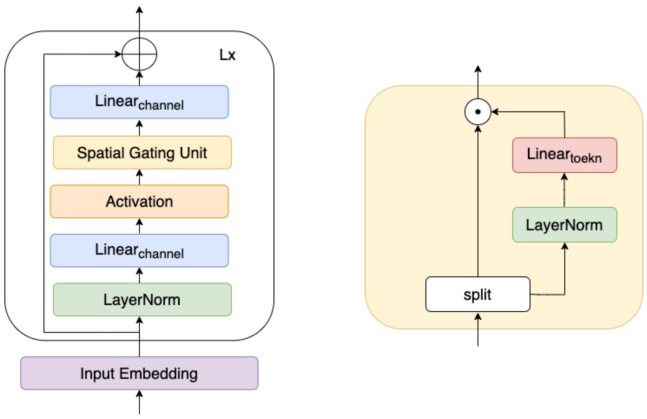

Figure 2: Overview of the gMLP (left) and Spatial Gating Unit (right). Linear$_{\text{channel}}$ and Linear$_{\text{token}}$ refer to linear layer across the channel dimension and token dimension.

Another structure is that token mixing operation is sandwiched between channel mixing operation. gMLP has this structure. In gMLP, first, linear projection across the channel dimension is applied to the input. Let $\mathbf{X} \in \mathbb{R}^{D \times N}$ be the D-dimensional input with sequence length N.

$$\begin{aligned} \mathbf{X} &= \text{Linear}_{\text{channel}}^{(D \to D')}(\mathbf{X_{in}}) \\ &= \mathbf{W_1 X_{in}} + \mathbf{b_1} \end{aligned} \tag{1}$$

where Linear$_{token}(\cdot)$ refers to linear projection across the channel dimension. $\mathbf{W_1} \in \mathbb{R}^{D' \times D}$ and $\mathbf{b_1} \in \mathbb{R}^{D'}$ refer to weights and biases of linear projection. Next, the SGU is followed.

$$\mathbf{Z} = \text{SGU}(\mathbf{X}) \in \mathbb{R}^{D \times N} \tag{2}$$

Finally, linear projection across the channel dimension is applied again.

$$\mathbf{X_{out}} = \text{Linear}_{\text{channel}}^{(2D \to D)}(\mathbf{Z}) \tag{3}$$

In this way, a module for token-mixing called SGU is sandwiched between linear projection across the channel dimension.

2.2 ARCHITECTURE FOR TOKEN MIXING

For the MLP-based model, how to grasp the relationship between tokens is an important point. The conventional MLP-based architecture has a module that applies an MLP layer to the transposed input and converts its token dimension, in addition to the general MLP layer applied to the channel dimension. For example, MLP-Mixer simply applies the MLP across the token dimension gMLP has a module called Spatial Gating Unit (SGU) that consists of MLP layer across the token dimension. We keep the name Spatial Gating Unit by analogy to the gMLP for images even though the dimension here is temporal indeed The overview of the SGU is illustrated in Figure 2. The SGU splits the input $\mathbf{X} \in \mathbb{R}^{D' \times N}$ into $\mathbf{X}_{\text{r}} \in \mathbb{R}^{\frac{D'}{2} \times N}$ and $\mathbf{X}_{\text{g}} \in \mathbb{R}^{\frac{D'}{2} \times N}$, and the perform following gating operation.

$$\text{SGU}(\mathbf{X}) = \mathbf{X}_{\text{r}} \odot \mathbf{H}_{\text{SGU}} \tag{4}$$

where $\odot$ denotes element-wise multiplication. This gating operation mixes the information of token effectively. $\mathbf{H}_{\text{SGU}}$ is formulated as follows,

$$\begin{aligned} \mathbf{H}_{\text{SGU}} &= \text{Linear}_{\text{token}}^{(N \to N)}(\mathbf{X}_{\text{g}}^{\mathsf{T}}) \\ &= (\mathbf{W_2 X}_{\text{g}}^{\mathsf{T}})^{\mathsf{T}} + \mathbf{b_2}, \end{aligned} \tag{5}$$

where Linear$_{token}(\cdot)$ refers to linear projection across the token dimension. $\mathbf{W_2} \in \mathbb{R}^{N \times N}$ and $\mathbf{b_2} \in \mathbb{R}^{N}$ refer to weights and biases of linear projection. The size of $\mathbf{W_2}$ is fixed, so the SGU cannot take variable sequence length inputs. We propose two approaches as an alternative to the SGU that can mix temporal information of the variable sequence length inputs.

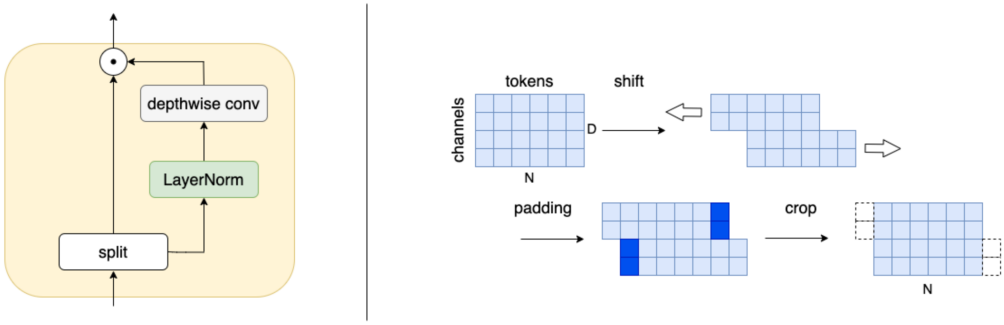

Figure 3: Overview of Convolutional Gating Unit (left) and temporal-shift operation (right).

3 ARCHITECTURE FOR VARIABLE LENGTH INPUT

We describe here our three proposed approaches for token mixing. These approaches are suitable to sequences of any length. Furthermore, they are shift invariant.

3.1 CONVOLUTIONAL-MLP (C-MLP)

The conventional MLP-based architectures described in Section 2 mix information globally over the full extent of the data. However, for long sequences, we think it is sufficient to mix temporally local information. We do so by using a convolutional layer in time direction. We call this architecture Convolutional-MLP (C-MLP). C-MLP has the same structure as gMLP, but a Convolutional Gating Unit (CGU) replaces the SGU. The structure of a CGU is shown in Figure 3. CGU replaces linear projection across the token dimension in SGU with depthwise convolution. $\mathbf{H}_{\mathrm{CGU}}$ corresponding to $\mathbf{H}_{\mathrm{SGU}}$ in Equation 5 is formulated as follows,

$$\mathbf{H}_{\mathrm{CGU}} = \mathbf{K} \star \mathbf{X}_{\mathrm{g}} \in \mathbb{R}^{\frac{D}{2} \times N}, \tag{6}$$

$$\mathbf{H}_{:,i} = \sum_{j=1}^{k} \mathbf{K}_{:,j} \odot \mathbf{X}_{:,k+i-j} \tag{7}$$

where $\mathbf{K} \in \mathbb{R}^{\frac{D}{2} \times N}$ is the $\frac{D}{2}$-dimensional kernel with kernel size n. $\mathbf{X}_{\mathrm{g}} \in \mathbb{R}^{\frac{D}{2} \times N}$ and $\mathbf{H}_{\mathrm{CGU}} \in \mathbb{R}^{\frac{D}{2} \times N}$ are $\frac{D}{2}$-dimensional features with sequence length N.

The operation in Equation 7 is realized by depthwise convolution. This operation to obtain filter $\mathbf{H}_{\mathrm{CGU}}$ is independent of the channel dimension. We also experiment with the case that linear projection across the channel dimension is applied to the filter $\mathbf{H}_{\mathrm{CGU}}$. In this case, new filter $\mathbf{H}'_{\mathrm{CGU}}$ is formulated as follows.

$$\mathbf{H}'_{\mathrm{CGU}} = \mathrm{Linear}_{\mathrm{channel}}^{(\frac{D}{2} \to \frac{D}{2})}(\mathbf{H}_{\mathrm{CGU}}) \in \mathbb{R}^{\frac{D}{2} \times N} \tag{8}$$

We refer to this architecture as Convolutional-MLP′ (C-MLP′).

3.2 TEMPORAL-SHIFT MLP (TS-MLP)

In C-MLP, we used learnable kernel for convolution across the temporal dimension. Next, we consider the case performing a convolution using kernels with fixed parameters. Let $\mathbf{X} \in \mathbb{R}^{D \times N}$ be the D-dimensional input with sequence length N. We prepare one kernel for each channel dimension. Let $\mathbf{K} = \{k_1, \ldots, k_D\}$ be the kernels. We set the kernels as follows.

$$\mathbf{k}_i = [0, 0, 1], \mathbf{k}_j = [1, 0, 0] \tag{9}$$

where $i \in \{1, \ldots, \frac{D}{2}\}$, $j \in \{\frac{D}{2} + 1, \ldots, D\}$. This corresponds to the shift-operation proposed in S^2-MLP (Yu et al., 2021a). In this case, the input is shifted in the temporal direction with shift size 1. We propose Temporal Shift-MLP (TS-MLP) as a method of mixing time-dimensional information by applying a shift operation in the time direction. TS-MLP uses this Temporal-Shift Gating Unit

(TSGU). The TSGU replaces depthwise convolution in CGU with temporal-shift operation. Matrix $\mathbf{H}_{\mathrm{TSGU}}$ corresponds to $\mathbf{H}_{\mathrm{CGU}}$ in Equation 6, and is formulated as follows,

$$\mathbf{H}_{\mathrm{TSGU}} = \mathrm{Shift}(\mathbf{X}_{\mathbf{g}}) \tag{10}$$

Shift(·) indicates the temporal-shift operation. It is illustrated in Figure 3. The input is divided into two groups in the channel dimension, one is shifted forward and the other is shifted backward along the time dimension. Next, both ends are padded with zeros and cut to the original length. The shift operation is parameter-free and mixes time information by shifting it little by little in the time direction.

3.3 Non-gating Variant: Circular Convolution via the FFT

In both C-MLP and TS-MLP, we use a gating unit based on the SGU, as in gMLP. To investigate the usefulness of gating, we introduce a third architecture that does not use it. This is a circular depthwise convolution constructed as follows. It can be understood as similar to C-MLP, but without the split, and where all the input is processed through the depthwise convolution. The fixed size filters are defined in the time-domain, but applied in the frequency domain. Thus, the filter is easily applied to sequences of any length by applying an FFT padded to the length of the input sequence to the filters, and working in the frequency domain. We call this architecture Fourier-MLP (F-MLP). The output of the circular depthwise convolution is computed as follows,

$$\hat{\mathbf{X}} = \mathcal{F}^{-1}\left[\mathcal{F}[\mathbf{X}] \odot \mathcal{F}[\mathbf{H}]\right], \tag{11}$$

where $\mathbf{X} \in \mathbb{R}^{D \times N}$ is the input signal and $\mathbf{H} \in \mathbb{R}^{D \times n}$ contains the filters in its rows. The FFT and its inverse are denoted $\mathcal{F}[\cdot]$ and $\mathcal{F}^{-1}[\cdot]$, respectively. We assume that the operation zero-pads $\mathbf{H}$ to size $D \times N$ to match $\mathbf{X}$

GFNet (Rao et al., 2021) also apply element-wise multiplication to learnable filters and input feature in frequency domain. GFNet defines learnable filters in frequency domain. To take variable length input, GFNet should define sufficient large filters and convert all input feature into frequency domain at the same size as filters. On the other hand, F-MLP defines filters in the time domain, and are not restricted to a fixed size input. Therefore, F-MLP has less parameters than GFNet.

4 EXPERIMENTS

We now turn to the evaluation of the proposed architecture for the task of ASR.

4.1 Experimental Setup

We applied our architectures to the non-autoregressive CTC-based ASR model. The experiments were conducted using the ESPnet tool kit (Watanabe et al., 2018). We use Transformer-encoder as self-attention based baseline, and FNet (Lee-Thorp et al., 2021) and GFNet (Rao et al., 2021) as MLP-based baseline. All methods including the proposed ones consist of 18-layers. Other models also consist of 18-layer encoder. The models are trained for 50 epochs. We use the model parameters obtained by averaging 10 models with top 10 validation scores. The outputs are decoded by greedy decoding for CTC, without using any external language model.

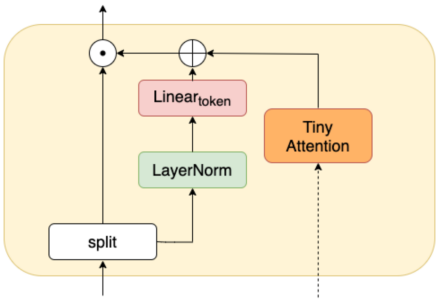

Figure 4: SGU with a tiny self-attention.

Encoder layer structure. There are some possible variations in the structure of the encoder layer. For example, gMLP consist of multiple layer of gMLP block and the SGU is sandwiched between channel projection layers in that block. In GFNet, mixing between tokens is performed first, and then mixing between channels. We try these two types of architecture. From preliminary results, we

Table 1: Experimental results on Librispeech and Tedlium2.

method	params	WER (Librispeech) test-clean	WER (Librispeech) test-other	WER (Tedlium2) dev	WER (Tedlium2) test
Baseline					
Transformer-based model	16.2M	13.3%	32.2%	14.4%	13.6%
FNet (Lee-Thorp et al., 2021)	11.5M	33.8%	63.3%	28.5%	28.3%
GFNet (Rao et al., 2021)	16.2M	12.5%	31.4%	13.6%	12.7%
Ours					
F-MLP	11.5M	16.0%	37.0%	16.3%	15.8%
F-MLP+tiny attn	13.8M	12.1%	30.8%	13.2%	13.0%
C-MLP	9.3M	13.4%	33.8%	14.5%	13.5%
C-MLP+tiny attn	12.2M	11.4%	28.8%	12.6%	12.0%
C-MLP′	14.0M	12.1%	31.9%	12.8%	12.0%
C-MLP′+tiny attn	16.9M	**11.1**%	**28.3**%	**11.5**%	**11.4**%
TS-MLP	**9.1**M	16.0%	40.0%	18.1%	17.5%
TS-MLP+tiny attn	12.1M	13.5%	32.6%	14.1%	13.5%

choose the gMLP sandwich architecture for C-MLP and TS-MLP, and the GFNet style architecture for F-MLP. For Transformer encoder, we basically follow the architecture as in Karita et al. (2019b), which consists of 4 attention heads and 256 hidden units.

Tiny self-attention module. gMLP (Liu et al., 2021) demonstrate that adding a small self-attention module to the SGU can improve performance at the cost of a modest increase in resource usage. The input of the tiny self-attention module is the output of the normalization layer of the gMLP Block. The output of the tiny self-attention module is added to the end of the right path of SGU. We also experimented with this tiny self-attention module for our proposed gating units. We use self-attention with a single head of size 128.

4.2 DATASETS

We measure performance on two datasets: the 100h Librispeech in English, the 207 hours Tedlium2 in English. We use 80-dimensional log-mel-spectrogram features and 3-dimensional pitch features as inputs. We apply SpecAugment (Park et al., 2019) and speed perturbation (Ko et al., 2015) for data augmentation. For Librispeech, We tokenize text into 300 subwords, and for Tedlium2, we tokenize text into 500 subwords. We created subwords with SentencePiece (Kudo & Richardson, 2018)

4.3 RESULTS

Table 1 provides comparison of the parameter size and word error rate (WER) on Librispeech and Tedlium2. We evaluate Transformer-based model as the baseline. We also evaluate FNet and GFNet as they resemble F-MLP.

In Table 1, we see that F-MLP is $71.0\,\%$ of the model size while model size of GFNet which also use Fourier transform is the same as that of Transformer-based model. Compared to the Transformer-based model, F-MLP degrade WER by $1.2/0.3\,\%$ on Librispeech test-clean/test-other set and $1.2/0.3\,\%$ on Tedlium2 dev/test set. However, F-MLP with a tiny self-attention improves WER by $1.2/0.3\,\%$ on Librispeech test-clean/test-other set and $1.2/0.3\,\%$ on Tedlium2 dev/test set from F-MLP and outperforms Transformer-based model and GFNet. In addition, even when combined with tiny self-attention, its model size is smaller than that of Transformer-based model and GFNet.

C-MLP achieves competitive performance with Transformer-based model with only $57.1\,\%$ of its parameters. C-MLP′ achieves the best WER in Table 1 and improves WER by $1.2/0.3\,\%$ on Librispeech test-clean/test-other set, and $1.6/1.6\,\%$ on Tedlium2 dev/test set. It increases the model size a little, but it has still only $4.3\,\%$ of model size of Transformer-based model. C-MLP with a tiny attention improves WER by $1.9/3.4\,\%$ on Librispeech test-clean/test-other set, and $1.8/1.6\,\%$

Table 2: Comparison of model size and WER. In order to stabilize the learning of deep models, all models were trained using InterCTC (Lee & Watanabe, 2021), a regularization technique used in CTC-based ASR.

			WER (Librispeech)		WER (Tedlium2)	
method	**layers**	**params**	**test-clean**	**test-other**	**dev**	**test**
Transformer	9	9.1M	14.1%	32.9%	15.2%	14.6%
F-MLP	15	9.9M	15.5%	37.3%	17.3%	16.4%
C-MLP	18	9.3M	**11.5**%	31.2%	13.6%	**12.7**%
TS-MLP	18	9.1M	14.6%	38.0%	17.6%	16.8%
F-MLP+tiny attn	12	9.9M	11.9%	30.3%	13.7%	12.8%
C-MLP+tiny attn	12	8.8M	11.7%	**29.9**%	**13.5**%	12.9%
TS-MLP+tiny attn	12	8.7M	13.3%	32.9%	14.5%	14.0%
Transformer	18	16.2M	10.9%	27.8%	11.5%	11.1%
F-MLP	27	16.3M	13.9%	34.6%	15.7%	14.8%
C-MLP	36	16.5M	**9.6**%	27.0%	11.4%	**10.1**%
TS-MLP	36	16.3M	11.2%	32.7%	13.5%	12.7%
F-MLP+tiny attn	21	15.8M	10.4%	27.0%	11.9%	11.4%
C-MLP+tiny attn	27	17.3M	9.8%	**26.0**%	**11.0**%	10.4%
TS-MLP+tiny attn	27	16.3M	9.9%	26.2%	**11.0**%	10.3%
Transformer	36	30.4M	9.9%	25.8%	10.7%	9.7%
F-MLP	51	29.0M	13.1%	32.5%	16.6%	15.4%
C-MLP	72	31.1M	8.6%	25.0%	9.8%	8.8%
TS-MLP	72	30.5M	10.0%	30.6%	11.6%	10.8%
F-MLP+tiny attn	42	29.8M	9.0%	24.1%	10.5%	9.8%
C-MLP+tiny attn	51	31.0M	**7.9**%	**21.0**%	**9.3**%	**8.5**%
TS-MLP+tiny attn	51	30.6M	8.8%	23.5%	9.8%	9.1%

on Tedlium2 dev/test set, while using only $75.3\,\%$ of the parameters. TS-MLP, which can be said to be a special case of C-MLP, has the smallest number of parameters. TS-MLP has only $56.2\,\%$ of parameters of Transformer-based CTC while degrade WER by $2.7/7.8\,\%$ on Librispeech test-clean/test-other set and $3.7/3.9\,\%$ on Tedlium2 dev/test set.

4.4 Model Scaling Analysis

Table 1 shows performance when the models are scaled to have approximately the same number of parameters. We evaluate three model size corresponding to the number of parameters for Transformer with 9, 18, and 36 layers. Under most conditions, C-MLP shows the highest accuracy. When the number of layers is scaled up and deepened, C-MLP and TS-MLP show a large performance improvement compared to the other models. This may be because the number of layers of C-MLP and TS-MLP is larger than other models due to their smaller size. We conjecture that the larger number of layers allows better mixing between a wider range of locations, leading to improved representations.

4.5 Computational Cost Analysis

We investigate inference speed with randomly created 83-dimensional inputs with sequence lengths of 512, 1024, 2048, 4096, 8192. A sequence of length of 512 corresponds to about $5.1\,\mathrm{s}$ of audio. We use an 18-layer encoder for every method. We conduct our experiments with a batch size 1 on a single Intel® Xeon® Gold 6230 CPU @ 2.10GHz with 4 cores. The reported times are the average of ten measurements. We show the inference speed for each input sequence length in Figure 5 and show computational characteristics of each architecture in Table 3. N denotes the sequence length of input and D denotes the size of channel dimension. WE set hidden dimension of linear layer in a tiny self-attention module to half of the hidden dimension of linear layer in a self-attention. This is sufficient to reduce the cost of self-attention to a reasonable level. Therefore, as shown in Table 3, in a tiny self-attention, $4ND^2$, which corresponds to the computational complexity of the linear layer, becomes half $2ND^2$. From Table 3, we see that when sequence length N is small, the

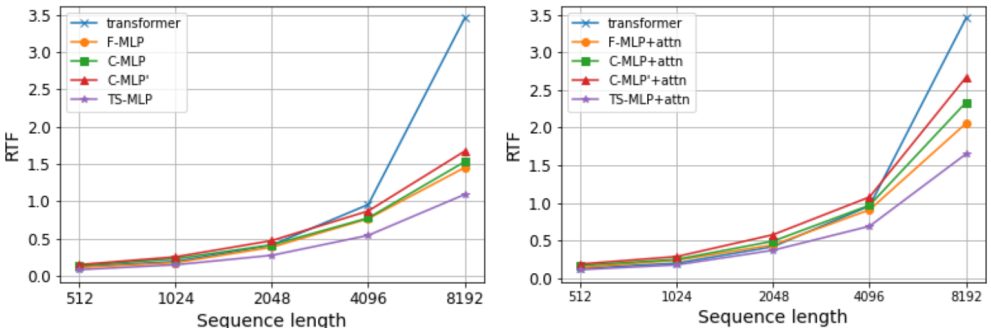

Figure 5: Comparison of inference speed for different sequence lengths (log scale)

Table 3: Computational cost of architectures. N is the sequence length and D is the size of channel dimension of a feature. l is the size of filters in Fourier Filter Unit. k is the kernel size of the convolution in Convolutional Gating Unit.

Method	**computational complexity**	**number of parameters**
linear projection	ND^2	D^2
self-attention	$4ND^2 + 2n^2D$	$4D^2$
Fourier Filter Unit	$ND\log_2 N + ND$	lD
Convolutional Gating Unit	kND^2	kD
Temporal Shift Gating Unit	$N + Nd$	-
tiny self-attention	$2ND^2 + 2N^2D$	$2D^2$

influence of the size of D on operation complexity is strong. As shown in Figure 5, inference speed of C-MLP and C-MLP′ is slower than Transformer CTC when N is small. On the other hand, the influence of the size of D on operation complexity is small as shown in Table 3. So, in Figure 5, we see that inference speed of Transformer-based CTC is slower than any of our architectures when N is large. TS-MLP, which can be said to be a special case of C-MLP, only performs a simple shift operation and element-wise multiplication. The TSGU has no parameters. Therefore, it achieves a fastest inference speed in Figure 5 regardless of the input sequence length. When combined with a tiny self-attention, the inference time increases, but the trend remains.

5 CONCLUSION

We proposed three new network architectures based on MLP-mixing for sequences of variable size. Each uses a different mechanism to mix information at long-range across the sequence. C-MLP relies on convolutional gating, TS-MLP on time-shift gating, and F-MLP simply on circular convolution without gating. Extensive experiments revealed that these MLP architectures are sufficient to outperform Transformer-based models for non-autoregressive ASR. Among the different models, C-MLP was the best, suggesting that gating is useful and necessary to squeeze out the best performance. When the proposed models alone were not sufficient, adding a tiny self-attention layer was enough to bridge the gap, while keeping the parameter count low. However, this was not always necessary when matching the number of parameters of the different networks. We thus conclude that all three proposed MLP-like architectures are not only suitable for ASR, but also highly practical due to their simplicity and good performance. In the future, we will explore their application to other tasks such as natural language processing or acoustic event detection.

REFERENCES

Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In

Proceedings of International Conference on Machine Learning (ICML), pp. 369–376. PMLR, 2006.

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. In *Proceedings of Interspeech 2020*, pp. 5036–5040, 2020.

Shigeki Karita, Nanxin Chen, Tomoki Hayashi, Takaaki Hori, Hirofumi Inaguma, Ziyan Jiang, Masao Someki, Nelson Enrique Yalta Soplin, Ryuichi Yamamoto, Xiaofei Wang, et al. A comparative study on transformer vs rnn in speech applications. In *Proceedings of IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pp. 449–456. IEEE, 2019a.

Shigeki Karita, Nelson Enrique Yalta Soplin, Shinji Watanabe, Marc Delcroix, Atsunori Ogawa, and Tomohiro Nakatani. Improving transformer-based end-to-end speech recognition with connectionist temporal classification and language model integration. In *Proceedings of Interspeech 2019*, pp. 1408–1412, 2019b.

Tom Ko, Vijayaditya Peddinti, Daniel Povey, and Sanjeev Khudanpur. Audio augmentation for speech recognition. In *Proceedings of Interspeech 2015*, 2015.

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 (EMNLP)*, pp. 66–71, 2018.

Jaesong Lee and Shinji Watanabe. Intermediate loss regularization for CTC-based speech recognition. In *Proceedings of International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2021.

James Lee-Thorp, Joshua Ainslie, Ilya Eckstein, and Santiago Ontanon. Fnet: Mixing tokens with fourier transforms. *arXiv preprint arXiv:2105.03824*, 2021.

Hanxiao Liu, Zihang Dai, David R So, and Quoc V Le. Pay attention to mlps. *arXiv preprint arXiv:2105.08050*, 2021.

Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur. Librispeech: an asr corpus based on public domain audio books. In *Proceedings of International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 5206–5210, 2015.

Daniel S Park, William Chan, Yu Zhang, Chung-Cheng Chiu, Barret Zoph, Ekin D Cubuk, and Quoc V Le. Specaugment: A simple data augmentation method for automatic speech recognition. In *Proceedings of Interspeech 2019*, pp. 2613–2617, 2019.

Yongming Rao, Wenliang Zhao, Zheng Zhu, Jiwen Lu, and Jie Zhou. Global filter networks for image classification. *arXiv preprint arXiv:2107.00645*, 2021.

Anthony Rousseau, Paul Deléglise, Yannick Esteve, et al. Enhancing the TED-LIUM corpus with selected data for language modeling and more TED talks. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC)*, pp. 3935–3939, 2014.

Ilya Tolstikhin, Neil Houlsby, Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Thomas Unterthiner, Jessica Yung, Daniel Keysers, Jakob Uszkoreit, Mario Lucic, et al. Mlp-mixer: An all-mlp architecture for vision. *arXiv preprint arXiv:2105.01601*, 2021.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 5998–6008, 2017.

Shinji Watanabe, Takaaki Hori, Shigeki Karita, Tomoki Hayashi, Jiro Nishitoba, Yuya Unno, Nelson-Enrique Yalta Soplin, Jahn Heymann, Matthew Wiesner, Nanxin Chen, et al. Espnet: End-to-end speech processing toolkit. In *Proceedings of Interspeech 2018*, pp. 2207–2211, 2018.

Tan Yu, Xu Li, Yunfeng Cai, Mingming Sun, and Ping Li. S^2-mlp: Spatial-shift mlp architecture for vision. *arXiv preprint arXiv:2106.07477*, 2021a.

Tan Yu, Xu Li, Yunfeng Cai, Mingming Sun, and Ping Li. S^2-mlpv2: Improved spatial-shift mlp architecture for vision. *arXiv preprint arXiv:2106.07477*, 2021b.