

EFFECTIVE LOW-COST TIME-DOMAIN AUDIO SEPARATION USING GLOBALLY ATTENTIVE LOCALLY RECURRENT NETWORKS

Max W. Y. Lam¹, Jun Wang¹, Dan Su¹, Dong Yu²

¹Tencent AI Lab, Shenzhen, China

²Tencent AI Lab, Bellevue WA, USA

ABSTRACT

Recent research on the time-domain audio separation networks (TasNets) has brought great success to speech separation. Nevertheless, conventional TasNets struggle to satisfy the memory and latency constraints in industrial applications. In this regard, we design a low-cost high-performance architecture, namely, globally attentive locally recurrent (GALR) network. Alike the dual-path RNN (DPRNN), we first split a feature sequence into 2D segments and then process the sequence along both the intra- and inter-segment dimensions. Our main innovation lies in that, on top of features recurrently processed along the inter-segment dimensions, GALR applies a self-attention mechanism to the sequence along the inter-segment dimension, which aggregates context-aware information and also enables parallelization. Our experiments suggest that GALR is a notably more effective network than the prior work. On one hand, with only 1.5M parameters, it has achieved comparable separation performance at a much lower cost with 36.1% less runtime memory and 49.4% fewer computational operations, relative to the DPRNN. On the other hand, in a comparable model size with DPRNN, GALR has consistently outperformed DPRNN in three datasets, in particular, with a substantial margin of 2.4dB absolute improvement of SI-SNRi in the benchmark WSJ0-2mix task.

Index Terms— speech separation, TasNet, low-cost, multi-head attention

1. INTRODUCTION

Audio separation is a fundamental problem in signal processing, and the most typical problem is called “cocktail party problem” [1] including multi-talker speech separation, overlapped speech-music separation, etc. Recent advances in deep learning models [2, 3, 4, 5] have drastically advanced state-of-the-art speech separation performances on several benchmark datasets. Currently, one outstanding category of the best-performing solutions is based on the time-domain audio separation network (TasNet) [6], which takes mixture waveforms as inputs and directly reconstruct sources by computing time-domain loss with permutation invariant training (PIT) [7, 8]. In particular, there were several types of TasNets: the initially proposed bi-directional long short-term memory (Bi-LSTM) based TasNet [6], the time convolutional network (TCN) based Conv-TasNet [9, 10], the dual-path recurrent neural network (DPRNN) [2] and the recently proposed dual-path Transformer network (DPTNet) [11].

These TasNet-based prior work has proven that a smaller window size improves the separation performance at the cost of a significantly longer 1-D feature sequence [2, 11]. To provide a more concrete illustration, we take a 4-second 16Hz sample rate waveform input as an example in Fig. 1, where the resultant feature sequence that a TasNet (with a window size of 2 samples and hop size of 1

sample) needs to model would be as long as 64000. Learning such long-term sequential dependency poses special challenges to various conventional sequential modeling networks, including attention models [12, 13], CNNs [14, 10], and RNNs (e.g., LSTMs [15] and GRUs [16]), each with respective difficulties as discussed below.

Attention models [17] has superiority in learning context-aware long-term dependencies, e.g., in BERT [18] for natural language processing tasks. Most recently, a series of research has also attempted to apply self-attention in speech signal processing, but generally to remarkably shorter feature sequences than a raw input, e.g., frame-level acoustic features for speech recognition [19, 20, 21, 22], layer features in a U-Net architecture [12] or short-time Fourier transform (STFT) features for speech enhancement [13]. Nevertheless, attention models have been hardly applied to time-domain source separation tasks as we discussed above because its complexity and memory consumption per layer is quadratic to the sequence length and become unacceptable for very long sequential modeling. 1-D CNNs with fixed receptive fields that are smaller than the very long sequence length, unlike RNNs that have dynamic receptive fields, are not able to fully utilize the sequence-level dependency [9]. RNNs are also limited by its nature of recursively processing and memorizing context [23, 24, 25]. To mitigate the long sequence modeling problem for RNNs, Luo et al. [2] introduced DPRNN, in which the long signal sequence is divided into shorter segments and interleave two RNNs, an inter-segment RNN and an inter-segment RNN, for local and global modeling, respectively.

To provide a better panorama about how a sequential context a TasNet (e.g., DPRNN) is dealing with looks, we plot the 4s waveform mixture in the upper in Fig. 1, where one of the two overlapping utterances, saying “Settle, no matter how, but settle”, has been marked in red. Under the dual-path setting, it is segmented into 512 segments, each with length 250. The lower plot zooms in on the 385th segment to show the details inside a segment around the lateral phoneme of // in the second “settle”. As we can see, high temporal correlation, acoustic signal structure, and continuities occur in inter-segment sequences, whilst strong discontinuities occur in inter-segment sequences. As revealed by Khandelwal et al. [23], RNNs are far more sensitive to the nearby elements than to the distant ones, and the model is capable of using about 200 tokens of context on average, but sharply distinguishes nearby context (recent 50 tokens) from the distant history. This suggests that RNNs are ideally suited for inter-segment modeling, but not necessarily so for inter-segment modeling. Moreover, Ravanelli et al. [24] discovered that RNNs reset the stored memory to avoid bias towards an “unrelated” history. However, unlike language modeling where information could be safely discarded when moving from one text to another semantically unrelated text, we argue that for specific tasks such as audio separation, faraway memory could potentially be important. For example, as shown in Fig. 1, the first “settle”

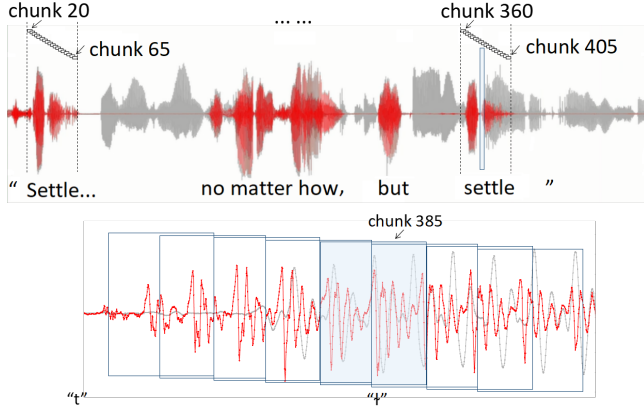


Fig. 1: Upper: a 4s raw waveform mixture of two overlapping utterances; Lower: zooming in on the 385th segment around the lateral phoneme of //.

(20th - 65th segment), which is very distant from the second “settle” (360th - 405th segment), could be more useful than nearby elements. In contrast, one strength of attention mechanisms over RNNs lies in a fully connected sequence processing strategy, where every element is connected to other elements in a sequence via a direct path without any recursively processing, memorizing reset, or update mechanisms like RNNs. Given the above example, for the second “settle”, a self-attention model would have readily placed more importance to the first “settle”, despite the faraway context beyond 200 segments.

Motivated by the above observation, our work revises TasNets and DPRNN to better address long-range context modeling for audio separation, leading to a lower-cost, higher-performance structure called globally attentive locally recurrent (GALR) network. We resort to the self-attention mechanism [17] for inter-segment computation to model context-aware global dependencies. Meanwhile, we keep using RNNs for modeling local dependencies at the lower inter-segment context level, e.g., signal continuity, signal structure, etc, which are inherently important for waveform reconstruction.

The contribution of this paper is four-fold:

- To the best of our knowledge, this is the first work that jointly leverages the attention and recurrent mechanisms in an alternate and iterative manner, and most importantly, allows the system to take advantage of both techniques, which are complementary at modeling global long-range context and local detail dependencies, respectively.
- Our work elegantly solves the critical bottleneck of self-attention networks due to unacceptable computational and memory cost for modeling very long sequences, as we can control the global sequence length via the dual-path structure [2]. Related similar work includes R-Transformer [25] and DPTNet [11]. Nevertheless, R-Transformer has the attention mechanism at the middle-level directly applied to the whole sequence, thus its computational cost is still quadratic to the sequence length, which hinders its application to TasNet; DPTNet also applied a dual-path setting like DPRNN to a Transformer-based architecture, but both inter-segment and inter-segment sequences are processed by a combination of attention model and RNN, leading to additional computational cost much heavier than DPRNN and our proposed GALR, which we will discuss in more details in Section 3.2.4.
- The proposed GALR model has a significantly lower cost with 42.3% reduction in model-size and with a 36.1% reduction in run-time memory cost and a 49.4% reduction in computational opera-

tions, while achieving comparable or better performance comparing to DPRNN. Moreover, unlike RNNs and DPRNN, the global attentive structure can further reduce training and inference time via parallelization, as the attention can be computed for all segments in parallel and allows parallel computation for aggregating information across segments over long audio sequences.

- Finally, results show consistently higher performance over DPRNN in terms of SI-SNR_i and SDR_i across three different datasets, while still maintaining a lower cost.

2. MODEL DESIGN

This section presents our proposed globally attentive locally recurrent (GALR) network. Fig. 2 shows the inner machinery of GALR, of which the core processing component is a stack of GALR blocks. Each GALR block contains two modeling perspectives. The first modeling perspective is responsible for modeling the local structures of input signals recurrently; the second modeling perspective aims at capturing global dependencies with the multi-head self-attention mechanism. Next, we describe each part in detail.

2.1. Encoding Raw Signals

2.1.1. Encoder

In a TasNet-based separation system, an input mixture signal is represented as I half-overlapping frames, denoted by $\mathbf{x}_1, \dots, \mathbf{x}_I \in \mathbb{R}^M$, where M denotes the window length. Analogous to the short-time Fourier transform (STFT), we non-linearly transform each frame \mathbf{x}_i into a D -dimensional feature vector $\tilde{\mathbf{x}}_i \in \mathbb{R}^D$ using a 1D gated convolutional layer:

$$\tilde{\mathbf{x}}_i = \text{ReLU}(\mathbf{U} * \mathbf{x}_i), \quad (1)$$

where $*$ denotes the 1D convolution operation, $\mathbf{U} \in \mathbb{R}^{D \times M}$ contains D vectors (encoder basis functions) with length M each, and $\text{ReLU}(\cdot)$ is the rectified linear unit used in [6, 2, 10] to ensure the non-negativity.

2.1.2. Segmentation

Given an encoded signal input $\tilde{\mathbf{X}} \in \mathbb{R}^{D \times I}$, the segmentation module splits $\tilde{\mathbf{X}}$ into S half-overlapping segments each of length K . The first and last segments are padded with zeros to create $S = \lceil 2I/K \rceil + 1$ equal-size segments: $\mathbf{S}_s \in \mathbb{R}^{D \times K}$ for $s = 1, \dots, S$. These segments are packed into a 3D tensor, denoted by $\mathbf{T} \in \mathbb{R}^{D \times S \times K}$. Note that the size of each segment K is a hyperparameter that affects the number of segments and can be used to control the scale of the locality.

2.2. GALR Blocks

The input 3D tensor \mathbf{T} is then passed to a stack of N GALR blocks, as shown in Fig. 2, which is designed to decouple the mixture signal by alternating local and global sequence modeling. Each GALR block returns a 3D tensor with the same dimensionality as its input. We denote the input for block $n = 1, \dots, N$ as $\mathbf{T}^{(n)} \in \mathbb{R}^{D \times S \times K}$, where $\mathbf{T}^{(1)} = \mathbf{T}$. As shown on the right of Fig. 2, a GALR block is composed of two phases of computation, a locally recurrent layer and a globally attentive layer, respectively corresponding to inter-segment processing and inter-segment processing. Each of which is described in detail below.

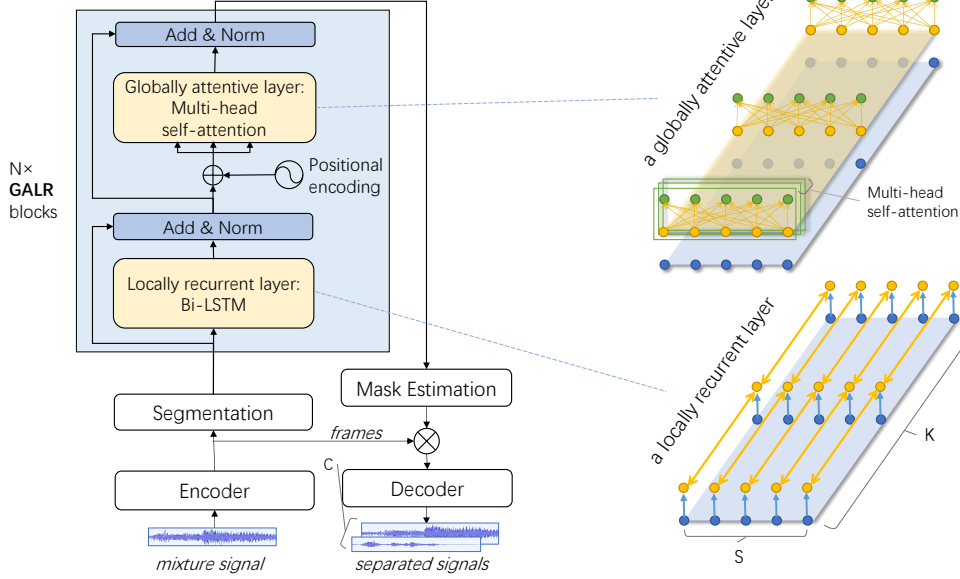


Fig. 2: Left: the overall architecture of our GALR network. Right: detailed illustration about how the intra- and inter-segment sequences are processed in the locally recurrent layer (lower right) and the globally attentive layer (upper right) inside each GALR block, respectively.

2.2.1. Locally Recurrent Model

A recurrent model is adopted to model the local information of the input sequence upon segmentation. To model such short-term dependencies within each segment, we employ a bi-directional LSTM of H hidden nodes:

$$\mathbf{L}^{(n)} = \left[\mathbf{R}^{(n)} f_{\text{Bi-LSTM}}^{(n)} \left(\mathbf{T}^{(n)}[:, s, :] \right) + \mathbf{Y}^{(n)}, s = 1, \dots, S \right], \quad (2)$$

where $\mathbf{R}^{(n)} \in \mathbb{R}^{D \times 2H}$ and $\mathbf{Y}^{(n)} \in \mathbb{R}^{D \times 1}$ form a linear layer, $\mathbf{L}^{(n)} \in \mathbb{R}^{D \times S \times K}$ is the output of the Bi-LSTM $f_{\text{Bi-LSTM}}^{(n)}(\cdot)$, and $\mathbf{T}^{(n)}[:, s, :] \in \mathbb{R}^{D \times K}$ refers to the local sequence within the s^{th} segment. The output of this locally recurrent model then goes through a layer normalization operation $\text{LN}(\cdot)$ with a residual connection to the block's input, which in practice is critical for model regularization and training acceleration [2]:

$$\hat{\mathbf{L}}^{(n)} = \text{LN}(\mathbf{L}^{(n)}) + \mathbf{T}^{(n)}. \quad (3)$$

2.2.2. Globally Attentive Model

We build a globally attentive model on top of the locally recurrent model to capture the long-term dependencies. Recent works in the speech community [19, 26, 27] have unveiled the extraordinary performance of attention mechanisms in learning long-term global dependencies [17]. Here in our case, the multi-head attention mechanism especially becomes the perfect fit due to three reasons. Firstly, the inherent computational burden of attention models becomes less-problematic since we now can control the sequence length by changing the window length of segmentation. Secondly, global dependencies across segments are directly modeled without needing to memorize segments one by one as in RNNs. Thirdly, given that the input is composed of different sources, it is sensible to use multiple attention schemes (a.k.a. heads) on the whole sequence.

Before applying the attention mechanism, the output of locally recurrent model first goes through the following:

$$\mathbf{G}^{(n)} = \text{LN}_D(\hat{\mathbf{L}}^{(n)}) + \mathbf{P} \quad (4)$$

where $\text{LN}_D(\cdot)$ denotes the layer normalization applied only along the feature dimension D , \mathbf{P} denotes the positional encoding matrix introduced in [17]. For global sequence modeling, we consider the sequence of frames across all segments, i.e., $\mathbf{G}_k^{(n)} \triangleq [\mathbf{G}^{(n)}[:, s, k], s = 1, \dots, S]$. In order to create J heads, we linearly map $\mathbf{G}_k^{(n)}$ into J different forms of query, key, and value matrices:

$$\mathbf{Q}_{k,j}^{(n)} = \mathbf{W}_{\text{query},j}^{(n)} (\mathbf{W}_{\text{query}}^{(n)} \mathbf{G}_k^{(n)} + \mathbf{b}_{\text{query}}^{(n)}) \quad (5)$$

$$\mathbf{K}_{k,j}^{(n)} = \mathbf{W}_{\text{key},j}^{(n)} (\mathbf{W}_{\text{key}}^{(n)} \mathbf{G}_k^{(n)} + \mathbf{b}_{\text{key}}^{(n)}) \quad (6)$$

$$\mathbf{V}_{k,j}^{(n)} = \mathbf{W}_{\text{value},j}^{(n)} (\mathbf{W}_{\text{value}}^{(n)} \mathbf{G}_k^{(n)} + \mathbf{b}_{\text{value}}^{(n)}) \quad (7)$$

for $k = 1, \dots, K, j = 1, \dots, J$, where $\mathbf{W}_{\text{query}}^{(n)}, \mathbf{W}_{\text{key}}^{(n)}, \mathbf{W}_{\text{value}}^{(n)} \in \mathbb{R}^{D \times D}$, $\mathbf{b}_{\text{query}}^{(n)}, \mathbf{b}_{\text{key}}^{(n)}, \mathbf{b}_{\text{value}}^{(n)} \in \mathbb{R}^{D \times 1}$ and $\mathbf{W}_{\text{query},j}^{(n)}, \mathbf{W}_{\text{key},j}^{(n)}, \mathbf{W}_{\text{value},j}^{(n)} \in \mathbb{R}^{D/J \times D}$. Note that the attention parameters are not dependent on k , which means that we tie the attention weights for all K sequences, i.e., $[\mathbf{G}_k^{(n)}, k = 1, \dots, K]$. Tying weights is sensible here because the cross-segment sequences formed within a relatively small segment size ought to have very high correlations.

Given the query, key, and value inputs, we then compute the scaled dot-product attention following [17] in J heads:

$$\mathbf{A}_{k,j}^{(n)} = \text{Softmax} \left(\frac{\mathbf{Q}_{k,j}^{(n)\top} \mathbf{K}_{k,j}^{(n)}}{\sqrt{D/J}} \right) \mathbf{V}_{k,j}^{(n)}. \quad (8)$$

Next, the attention matrices computed at different heads are combined using an affine transformation after concatenating the matrices:

$$\mathbf{A}_k^{(n)} = \mathbf{W}_{\text{attn}}^{(n)} \text{Concat} \left(\mathbf{A}_{k,1}^{(n)}, \dots, \mathbf{A}_{k,J}^{(n)} \right), \quad (9)$$

where $\mathbf{W}_{\text{attn}}^{(n)} \in \mathbb{R}^{D \times D}$ is the weight matrix for the heads. The attention outputs are then concatenated back to a 3D tensor, i.e.,

$\mathbf{A}^{(n)} = [\mathbf{A}_k^{(n)}, k = 1, \dots, K]$. Given the attention output, we employ a sub-layer connection with reference to the well-known Transformer model [17] given by

$$\hat{\mathbf{G}}^{(n)} = \text{LN}(\mathbf{G}^{(n)} + \text{Dropout}(\mathbf{A}^{(n)})), \quad (10)$$

where $\text{Dropout}(\cdot)$ denotes the dropout regularization [28] operation. Finally, the GALR block outputs a residual sum between the local model output and global model output:

$$\mathbf{T}^{(n+1)} = \hat{\mathbf{G}}^{(n)} + \hat{\mathbf{L}}^{(n)}, \quad (11)$$

which defines a recurrence relation between N GALR blocks.

2.2.3. Low-dimension Segment Representation

Note that the attention mechanism is repeated K times in Eq. (8-9), we observe that our globally attentive model can use a low-dimension trick to reduce memory and floating-point operations, while maintaining the performance. Due to high correlations between the cross-segment sequences, we indeed can approximate the global dependencies with a down-sampled number of sequences. In this regard, we employ two affine transformations $C_{\text{map}}(\cdot)$ and $C_{\text{inv}}(\cdot)$ for mapping K dimensions into Q dimensions and inversely mapping Q dimensions back to K dimensions, respectively, where $Q \ll K$. Mathematically, we only need to re-write Eq. (4) and Eq. (11) as

$$\mathbf{G}^{(n)} = \text{LN}_D(C_{\text{map}}(\hat{\mathbf{L}}^{(n)})) + \mathbf{P}, \quad (12)$$

$$\mathbf{T}^{(n+1)} = C_{\text{inv}}(\mathbf{G}^{(n)}) + \hat{\mathbf{L}}^{(n)}, \quad (13)$$

where $C_{\text{map}}(\cdot)$ and $C_{\text{inv}}(\cdot)$ contain affine mapping parameters in shapes $Q \times (K + 1)$ and $K \times (Q + 1)$, respectively.

2.3. Signals Reconstruction

2.3.1. Mask Estimation

After N consecutive GALR blocks, we obtain a representation of the mixture signal that facilitates the separation of C sources. We then use a 2D convolutional layer to transform this 3D representation into C 3D tensors. Then, we transform each of the C 3D tensors back to a matrix $\mathbf{S}_c \in \mathbb{R}^{D \times L}$ for $c = 1, \dots, C$ by applying the OverlapAdd method described in [2]. After that, we have a beam-forming procedure [29] that applies two 1D gated convolution layers to each of the C matrices:

$$\hat{\mathbf{S}}_c = \tanh(\mathbf{U}_{\text{tanh}} * \mathbf{S}_c) \odot \sigma(\mathbf{U}_{\text{sigmoid}} * \mathbf{S}_c), \quad (14)$$

where \odot denotes element-wise multiplication, $\sigma(\cdot)$ is the Sigmoid function, and $\mathbf{U}_{\text{tanh}} \in \mathbb{R}^{D \times D}$ and $\mathbf{U}_{\text{sigmoid}} \in \mathbb{R}^{D \times D}$ are two parameter matrices in the 1D gated convolution. The Tanh and Sigmoid functions here act as the beam-forming filters.

To produce a mask matrix for each source, the final step is to employ a resilient linear (ReLU) mask function

$$\mathbf{M}_c = \text{ReLU}(\mathbf{U}_{\text{relu}} * \hat{\mathbf{S}}_c), \quad (15)$$

where $\mathbf{U}_{\text{relu}} \in \mathbb{R}^{D \times D}$ is a 1D convolution for learning mask.

2.3.2. Decoder for Waveform Reconstruction

The c^{th} estimated mask is applied back to the initially encoded mixture $\tilde{\mathbf{X}} = [\tilde{\mathbf{x}}_1, \dots, \tilde{\mathbf{x}}_L]$ to reconstruct source c :

$$\hat{\mathbf{s}}_c = \text{OverlapAdd}(\mathbf{B}(\tilde{\mathbf{X}} \odot \mathbf{M}_c)), \quad (16)$$

where $\mathbf{B} \in \mathbb{R}^{M \times D}$ is a matrix containing the basis signals with each column corresponding to a 1D filter.

2.4. Permutation Invariant Training

In a standard training framework, given a speech separation model f_θ with a set of parameters denoted by θ , a loss function $\mathcal{L}(f_\theta(\mathbf{x}), \mathbf{y})$ is used to penalize the divergence between the predicted outputs $f_\theta(\mathbf{x}) = \{\hat{\mathbf{s}}_1, \dots, \hat{\mathbf{s}}_C\}$ and the clean sources $\mathbf{y} = \{\mathbf{s}_1, \dots, \mathbf{s}_C\}$. As an end-to-end network, our proposed GALR model outputs waveforms of the estimated clean signals so that we can directly use the scale-invariant source-to-noise ratio (SI-SNR) [6] as our maximization objection with permutation invariant training (PIT) [7, 8]:

$$\mathcal{L}_{\text{SI-SNR}}(f_\theta(\mathbf{x}), \mathbf{y}) = -10 \log_{10} \frac{\|\Pi_{\mathbf{s}}(\hat{\mathbf{s}})\|_2^2}{\|\hat{\mathbf{s}} - \Pi_{\mathbf{s}}(\hat{\mathbf{s}})\|_2^2}, \quad (17)$$

where $\Pi_{\mathbf{a}}(\mathbf{b}) = \frac{\mathbf{a}^\top \mathbf{b}}{\|\mathbf{a}\|_2} \mathbf{a}$ is the projection of \mathbf{b} onto \mathbf{a} . SI-SNR has also been used in many end-to-end separation models [6, 10, 30, 2].

3. EVALUATION AND ANALYSIS

3.1. Experimental Setup

3.1.1. Data Preparation

We used three datasets for our experiments: (1) WSJ0-2mix, a two-speaker speech dataset [31, 32] that consists of 30 hours of training, 10 hours of validation, and 5 hours of evaluation data and is widely used as the benchmark in monaural speech separation [2, 3, 6, 10, 33, 34, 35], (2) Libri-2mix, a larger two-speaker audio mixture dataset generated from a publicly available English speech corpus Librispeech [36] that contains 982.1 hours of speech from 2484 speakers, and (3) WSJ0-music, a speech-music mixture audio dataset generated in [35]. All mixture audios were simulated by randomly combining utterances from different speakers or music clips at a sampling rate of 8 kHz with SNRs between 0 dB and 5 dB.

3.1.2. Model Setup

In our implementation, we used the setting of encoder-decoder modules in [6, 10] and the segmentation module described in [2]. In the middle part of the separation network, 6 consecutive blocks were used to model local and global sequences, i.e., $N = 6$. We fixed the number of hidden nodes (H) in Bi-LSTM to 128 as in [2]. The multi-head attention based global model was made of 8 heads, i.e., $J = 8$. In each attention layer, the dropout rate was set to 0.1. Regarding other model hyperparameters, we varied the number of filters (D), the window length (M), and the segment size (K) as shown in Table 3. Notably, when we set $D = 64$ as in [6, 10] the model size of GALR is much smaller than that in the previous works, therefore, we also tried $D = 128$ to obtain a comparable model size with DPRNN. Meanwhile, we implemented the most recently proposed DPTNet [11] as another reference.

It is worthwhile to explain why we omitted the configuration of $M = 2$ and $K = 250$, which corresponds to the highest SI-SNRi score in [2]. Although the authors in [2] reported that setting shorter window length leads to better SI-SNRi performance, the associated computational burden was not disclosed. Given a limited GPU memory, halving the window length resorts to halving the batch size and doubling the training time. We tried to run the highest scores under the setting of $M = 2$ for both DPRNN and GALR. However, building such systems in a small dataset like WSJ0-2mix costs more than ten days of training. For a realistic industrial development, we generally need to tackle 10-100 times larger datasets. Therefore, the corresponding training efficiency is unacceptable for most realistic

Table 1: Comparison of dual-path processing time complexities for different block types.

Block	Local-path Complexity	Global-path Complexity	Maximum Path Length [17]
DPTNet [11]	$\mathcal{O}(KSH^2 + K^2SD)$	$\mathcal{O}(KSH^2 + KS^2D)$	$\mathcal{O}(S + K)$
DPRNN [2]	$\mathcal{O}(KSH^2)$	$\mathcal{O}(KSH^2)$	$\mathcal{O}(S + K)$
GALR	$\mathcal{O}(KSH^2)$	$\mathcal{O}(QS^2D)$	$\mathcal{O}(K)$

scenarios, not to mention that the high latency at inference time is also problematic for system deployment.

3.1.3. Training Details

All the models were trained on 8 NVIDIA Tesla M40 GPU devices using PyTorch [37] for fair comparisons. We found that the performances of all separation models deteriorated as we used 8 GPUs in place of 1 GPU. A similar observation has been obtained in [38, 39], where this multi-GPU training approach is termed model averaging. Note that it is impractical to use a single GPU for model training with its unacceptably long training time. For a fair part-to-part comparison, we report the performances of GALR under the same 8-GPU training condition, though the SI-SNRi can be further improved in the case of using fewer GPUs.

For the benchmark WSJ0-2mix separation task, we referred to the training protocol in [2], where clipped 4-second waveforms were used for permutation invariant training [7] to minimize pairwise SI-SNR loss [6]. Concerning optimization, we used Adam [40] optimizer with an initial learning rate of $1e^{-3}$ and a weight decaying rate of $1e^{-6}$. The learning rate was exponentially decayed at a rate of 0.96 for every two epochs. The training was considered converged when no lower validation loss can be observed in 10 consecutive epochs. A gradient clipping method was used to ensure the maximum l2-norm of each gradient is less than 5. All models were assessed in terms of SI-SNRi and SDRi [41].

3.2. Performance Analysis

3.2.1. Investigation of GALR Optimality

Table 2: SI-SNRi results of WSJ0-2mix when permuting Bi-LSTM and attention model in local and global modeling

Approach	Local Bi-LSTM	Local Attention
Global Bi-LSTM	15.9	12.3
Global Attention	17.0	14.6

First and foremost, we experimented on WSJ0-2mix to validate our hypothesis that the proposed GALR architecture is the optimal choice amongst while permuting recurrent and attention models for local and global sequence modeling. In this experiment, we chose the bi-directional LSTM as a representative recurrent model. As shown in Table 2, we obtained 4 distinctive SI-SNRi scores from 4 kinds of TasNet systems. Interestingly, we found two consistent patterns: (1) in local modeling, the recurrent model performed better than the attention model; (2) in global modeling, the attention model performed better than the recurrent model. Overall, the proposed GALR architecture (bottom-left) gave the best performance among the four architectures, which matches our expectations.

3.2.2. Time Complexity Analysis

Next, we compared the algorithmic complexities of all three models in Table 1, where we reported dual-path processing complexities and

the maximum path length (MPL) [17] that was needed to connect any two positions in the signal sequence in Big-O notation. Considering the global-path processing complexity, the cost of DPTNet was about the sum of the costs of both DPRNN and GALR. Notably, for very long input sequences, e.g., in the case of small window length, we needed to use a larger K to improve the computational performance of both models. By introducing low-dimensional mapping with $Q \ll K$, we found that GALR could significantly relieve the computational burden carried by a large K , as reported in terms of actual FLOPs in Table 3. Besides, with regard to MPL, amongst the three models, only GALR managed to connect all positions with $\mathcal{O}(K)$, whereas both DPRNN and DPTNet required $\mathcal{O}(S + K)$ sequential operations. As discussed in [42], the shorter the MPL, the easier it was to learn long-term dependencies, which echoed the theoretical advantage of GALR.

3.2.3. Visualizing Global Attention with Multiple Heads

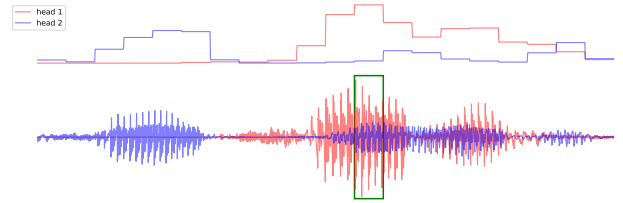


Fig. 3: An example given by a GALR model trained on WSJ0-2mix. The clean speech signals unseen by the model are shown in red and blue. The two graphs above the signals denotes the softmax values of two selected heads attending on the target segment in green frame.

We were also interested in understanding how the multi-head self-attention works in speech separation. To reason about the physical meaning of our global attention mechanism, we examined the values of the softmax matrices defined in Eq. 8 computed in different heads. The softmax values were plotted against the time axis with respect to the source signals, as shown in Fig. 3. Interestingly, we observed two heads where the attention values were related to the energies of the two clean speeches. This explained how multi-head self-attention mechanism worked in GALR – the attention matrices of different heads were combined over stacks of GALR blocks to output an easily separable representation for the mixture signal. This attention behavior was also mimicking how humans conceptually following the speech of one talker with regards to its volume.

3.2.4. Performances in Benchmark WSJ0-2mix

As for the architecture for TasNet, we compared the result of our GALR model to the state-of-the-art DPRNN [2] and DPTNet [11]. Considering only the lightest model with $M = 16$, we found that DPTNet costs 152% more time and 225% more memory than our proposed GALR. Considering this remarkable surge of memory consumption, we noted that DPTNet might be not practically preferable for most realistic industrial tasks; therefore we only compared

Table 3: Performances of DPRNN and our proposed GALR in WSJ0-2mix test set with different configurations.

TasNet	D	M	K	Q	Size	SI-SNRi	SDRi	Runtime Memory	GFLOPs
DPTNet [11]	64	16	100	-	2.8M	15.5	15.7	419 MiB	12.6
DPRNN [2]	64	16	100	-	2.6M	15.9	16.2	231 MiB	10.7
	64	8	150	-		17.0	17.3	456 MiB	22.2
	64	4	200	-		17.9	18.1	929 MiB	42.3
GALR	64	16	100	32	1.5M	16.2	16.5	161 MiB	5.6
	64	8	150	16		17.1	17.4	309 MiB	11.5
	64	4	200	8		17.7	17.9	594 MiB	21.4
	128	16	100	32	2.3M	17.0	17.3	186 MiB	8.3
	128	8	150	16		18.7	18.9	363 MiB	16.5
	128	4	200	8		20.3	20.5	730 MiB	30.8

GALR with DPRNN in the following tasks. We replicated the experiment in [2] with the same configurations of window length and segment size. The results are shown in Table 3. Besides the standard separation measure SI-SNRi, we also analyzed the runtime cost of each model for processing a 1s mixture input in terms of memory measured in GPU and floating-point operations (FLOPs) approximated with a third-party module,¹ which represents the model efficiency. On the one hand, the GALR with a model size comparable to DPRNN consistently gave superior SI-SNRi performances over DPRNN under the same configurations of window length and segment size. On the other hand, the smaller GALR attained comparable or better separation performances while requiring only 57.3% parameters and reducing up to 36.1% runtime memory and 49.4% computational operations.

3.2.5. Performances in Large-scale Libri-2mix

Table 4: Performances of DPRNN and our proposed GALR in Libri-2mix

TasNet	Size	SI-SNRi	SDRi
DPRNN [2]	2.6M	12.0	12.5
GALR	2.3M	12.2	12.7

To validate the consistency of GALR’s improvements over DPRNN, we experimented on a larger dataset Libri-2mix. In Libri-2mix, we split the Librispeech [36] corpus into (1) a training set containing 12055 utterances (5 utterances per speaker) drawn from 2411 speakers, (2) a validation set containing another 7233 utterances (5 utterances per speaker) drawn from the same 2411 speakers, and 2411 speakers (3) a test set containing 4380 utterances (60 utterances per speaker) drawn from 73 speakers. Note that the separation task in Libri-2mix was much more challenging than that in WSJ0-2mix because of not only increasing training speakers from 101 to 2411 but also the limitation of only five sampled utterances per speaker for training. Despite the increased separation difficulty, we conceived that the separation scenario became more realistic as it was often hard to collect many clean utterances from the user in real-world applications.

In this large-scale separation dataset, to strike a balance between the training speed and the separation performance, we decided to use a configuration of $M = 8$ and $K = 150$ to train both DPRNN and GALR. For a fair comparison in terms of the number of parameters, we only trained the GALR with 2.3M parameters to be comparable to DPRNN. The separation results of DPRNN and GALR were

¹<https://github.com/sovrasov/flops-counter.pytorch>

summarized and shown in Table 4. We observed that GALR maintained its advantage in speed and memory over DPRNN and meanwhile achieved 0.2dB absolute improvements in SI-SNRi and SDRi. The consistent results suggest that GALR keeps performing low-cost separation without sacrificing separation efficacy.

3.2.6. Performances in Separating Speech-Music Mixture

Table 5: Performances of DPRNN and our proposed GALR in WSJ0-music

TasNet	Size	SI-SNRi	SDRi
DPRNN [2]	2.6M	14.5	14.8
GALR	2.3M	15.9	16.2

Besides two-speaker speech separation tasks, we were also interested in separating speech from a speech-music mixture. In this paper, we simulated the speech-music mixture using the speech from the WSJ0-2mix corpus and the music clips in [35]. The results were shown in Table 5. Comparing to the two-speaker separation, we found that GALR obtained a even greater superiority over DPRNN in the speech-music scenario, which is also very common in real-world conversation scenarios. In particular, there is a growing demand in the industry where the speech-music separation becomes critical for the deployment of many real-world signal processing systems, e.g., micro-video automatic captioning. Therefore, the consistent and larger superiority of GALR over DPRNN in the speech-music separation task is valuable for both conventional and emerging application deployments.

4. CONCLUSIONS

This paper introduces a novel architecture – globally attentive locally recurrent network (GALR), which combines the advantages of recurrent networks and attention mechanisms for effective, low-cost time-domain signal processing. We provide empirical evidences that the self-attention mechanism is a better candidate for modeling the long-range context sequence than the RNNs. Results across three different datasets also suggest the superiority of attention models over recurrent models in modeling global sequences, which leads to greater modeling power, reduced model size, and less runtime memory. We believe that a compact, low-cost, and effective separation system is more practical to the industry and will empower wider applications of speech separation for robust signal processing.

5. REFERENCES

- [1] E Colin Cherry, "Some experiments on the recognition of speech, with one and with two ears," *The Journal of the acoustical society of America*, vol. 25, no. 5, pp. 975–979, 1953.
- [2] Yi Luo, Zhuo Chen, and Takuya Yoshioka, "Dual-path rnn: efficient long sequence modeling for time-domain single-channel speech separation," *arXiv preprint arXiv:1910.06379*, 2019.
- [3] Yuzhou Liu and DeLiang Wang, "Divide and conquer: A deep casa approach to talker-independent monaural speaker separation," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 27, no. 12, pp. 2092–2102, 2019.
- [4] Giovanni Morrone, Sonia Bergamaschi, Luca Pasa, Luciano Fadiga, Vadim Tikhonoff, and Leonardo Badino, "Face landmark-based speaker-independent audio-visual speech enhancement in multi-talker environments," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 6900–6904.
- [5] Jianwei Yu, Shi-Xiong Zhang, Jian Wu, Shahram Ghorbani, Bo Wu, Shiyin Kang, Shansong Liu, Xunying Liu, Helen Meng, and Dong Yu, "Audio-visual recognition of overlapped speech for the Irs2 dataset," *arXiv preprint arXiv:2001.01656*, 2020.
- [6] Yi Luo and Nima Mesgarani, "Tasnet: time-domain audio separation network for real-time, single-channel speech separation," in *Proc. ICASSP*. IEEE, 2018, pp. 696–700.
- [7] Dong Yu, Morten Kolbæk, Zheng-Hua Tan, and Jesper Jensen, "Permutation invariant training of deep models for speaker-independent multi-talker speech separation," in *Proc. ICASSP*. IEEE, 2017, pp. 241–245.
- [8] Morten Kolbæk, Dong Yu, Zheng-Hua Tan, Jesper Jensen, Morten Kolbaek, Dong Yu, Zheng-Hua Tan, and Jesper Jensen, "Multitalker speech separation with utterance-level permutation invariant training of deep recurrent neural networks," *TASLP*, vol. 25, no. 10, pp. 1901–1913, 2017.
- [9] Shaojie Bai, J Zico Kolter, and Vladlen Koltun, "An empirical evaluation of generic convolutional and recurrent networks for sequence modeling," *arXiv preprint arXiv:1803.01271*, 2018.
- [10] Yi Luo and Nima Mesgarani, "Conv-tasnet: Surpassing ideal time-frequency magnitude masking for speech separation," *IEEE/ACM transactions on audio, speech, and language processing*, vol. 27, no. 8, pp. 1256–1266, 2019.
- [11] Jingjing Chen, Qirong Mao, and Dong Liu, "Dual-path transformer network: Direct context-aware modeling for end-to-end monaural speech separation," *arXiv preprint arXiv:2007.13975*, 2020.
- [12] Ritwik Giri, Umut Isik, and Arvinth Krishnaswamy, "Attention wave-u-net for speech enhancement," in *2019 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*. IEEE, 2019, pp. 249–253.
- [13] Jaeyoung Kim, Mostafa El-Khamy, and Jungwon Lee, "T-gsa: Transformer with gaussian-weighted self-attention for speech enhancement," *arXiv preprint arXiv:1910.06762*, 2019.
- [14] Daniel Stoller, Sebastian Ewert, and Simon Dixon, "Wave-u-net: A multi-scale neural network for end-to-end audio source separation," in *19th International Society for Music Information Retrieval Conference, ISMIR*, 2018.
- [15] Felix A Gers, Jürgen Schmidhuber, and Fred Cummins, "Learning to forget: Continual prediction with lstm," 1999.
- [16] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," *arXiv preprint arXiv:1412.3555*, 2014.
- [17] 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*, 2017, pp. 5998–6008.
- [18] 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.
- [19] Linhao Dong, Shuang Xu, and Bo Xu, "Speech-transformer: a no-recurrence sequence-to-sequence model for speech recognition," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 5884–5888.
- [20] Ngoc-Quan Pham, Thai-Son Nguyen, Jan Niehues, Markus Müller, Sebastian Stüker, and Alexander Waibel, "Very deep self-attention networks for end-to-end speech recognition," *arXiv preprint arXiv:1904.13377*, 2019.
- [21] Kyu J Han, Jing Huang, Yun Tang, Xiaodong He, and Bowen Zhou, "Multi-stride self-attention for speech recognition," in *INTERSPEECH*, 2019, pp. 2788–2792.
- [22] Haoneng Luo, Shiliang Zhang, Ming Lei, and Lei Xie, "Simplified self-attention for transformer-based end-to-end speech recognition," *arXiv preprint arXiv:2005.10463*, 2020.
- [23] Urvashi Khandelwal, He He, Peng Qi, and Dan Jurafsky, "Sharp nearby, fuzzy far away: How neural language models use context," *arXiv preprint arXiv:1805.04623*, 2018.
- [24] Mirco Ravanelli, Philemon Brakel, Maurizio Omologo, and Yoshua Bengio, "Light gated recurrent units for speech recognition," *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. 2, no. 2, pp. 92–102, 2018.
- [25] Zhiwei Wang, Yao Ma, Zitao Liu, and Jiliang Tang, "R-transformer: Recurrent neural network enhanced transformer," *arXiv preprint arXiv:1907.05572*, 2019.
- [26] Naihan Li, Shujie Liu, Yanqing Liu, Sheng Zhao, and Ming Liu, "Neural speech synthesis with transformer network," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2019, vol. 33, pp. 6706–6713.
- [27] Yi Ren, Yangjun Ruan, Xu Tan, Tao Qin, Sheng Zhao, Zhou Zhao, and Tie-Yan Liu, "Fastspeech: Fast, robust and controllable text to speech," in *Advances in Neural Information Processing Systems*, 2019, pp. 3165–3174.
- [28] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *The journal of machine learning research*, vol. 15, no. 1, pp. 1929–1958, 2014.
- [29] Yi Luo, Enea Ceolini, Cong Han, Shih-Chii Liu, and Nima Mesgarani, "Fasnet: Low-latency adaptive beamforming for multi-microphone audio processing," *arXiv preprint arXiv:1909.13387*, 2019.

- [30] Gene-Ping Yang, Chao-I Tuan, Hung-Yi Lee, and Lin-shan Lee, “Improved speech separation with time-and-frequency cross-domain joint embedding and clustering,” *arXiv preprint arXiv:1904.07845*, 2019.
- [31] J Garofalo, D David Graff, D Paul, and D Pallett, “Continuous speech recognition (csr-i) wall street journal (wsj0) news, complete. linguistic data consortium, philadelphia (1993),” .
- [32] John R Hershey, Zhuo Chen, Jonathan Le Roux, and Shinji Watanabe, “Deep clustering: Discriminative embeddings for segmentation and separation,” in *Proc. ICASSP. IEEE*, 2016, pp. 31–35.
- [33] Zhong-Qiu Wang, Jonathan Le Roux, DeLiang Wang, and John R Hershey, “End-to-end speech separation with unfolded iterative phase reconstruction,” *arXiv preprint arXiv:1804.10204*, 2018.
- [34] Liwen Zhang, Ziqiang Shi, Jiqing Han, Anyan Shi, and Ding Ma, “Furcanext: End-to-end monaural speech separation with dynamic gated dilated temporal convolutional networks,” in *International Conference on Multimedia Modeling*. Springer, 2020, pp. 653–665.
- [35] Max WY Lam, Jun Wang, Dan Su, and Dong Yu, “Mixup-breakdown: a consistency training method for improving generalization of speech separation models,” *Proc. ICASSP*, 2020.
- [36] Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur, “Librispeech: an asr corpus based on public domain audio books,” in *Proc. ICASSP. IEEE*, 2015, pp. 5206–5210.
- [37] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer, “Automatic differentiation in PyTorch,” in *NIPS Autodiff Workshop*, 2017.
- [38] Hang Su and Haoyu Chen, “Experiments on parallel training of deep neural network using model averaging,” *arXiv preprint arXiv:1507.01239*, 2015.
- [39] Kai Chen and Qiang Huo, “Scalable training of deep learning machines by incremental block training with intra-block parallel optimization and blockwise model-update filtering,” in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2016, pp. 5880–5884.
- [40] Diederik P Kingma and Jimmy Ba, “Adam: A method for stochastic optimization,” *arXiv preprint arXiv:1412.6980*, 2014.
- [41] Jonathan Le Roux, Scott Wisdom, Hakan Erdogan, and John R Hershey, “Sdr-half-baked or well done?,” in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 626–630.
- [42] Sepp Hochreiter, Yoshua Bengio, Paolo Frasconi, Jürgen Schmidhuber, et al., “Gradient flow in recurrent nets: the difficulty of learning long-term dependencies,” 2001.