AN EFFICIENT ENCODER-DECODER ARCHITECTURE WITH TOP-DOWN ATTENTION FOR SPEECH SEPARATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Deep neural networks have shown excellent prospects in speech separation tasks. However, obtaining good results while keeping a low model complexity remains challenging in real-world applications. In this paper, we provide a bio-inspired efficient encoder-decoder architecture by mimicking the brain's top-down attention, called TDANet, with decreased model complexity without sacrificing performance. The top-down attention in TDANet is extracted by the global attention (GA) module and the cascaded local attention (LA) layers. The GA module takes multi-scale acoustic features as input to extract global attention signal, which then modulates features of different scales by direct top-down connections. The LA layers use features of adjacent layers as input to extract the local attention signal, which is used to modulate the lateral input in a top-down manner. On three benchmark datasets, TDANet consistently achieved competitive separation performance to previous state-of-the-art (SOTA) methods with higher efficiency. Specifically, TDANet's multiply-accumulate operations (MACs) are only 5% of Sepformer, one of the previous SOTA models, and CPU inference time is only 10% of Sepformer. In addition, a large-size version of TDANet obtained SOTA results on three datasets, with MACs still only 10% of Sepformer and the CPU inference time only 24% of Sepformer. Our study suggests that top-down attention can be a more efficient strategy for speech separation.

1 INTRODUCTION

In cocktail parties, people's communications are inevitably disturbed by various sounds (Bronkhorst, 2015; Cherry, 1953), such as environmental noise and extraneous audio signals, potentially affecting the quality of communication. Humans can effortlessly perceive the speech signal of a target speaker in a cocktail party to improve the accuracy of speech recognition (Haykin & Chen, 2005). In speech processing field, the corresponding challenge is to separate different speakers' audios from the mixture audio, known as *speech separation*.

Due to rapid development of deep neural networks (DNNs), DNN-based speech separation methods have significantly improved (Luo & Mesgarani, 2019; Luo et al., 2020; Tzinis et al., 2020; Chen et al., 2020; Subakan et al., 2021; Hu et al., 2021; Li & Luo, 2022). As in natural language processing, the SOTA speech separation methods are now embracing increasingly complex models to achieve better separation performance, such as DPTNet (Chen et al., 2020) and Sepformer (Subakan et al., 2021). These models typically use multiple transformer layers (Vaswani et al., 2017) to capture longer contextual information, leading to a large number of parameters and high computational cost and having a hard time deploying to edge devices. We question whether such complexity is always needed in order to improve the separation performance.

Human brain has the ability to process large amounts of sensory information with extremely low energy consumption (Attwell & Laughlin, 2001; Howarth et al., 2012). We therefore resort to our brain for inspiration. Numerous neuroscience studies have suggested that in solving the cocktail part problem, the brain relies on a cognitive process called *top-down attention* (Wood & Cowan, 1995; Haykin & Chen, 2005; Fernández et al., 2015). It enables human to focus on task-relevant stimuli and ignore irrelevant distractions. Specifically, top-down attention modulates (enhance or

inhibit) cortical sensory responses to different sensory information (Gazzaley et al., 2005; Johnson & Zatorre, 2005). With neural modulation, the brain is able to focus on speech of interest and ignore others in a multi-speaker scenario (Mesgarani & Chang, 2012).

We note that encoder-decoder speech separation networks (e.g., SuDORM-RF (Tzinis et al., 2020) and A-FRCNN (Hu et al., 2021)) contain top-down, bottom-up, and lateral connections, similar to the brain's hierarchical structure for processing sensory information (Park & Friston, 2013). These models mainly simulate the interaction between lower (e.g., A1) and higher (e.g., A2) sensory areas in primates, neglecting the role of higher cortical areas such as frontal cortex and occipital cortex in accomplishing challenging auditory tasks like the cocktail party problem (Bareham et al., 2018; Cohen et al., 2005). But they provide good frameworks for applying top-down attention mechanisms.

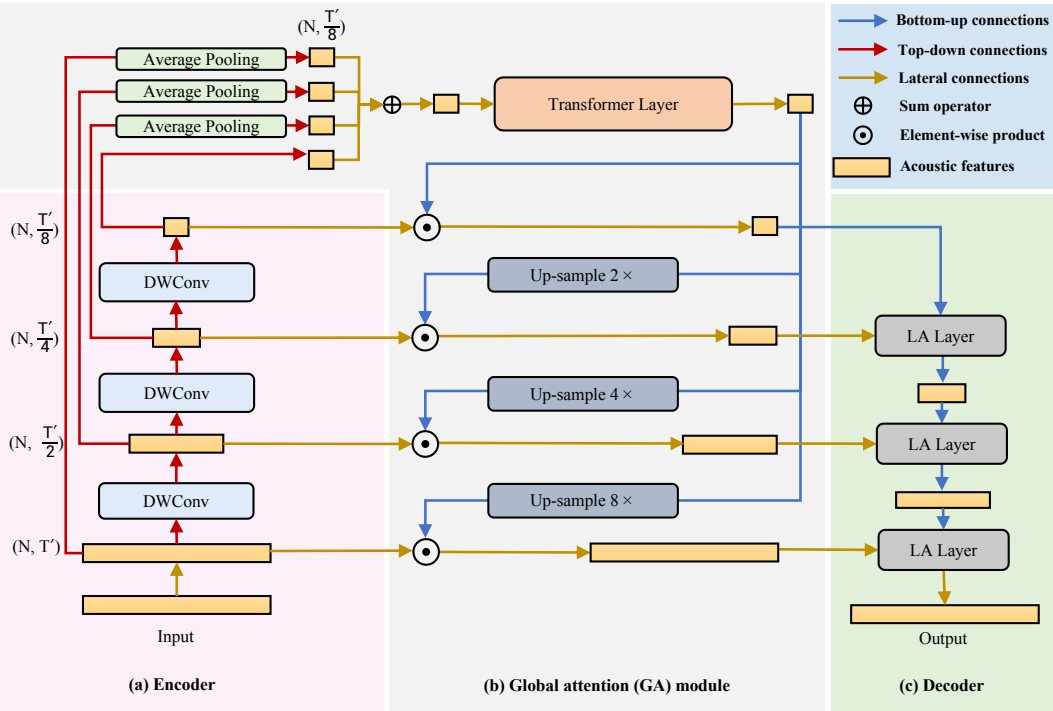

Figure 1: Main architecture of TDANet. N and T' denote the number of channels and length of features, respectively. By down-sampling S times, TDANet contains $S + 1$ features with different temporal resolutions. Here, we set S to 3. The red, blue, and orange arrows indicate bottom-up, top-down, and lateral connections, respectively. (a) The structure of the encoder, where "DWConv" denotes a depthwise convolutional layer with a kernel size of 5 and stride size of 2 followed by GLN. (b) The "Up-sample" layer denotes nearest neighbor interpolation. (c) The structure of decoder, where the LA layer adaptively modulates features of different scales by a set of learnable parameters.

We propose an encoder-decoder architecture equipped with top-down attention for speech separation, which is called TDANet. As shown in Figure 1, TDANet adds a global attention (GA) module to the encoder-decoder architecture, which modulates features of different scales in the encoder top-down through the attention signal obtained from multi-scale features. The modulated features are gradually restored to high-resolution auditory features through local attention (LA) layers in the top-down decoder. The experimental results demonstrated that TDANet achieved competitive separation performance on three datasets (LRS2-2Mix, Libri2Mix (Cosentino et al., 2020), and WHAM! (Wichern et al., 2019)) with far less computational cost. Taking LRS2-2Mix dataset as an example – when comparing with Sepformer, TDANet's MACs are only 5% of it and CPU inference time is only 10% of it.

2 RELATED WORK

In recent years, DNN-based speech separation methods have received widespread attention, and speech separation performance has been substantially improved. Existing DNN-based methods are categorized into two major groups: time-frequency domain (Hershey et al., 2016; Chen et al., 2017; Yu et al., 2017) and time domain (Luo & Mesgarani, 2019; Luo et al., 2020; Tzinis et al., 2020; Chen et al., 2020; Subakan et al., 2021; Hu et al., 2021). Time domain methods obtain better results compared to time-frequency domain methods as they avoid explicit phase estimation. Recently, time domain methods Sepformer (Subakan et al., 2021) and DPTNet (Chen et al., 2020) replace LSTM layers with transformer layers to avoid performance degradation due to long-term dependencies and to process data in parallel for efficiency. These methods achieve better separation performance, but the number of model parameters and computational cost become larger due to additional transformer layers. Although large-scale neural networks can achieve better performance in various tasks such as BERT (Devlin et al., 2019) and DALL·E 2 (Ramesh et al., 2022), it is also an important topic to design lightweight models that can be deployed to low-resource platforms.

The encoder-decoder speech separation model SuDORM-RF (Tzinis et al., 2020) achieves a certain extent of trade-off between model peformance and complexity, but its performance is still far from SOTA methods. Other researchers have designed a fully recurrent convolutional neural network A-FRCNN (Hu et al., 2021) with an asynchronous update scheme and obtained competitive separation results with a relatively small number of parameters, but the computational complexity still has room to improve.

Some earlier works (Xu et al., 2018; Shi et al., 2018; 2019) make use of top-down attention in designing speech separation models, but they only consider the projection of attentional signals to the top layer of the multilayer LSTM and without applying attention to lower layers. This approach may not fully take advantage of top-down attention. In addition, these approaches have a large gap in model performance and complexity with recent models including SuDORM-RF and A-FRCNN.

3 METHOD

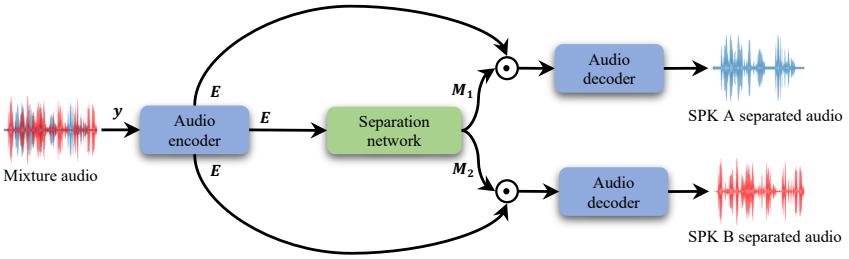

Figure 2: The overall pipeline for the speech separation task. We assume that the mixture audio contains two speakers here. The parameters are shared between the two audio decoders.

3.1 OVERALL PIPELINE

Given an audio containing multiple speakers, the speech separation methods aim to extract different speakers' utterances and route them to different output channels. Let $\boldsymbol{y} \in \mathbb{R}^{1\times T}$ be a multi-speaker time-domain audio signal with the length T:

$$\boldsymbol{y} = \sum_{i}^{C} \boldsymbol{x}_i + \boldsymbol{n}, \tag{1}$$

consisting of C speaker signals $\boldsymbol{x}_i \in \mathbb{R}^{1\times T}$ plus the noise signal $\boldsymbol{n} \in \mathbb{R}^{1\times T}$. We expect the separated speeches $\bar{\boldsymbol{x}}_i \in \mathbb{R}^{1\times T}$ from all speakers to be closer to target speeches $\boldsymbol{x}_i$.

Our proposed method uses the same three-part pipeline as in Conv-TasNet (Luo & Mesgarani, 2019): *an audio encoder*, *a separation network* and *an audio decoder*. The audio encoder transforms

$\boldsymbol{y}$ into a frame-based N-dimensional embedding sequence $\boldsymbol{E} \in \mathbb{R}^{N \times T'}$ with length T', called mixture audio embedding, which can be implemented using a 1-D convolutional layer with kernel size L and stride size $\lfloor L/4 \rfloor$. Similar to existing separation methods (Luo & Mesgarani, 2019; 2018; Luo et al., 2020; Subakan et al., 2021; Chen et al., 2020), instead of directly estimating the target speech embeddings, we use a DNN-based separation network $f(\boldsymbol{E}; \boldsymbol{\theta})$ to generate a set of masks $\boldsymbol{M} = \{\boldsymbol{M}_i \in \mathbb{R}^{N \times T'} | i = 1, ..., C\}$ associated with the target speakers. Each target speech embedding $\bar{\boldsymbol{E}}_i \in \mathbb{R}^{N \times T'}$ is generated by applying the corresponding mask $\boldsymbol{M}_i$ to the mixture audio embedding $\boldsymbol{E}$:

$$\bar{\boldsymbol{E}}_i = \boldsymbol{E} \odot \boldsymbol{M}_i, \tag{2}$$

where $\odot$ denotes element-wise product. Finally, the target waveform $\bar{\boldsymbol{x}}_i$ is reconstructed using the target speech embedding $\bar{\boldsymbol{E}}_i$ through an audio decoder, which can be implemented using a 1-D transposed convolutional layer with the same kernel and stride sizes as the audio encoder.

Overall there are three main components: (a) encoder (see Section 3.2), (b) global attention (GA) module (see Section 3.3) and (c) decoder (see Section 3.4). The process detail is as follows. First, the encoder obtains multi-scale features using the bottom-up connections. Second, the GA module extracts global features at the top layer using multi-scale features, and then the global features are used as attention to modulate features from different scales using top-down connections. Third, the adjacent layer features are used as the input into the decoder to extract the acoustic features through LA layers. This completes one cycle of information processing inside TDANet. Similar to A-FRCNN Hu et al. (2021), we cycle TDANet several times and use the output features from the last cycle as the separation network output (see Section 3.5).

3.2 Encoder of TDANet

The encoder of TDANet is designed for extracting features at different temporal resolutions. Lower layers have higher temporal resolutions, while higher layers have lower temporal resolutions. The encoder works as follows (see Figure 1a). For a mixture audio embedding $\boldsymbol{E}$, the encoder processes $\boldsymbol{E}$ in a bottom-up manner step by step. The bottom-up connections in the encoder are implemented by down-sampling layers, which consist of a 1-D convolutional layer followed by a global normalization layer (GLN) (Luo & Mesgarani, 2019) and PReLU. To efficiently expand the perceptual field, these convolutional layers use dilation convolution with N kernels with size 5, stride size 2 and dilation size 2 instead of standard convolution to aggregate longer contexts. In this way, we obtain features with different resolutions $\{\boldsymbol{F}_i \in \mathbb{R}^{N \times \frac{T'}{2^{i-1}}} | i = 1, ..., S+1\}$, where S denotes the number of down-sampling.

3.3 Global attention (GA) module of TDANet

The GA module works in two steps:

(1) It receives multi-scale features $\{\boldsymbol{F}_i | i = 1, ..., S+1\}$ as input and computes global feature $\mathbf{G}^m \in \mathbb{R}^{N \times \frac{T'}{2^S}}$;

(2) It uses the $\mathbf{G}^m$ as top-down attention to modulate $\{\boldsymbol{F}_i | i = 1, ..., S+1\}$.

Details of the first step in the GA module: Taking multi-scale features $\{\boldsymbol{F}_i | i = 1, ..., S+1\}$ as input, we use the average pooling layers to compress these features in temporal dimension from $\frac{T'}{2^{i-1}}$ to $\frac{T'}{2^S}$ for reducing computational cost. The features with different temporal resolutions are fused into the global feature $\mathbf{G} \in \mathbb{R}^{N \times \frac{T'}{2^S}}$ by a summation operation. The calculation of global feature can be written as:

$$\mathbf{G} = \sum_{i}^{S} p(\boldsymbol{F}_i), \tag{3}$$

where $p(\cdot)$ denotes the average pooling layer. $\mathbf{G}$ will be treated as input to the transformer layer to obtain the speaker's acoustic patterns from global contexts.

The transformer layer contains two components: multi-head self-attention (MHSA) and feed-forward network (FFN) layers, as shown in Figure 3. The MHSA layer is the same as the encoder

structure defined in Transformer (Vaswani et al., 2017). This layer has been widely used in recent speech processing tasks (Chen et al., 2020; Subakan et al., 2021; Lam et al., 2021). First, the position information $\boldsymbol{d}$ of different frames is added to global feature $\mathbf{G}$ to get $\bar{\mathbf{G}} = \mathbf{G} + \mathbf{d}$, $\bar{\mathbf{G}} \in \mathbb{R}^{N \times \frac{T'}{2^S}}$. Then, each attention head calculates the dot product attention between all elements of $\bar{\mathbf{G}}$ in different feature spaces to direct model focus to different aspects of information. Finally, the output $\hat{\mathbf{G}} \in \mathbb{R}^{N \times \frac{T'}{2^S}}$ of MHSA is fed into GLN and then connected $\mathbf{G}$ through the residual connection to obtain $\check{\mathbf{G}} \in \mathbb{R}^{N \times \frac{T'}{2^S}}$.

The FFN layer followed by MHSA layer consists of three convolutional layers. First, $\check{\mathbf{G}}$ is processed in a 1×1 convolutional layer with GLN mapped to the $2N$-dimensional representation space. Then, the second convolutional layer composed of a 5×5 depthwise convolutional layer followed by GLN extracts relationship among $2N$ sequences. Finally, a 1×1 convolutional layer followed by GLN compresses the $2N$-dimensional features into N-dimensions to obtain $\mathbf{G}^m \in \mathbb{R}^{N \times \frac{T'}{2^S}}$ and also use a residual connection to alleviate the vanishing gradient problem.

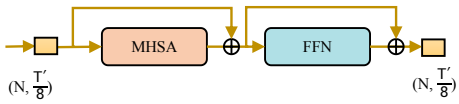

Figure 3: The structure of transformer layer.

Details of the second step in the GA module: We use $\mathbf{G}^m$ as top-down attention (see Figure 1b) to modulate $\boldsymbol{F}_i$ before adding it to the decoder. Specifically, we up-sample $\mathbf{G}^m$ along the time dimension using nearest neighbor interpolation to obtain the same time dimension as $\boldsymbol{F}_i$, and then use the Sigmoid function to obtain the attention signal. In this way, the multi-scale semantic information of $\mathbf{G}^m$ can guide the local features $\boldsymbol{F}_i$ to focus more on feature details that may be lost in the bottom-up path in order to improve quality of separated audios. The modulated features are calculated as:

$$\boldsymbol{F}_i' = \sigma(\phi(\mathbf{G}^m)) \odot \boldsymbol{F}, \tag{4}$$

where ϕ denotes nearest neighbor interpolation, σ stands for the Sigmoid function and $\odot$ denotes element-wise product. The modulated features $\boldsymbol{F}_i' \in \mathbb{R}^{N \times \frac{T'}{2^{i-1}}}$ are used as input to the decoder to extract different speakers' features.

The GA module functions as higher cortical areas such as frontal cortex and occipital cortex in the brain which play an important role in accomplishing complicated sensory tasks. Here, it is only designed for top-down modulation in performing speech separation tasks (Bareham et al., 2018; Cohen et al., 2005).

3.4 Decoder of TDANet

The decoder progressively extracts acoustic features through top-down connections whose core component is the LA layer, as shown in Figure 1c. The LA layer uses a small amount of parameters ($\sim$ 0.01M) to generate adaptive parameters ρ and τ from higher-level features as a guide for extracting finer-grained speech features. Figure 4 shows detailed architecture of the LA layer. First, LA layer uses N 1-D kernels with length 5 to perform convolution in temporal dimension using features $\boldsymbol{F}_{i+1}'$, resulting in $\tau \in \mathbb{R}^{N \times \frac{T'}{2^{i-1}}}$. Meanwhile, another identical 1-D convolutional layer followed by GLN and Sigmoid uses $\boldsymbol{F}_{i+1}'$ as input, resulting in attention signal $\rho \in \mathbb{R}^{N \times \frac{T'}{2^{i-1}}}$. Consequently,

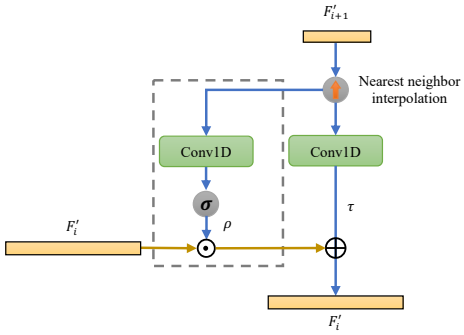

Figure 4: The structure of LA layer in the decoder, where "Conv1D" denotes the 5×5 depthwise convolutional layer followed by GLN.

$$\tau = h_2(\phi(\boldsymbol{F}_{i+1}')), \tag{5}$$

$$\rho = \sigma(h_1(\phi(\boldsymbol{F}_{i+1}'))), \tag{6}$$

where h_1 and h_2 denote two different parameters of 1-D convolutional layers followed by GLN. $\boldsymbol{F}_i'$, ρ and τ have the same dimensions. The learnable parameters are used to adaptively modulate $\boldsymbol{F}_i'$ via a top-down local attention ρ. This process is formulated by

$$\boldsymbol{F}_i' = \rho \odot \boldsymbol{F}_i' + \tau. \tag{7}$$

After the feature extraction from the highest layer to the lowest layer, the output of the decoder use C convolutional layers with N kernels with length of 1 and stride of 1 followed by an activation function (ReLU) to obtain C masks $\boldsymbol{M}_i$, respectively.

3.5 Unfolding method of TDANet

Clearly, the bottom-up, lateral and top-down connections (Figure 1) make TDANet a recurrent model. We need to unfold it through time for training and testing. We adopt the unfolding scheme summation connection (SC) as in A-FRCNN (Hu et al., 2021). Figure 1 shows a single block in the whole unfolding scheme. We repeat the block B times (weight sharing), such that the model's input adds up with each block's output as the next block's input.

4 Experiment Configurations

4.1 Dataset Simulations

We evaluated TDANet and other existing methods on three datasets: LibriMix, WHAM!, LRS2-2Mix). The speaker identities of `training/validation` set and `test` set are non-intersecting, meaning unseen speaker data from the training phase were used during testing.

Libri2Mix (Cosentino et al., 2020). In this dataset, the target speech in each mixture audio was randomly selected from a subset of LibriSpeech's train-100 (Panayotov et al., 2015) and mixed with uniformly sampled Loudness Units relative to Full Scale (LUFS) (Series, 2011) between -25 and -33 dB. To simulate real-world scenarios, random noise samples were added to the mixture audio with loudness uniformly sampled between -38 and -30 LUFS. Each mixture audio contains two different speakers and have a duration of 3 seconds.

WAHM! (Wichern et al., 2019). This dataset acts as a noisy version of WSJ0-2Mix (Hershey et al., 2016). In the WSJ0-2Mix dataset, the overlap between two speaker audios in the mixture is 100%, which is too idealistic, and thus model trained on this dataset does not achieve generalization to speeches from a broader range of speakers (Cosentino et al., 2020). In the WHAM! dataset, speeches were mixed with noise recorded in scenes such as cafes, restaurants and bars. The signal-to-noise ratio of the noise was uniformly sampled between -6db and 3db, making it more challenging mixture audios than WSJ0-2Mix.

LRS2-2Mix. The LRS2 dataset (Afouras et al., 2018) contains thousands of video clips acquired through BBC. LRS2 contains a large amount of noise and reverberation interference, which is more challenging and closer to the actual environment than the WSJ0 (Garofolo et al., 1993) and LibriSpeech (Panayotov et al., 2015) corpora. We created a new speech separation dataset, LRS2-2Mix, using the LRS2 corpus, where the training set, validation set and test set contain 20000, 5000 and 3000 utterances, respectively. The two different speaker audios with 16 kHz sample rate were randomly selected from the LRS2 corpus and were mixed with signal-to-noise ratios sampled between -5 dB and 5 dB. The data were simulated using a script consistent with WSJ0-2Mix[1]. The length of mixture audios is 2 seconds.

4.2 Model and training Configurations

We set the kernel size L of the *audio encoder* and *audio decoder* in the overall pipeline to 4 ms and stride size $\lfloor L/4 \rfloor$ to 1 ms. The number of down-sampling S was set to 4. The number of channels N of all convolutional layers in each layer was set to 512, and the SC method proposed by A-FRCNN (Hu et al., 2021) was used to unfold $B = 16$ times at the macro level. For the MHSA layer, we set the number of attention heads to 8, the dimension to 512, and used sine and cosine functions

[1] http://www.merl.com/demos/deep-clustering/create-speaker-mixtures.zip

with different frequencies for the position encoding. In addition, we set the number of channels for the three convolutional layers in the FFN layer to (512, 1024, 512), the kernel sizes to (1, 5, 1), the stride sizes to (1, 1, 1), and the bias settings to (False, True, False). To avoid overfitting, we set the probability of all dropouts to 0.1.

We trained all models for 500 epochs. The batch size was set to 1 at the utterance level. Our proposed model used the Adam (Kingma & Ba, 2015) optimizer with an initial learning rate of 0.001. We used maximization of scale-invariant signal-to-noise ratio (SI-SNR) (Hershey et al., 2016) as the training goal (See Appendix A for details). Once the best model was not found for 15 successive epochs, we adjusted the learning rate to half of the previous one. Moreover, we stopped training early when the best model was not found for 30 successive epochs. During the training process, we used gradient clipping with a maximum L2 norm of 5 to avoid gradient explosion. For all experiments, we used $8\times$ GeForce RTX 3080 for training and testing. The PyTorch implementation of our method can be found in **Supplementary Material**. This project is under the MIT license.

4.3 Evaluation Metrics

In all experiments, we report the results of scale-invariant signal-to-noise ratio improvement (SI-SNRi) (Le Roux et al., 2019) and signal-to-distortion ratio improvement (SDRi) (Vincent et al., 2006) used to measure clarity of separated audios. For measuring model efficiency, we report the processing time consumption per second (real-time factor, RTF) for all models, indicated by "RTF" in the tables throughout this paper. It was calculated by processing ten audio tracks of 1 second in length and 16 kHz in sample rate on CPU and GPU (total processing time / 10), represented as "CPU RTF" and "GPU RTF", respectively. The numbers were then averaged after running 1000 times. Also, we used the parameter size and the number of MACs to measure the model size, where the MACs were calculated using the open-source tool PyTorch-OpCounter[2]. This toolkit is used under the MIT license.

5 Results

5.1 Ablation study

To better understand the effectiveness of top-down attention, we investigated the impacts of two types of attention components in the model for separation performance, including GA module and LA layer. All experimental results were obtained by training and testing on the LRS2-2Mix dataset.

Table 1: Comparison over separation performance and efficiency among different types of top-down attention. "$\surd$" indicates the case where a component is used in the TDANet. "$\times$" indicates the opposite.

GA module	LA layer	SI-SNRi (dB)	SDRi (dB)	Params (M)	MACs (G/s)	CPU RTF (s)
$\times$	$\times$	10.1	10.5	**0.7**	**2.5**	**0.65**
$\surd$	$\times$	12.4	12.7	2.3	4.6	0.79
$\times$	$\surd$	11.9	12.2	**0.7**	2.6	0.65
$\surd$	$\surd$	**13.2**	**13.5**	2.3	4.7	0.79

We first constructed a control model that does not contain the GA module and LA layer. It takes the top-layer features of the encoder as input to the decoder. Note that excluding the LA layer means replacing the gray box in Figure 4 with summation operation. This architecture then becomes a typical U-net (Ronneberger et al., 2015), similar to a basic block used as in SuDORM-RF (Tzinis et al., 2020). From Table 1, it is seen that SI-SNRi of this model is only 10.1 dB.

When only adding the GA module into TDANet, the SI-SNRi increased by 2.3 dB compared to when it was absent (Table 1). The GA module contains a transformer layer and top-down projections. We are interested in whether the performance improvement came from the transformer layer or the top-down projections. Table 4 from Appendix B.1 demonstrates that the performance improvement is

[2]https://github.com/Lyken17/pytorch-OpCounter

independent of the transformer layer. When only adding the LA layer, we observed that the SI-SNRi increased by 1.8 dB without increasing computational cost. We noticed that the GA module had larger impact on performance improvement. One possible explanation is that the GA module modulates on a wide range of features, while the LA layer only focuses on adjacent ones. When both GA module and LA layer were used, the performance were the best.

More ablation studies are reported in Appendix B. We observed that multi-scale features using summation fusion strategy as GA module input were efficient for performance improvement without significantly increasing computational cost (Appendix B.2). In addition, TDANet obtained the best separation performance when both MHSA and FFN layers were present in the GA module (Appendix B.3).

5.2 Comparisons with state-of-the-art methods

We conducted extensive experiments to quantitatively compare speech separation performance of our proposed TDANet with some existing speech separation models on three datasets. The results are shown in Table 2. We selected some typical models that have shown good separation performance, including BLSTM-TasNet (Luo & Mesgarani, 2018), Conv-TasNet (Luo & Mesgarani, 2019), SuDORM-RF (Tzinis et al., 2020), DualPathRNN (Luo et al., 2020), DPTNet (Chen et al., 2020), Sepformer (Subakan et al., 2021), and A-FRCNN (Hu et al., 2021). SuDORM-RF contains two variants with good performance, suffixed with 1.0x and 2.5x, indicating that these variants consist of 16 and 40 blocks, respectively. A-FRCNN has a suffix "16" denoting that the model is expanded 16 times at the macro level. LRS2-2Mix is the newly proposed speech separation dataset in this paper, so none of these models reported results on this dataset. We trained and tested these models using the Asteroid Toolkit (Pariente et al., 2020) to obtain the missing results. This toolkit is under MIT license.

Separation performance. TDANet obtained competitive separation performance with much lower complexity than previous SOTA models on the three datasets (see Table 2), clearly demonstrating the importance of top-down attention for the encoder-decoder network. On the LRS2-2Mix dataset, TDANet lost only 0.3 dB SI-SNRi in separation performance but with only 8% of number of parameters compared to Sepformer, one of previous SOTA models. On the other two datasets, TDANet also obtained competitive results compared to previous SOTA models.

Some audio examples from LRS2-2Mix separated by different models are provided in the **Supplementary Material**. In most examples, we found that separation results with TDANet sound better than those from other models.

Table 2: Quantitative comparison between TDANet and other existing models on three datasets (test set). "-" indicates that their results have not been reported in other papers. "*" denotes that they were not reported in the original paper but were trained and tested using the Asteroid Toolkit (Pariente et al., 2020). The results of DPTNet on Libri2Mix and WHAM! datasets were reported in (Yao et al., 2022), but not in the original paper.

Model [Authors, Years]	LRS2-2Mix		Libri2Mix		WHAM!		Params
	SI-SNRi	SDRi	SI-SNRi	SDRi	SI-SNRi	SDRi	(M)
BLSTM-TasNet [Luo et al, 2018]	6.1*	6.8*	7.9	8.7	9.8	-	23.6
Conv-TasNet [Luo et al, 2019]	10.6*	11.0*	12.2	12.7	12.7	-	5.6
SuDoRM-RF1.0x [Tzinis et al, 2020]	11.0*	11,4*	13.5	14.0	12.9	13.3	2.7
SuDoRM-RF2.5x [Tzinis et al, 2020]	11.3*	11.7*	14.0	14.4	13.7	14.1	6.4
DualPathRNN [Luo et al, 2020]	12.7*	13.0*	16.1	16.6	13.7	14.1	2.7
DPTNet [Chen et al, 2020]	13.3*	13.6*	16.7	17.1	14.9	15.3	2.7
A-FRCNN-16 [Hu et al, 2021]	13.0*	13.3*	16.7	17.2	14.5	14.8	6.1
Sepformer [Subakan et al, 2021]	13.5*	13.8*	16.5	17.0	14.4	15.0	26.0
TDANet (*ours*)	13.2	13.5	16.9	17.4	14.8	15.0	2.3
TDANet Large (*ours*)	**14.2**	**14.5**	**17.4**	**17.9**	**15.2**	**15.4**	**2.3**

Separation effciency. In addition to number of model parameters, MACs as well as training and inference time are also important indicators of model complexity, as models with a small amount of parameters may also have a large number of MACs and could be slow to train and infer. By using

RTF and MACs as metrics, we evaluated complexity of pervious models (Table 3). We observed that TDANet outperformed the previous SOTA models in model complexity. For example, compared to DPTNet and Sepformer, TDANet's MACs are 5% of them, and backward time is 13% and 47% of them, respectively, significantly reducing the time consumption during training. In addition, compared with A-FRCNN and SuDORM-RF 2.5x, TDANet's GPU inference time is 19% and 18% of them, GPU training time is 47% and 34% of them, and CPU inference time is 15% and 46% of them, respectively. These results suggest that TDANet can be more easily deployed to low-resource devices.

Large-size version. For the proposed method, we also constructed a large-size version of TDANet, namely TDANet Large, by modifying the kernel length of the *audio encoder* and *audio decoder* from 4ms to 2ms and the stride size from 2ms to 0.5ms. Note that this does not change the model size. TDANet Large obtained SOTA separation performance without a significant increase in computational cost on all three datasets (see Table 2 and Table 3). TDANet Large can still maintain a small computational complexity compared to other existing lightweight models. For example, the inference time on the GPU is about 3 times faster, and MACs are about 14 times smaller compared to A-FRCNN.

Table 3: Comparison of inference time and MACs on the LRS2-2Mix test set. The test environment for feedforward RTF (F GPU RTF) and backward RTF (B GPU RTF) is Nvidia GeForce RTX 2080 Ti, while the test environment for CPU RTF is Intel(R) Xeon(R) Silver 4210 CPU @ 2.20GHz, single-threaded.

Model	F GPU RTF (ms)	B GPU RTF (ms)	CPU RTF (s)	MACs (G/s)
BLSTM-TasNet	233.85	654.14	5.9	43.0
Conv-TasNet	15.28	**56.91**	0.82	10.2
SuDoRM-RF 1.0x	27.86	95.37	**0.75**	4.6
SuDoRM-RF 2.5x	64.70	228.57	1.73	10.1
DualPathRNN	88.79	241.54	8.13	85.3
DPTNet	103.26	689.06	10.49	102.5
A-FRCNN-16	61.16	183.65	5.32	125.3
Sepformer	65.61	184.91	7.55	86.9
TDANet (*ours*)	**11.76**	86.56	0.79	4.7
TDANet Large (*ours*)	23.77	97.92	1.78	9.1

6 Conclusions

To fill the gap between the performance of SOTA and lightweight models, we designed an encoder-decoder speech separation network with top-down global attention and local attention, inspired by our brain's top-down attention for solving the cocktail party problem. Our extensive experimental results showed that TDANet could significantly reduce model complexity and inference speed while ensuring separation performance.

Limitations: The TDANet only reflects certain working principles of the auditory system and is not a faithful model of the auditory system. For example, it requires a bottom-up encoder and a top-down decoder, and it is unclear how such components are implemented in the brain.

References

Triantafyllos Afouras, Joon Son Chung, Andrew Senior, Oriol Vinyals, and Andrew Zisserman. Deep audio-visual speech recognition. *IEEE transactions on pattern analysis and machine intelligence*, 2018.

David Attwell and Simon B Laughlin. An energy budget for signaling in the grey matter of the brain. *Journal of Cerebral Blood Flow & Metabolism*, 21(10):1133–1145, 2001.

Corinne A Bareham, Stanimira D Georgieva, Marc R Kamke, David Lloyd, Tristan A Bekinschtein, and Jason B Mattingley. Role of the right inferior parietal cortex in auditory selective attention: An rtms study. *Cortex*, 99:30–38, 2018.

Adelbert W Bronkhorst. The cocktail-party problem revisited: early processing and selection of multi-talker speech. *Attention, Perception, & Psychophysics*, 77(5):1465–1487, 2015.

Jingjing Chen, Qirong Mao, and Dong Liu. Dual-Path Transformer Network: Direct Context-Aware Modeling for End-to-End Monaural Speech Separation. In *Annual Conference of the International Speech Communication Association (Interspeech)*, pp. 2642–2646, 2020.

Zhuo Chen, Yi Luo, and Nima Mesgarani. Deep attractor network for single-microphone speaker separation. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 246–250. IEEE, 2017.

E Colin Cherry. Some experiments on the recognition of speech, with one and with two ears. *The Journal of the acoustical society of America*, 25(5):975–979, 1953.

Yale E Cohen, Brian E Russ, and Gordon W Gifford III. Auditory processing in the posterior parietal cortex. *Behavioral and Cognitive Neuroscience Reviews*, 4(3):218–231, 2005.

Joris Cosentino, Manuel Pariente, Samuele Cornell, Antoine Deleforge, and Emmanuel Vincent. Librimix: an open-source dataset for generalizable speech separation. *arXiv preprint arXiv:2005.11262*, 2020.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: pre-training of deep bidirectional transformers for language understanding. In *Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pp. 4171–4186, 2019.

Luis Morís Fernández, Maya Visser, Noelia Ventura-Campos, César Ávila, and Salvador Soto-Faraco. Top-down attention regulates the neural expression of audiovisual integration. *NeuroImage*, 119:272–285, 2015.

John Garofolo, David Graff, Doug Paul, and David Pallett. Csr-i (wsj0) complete ldc93s6a. *Web Download. Philadelphia: Linguistic Data Consortium*, 83, 1993.

Adam Gazzaley, Jeffrey W Cooney, Kevin McEvoy, Robert T Knight, and Mark D'esposito. Top-down enhancement and suppression of the magnitude and speed of neural activity. *Journal of cognitive neuroscience*, 17(3):507–517, 2005.

Simon Haykin and Zhe Chen. The cocktail party problem. *Neural computation*, 17(9):1875–1902, 2005.

John R Hershey, Zhuo Chen, Jonathan Le Roux, and Shinji Watanabe. Deep clustering: discriminative embeddings for segmentation and separation. In *IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pp. 31–35. IEEE, 2016.

Clare Howarth, Padraig Gleeson, and David Attwell. Updated energy budgets for neural computation in the neocortex and cerebellum. *Journal of Cerebral Blood Flow & Metabolism*, 32(7): 1222–1232, 2012.

Xiaolin Hu, Kai Li, Weiyi Zhang, Yi Luo, Jean-Marie Lemercier, and Timo Gerkmann. Speech separation using an asynchronous fully recurrent convolutional neural network. volume 34, pp. 22509–22522, 2021.

Jennifer A Johnson and Robert J Zatorre. Attention to simultaneous unrelated auditory and visual events: behavioral and neural correlates. *Cerebral cortex*, 15(10):1609–1620, 2005.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*, 2015.

Max WY Lam, Jun Wang, Dan Su, and Dong Yu. Sandglasset: a light multi-granularity self-attentive network for time-domain speech separation. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 5759–5763. IEEE, 2021.

Jonathan Le Roux, Scott Wisdom, Hakan Erdogan, and John R Hershey. Sdr–half-baked or well done? In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 626–630. IEEE, 2019.

Kai Li and Yi Luo. On the design and training strategies for rnn-based online neural speech separation systems. *arXiv preprint arXiv:2206.07340*, 2022.

Yi Luo and Nima Mesgarani. Tasnet: time-domain audio separation network for real-time, single-channel speech separation. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 696–700. IEEE, 2018.

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*, 27(8): 1256–1266, 2019.

Yi Luo, Zhuo Chen, and Takuya Yoshioka. Dual-path rnn: efficient long sequence modeling for time-domain single-channel speech separation. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 46–50. IEEE, 2020.

Nima Mesgarani and Edward F Chang. Selective cortical representation of attended speaker in multi-talker speech perception. *Nature*, 485(7397):233–236, 2012.

Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur. Librispeech: an asr corpus based on public domain audio books. In *IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pp. 5206–5210. IEEE, 2015.

Manuel Pariente, Samuele Cornell, Joris Cosentino, Sunit Sivasankaran, Efthymios Tzinis, Jens Heitkaemper, Michel Olvera, Fabian-Robert Stöter, Mathieu Hu, Juan M. Martín-Doñas, David Ditter, Ariel Frank, Antoine Deleforge, and Emmanuel Vincent. Asteroid: the PyTorch-based audio source separation toolkit for researchers. In *Annual Conference of the International Speech Communication Association (Interspeech)*, 2020.

Hae-Jeong Park and Karl Friston. Structural and functional brain networks: from connections to cognition. *Science*, 342(6158):1238411, 2013.

Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*, 2022.

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention (MICCAI)*, pp. 234–241. Springer, 2015.

BS Series. Algorithms to measure audio programme loudness and true-peak audio level. 2011.

Jing Shi, Jiaming Xu, Guangcan Liu, Bo Xu, et al. Listen, think and listen again: Capturing top-down auditory attention for speaker-independent speech separation. In *IJCAI*, pp. 4353–4360, 2018.

Jing Shi, Jiaming Xu, and Bo Xu. Which ones are speaking? speaker-inferred model for multi-talker speech separation. In *Annual Conference of the International Speech Communication Association (Interspeech)*, pp. 4609–4613, 2019.

Cem Subakan, Mirco Ravanelli, Samuele Cornell, Mirko Bronzi, and Jianyuan Zhong. Attention is all you need in speech separation. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 21–25. IEEE, 2021.

Efthymios Tzinis, Zhepei Wang, and Paris Smaragdis. Sudo rm-rf: efficient networks for universal audio source separation. In *IEEE 30th International Workshop on Machine Learning for Signal Processing (MLSP)*, pp. 1–6. IEEE, 2020.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. volume 30, 2017.

Emmanuel Vincent, Rémi Gribonval, and Cédric Févotte. Performance measurement in blind audio source separation. *IEEE transactions on audio, speech, and language processing*, 14(4):1462–1469, 2006.

Gordon Wichern, Joe Antognini, Michael Flynn, Licheng Richard Zhu, Emmett McQuinn, Dwight Crow, Ethan Manilow, and Jonathan Le Roux. WHAM!: Extending Speech Separation to Noisy Environments. In *Annual Conference of the International Speech Communication Association (Interspeech)*, pp. 1368–1372, 2019.

Noelle L Wood and Nelson Cowan. The cocktail party phenomenon revisited: attention and memory in the classic selective listening procedure of cherry (1953). *Journal of Experimental Psychology: General*, 124(3):243, 1995.

Jiaming Xu, Jing Shi, Guangcan Liu, Xiuyi Chen, and Bo Xu. Modeling attention and memory for auditory selection in a cocktail party environment. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

Zengwei Yao, Wenjie Pei, Fanglin Chen, Guangming Lu, and David Zhang. Stepwise-refining speech separation network via fine-grained encoding in high-order latent domain. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 30:378–393, 2022.

Dong Yu, Morten Kolbæk, Zheng-Hua Tan, and Jesper Jensen. Permutation invariant training of deep models for speaker-independent multi-talker speech separation. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 241–245. IEEE, 2017.

A Training Method

We apply the permutation-invariant training (PIT) method (Yu et al., 2017) to solve the permutation problem (Hershey et al., 2016) in order to maximize the scale-invariant signal-to-noise ratio (SI-SNR) (Le Roux et al., 2019). The SI-SNR for each speaker is defined as:

$$\text{SI-SNR}_i = 20 \times log_{10} \frac{||\boldsymbol{A}_{target}||}{||\boldsymbol{e}_{noise}||}, \tag{8}$$

where

$$\boldsymbol{A}_{target} = \frac{\langle \bar{\boldsymbol{x}}_i, \boldsymbol{x}_i \rangle}{||\boldsymbol{x}_i||_2^2}, \boldsymbol{e}_{noise} = \bar{\boldsymbol{x}}_i - \boldsymbol{A}_{target}. \tag{9}$$

In the above equation, $\langle \cdot, \cdot \rangle$ denotes the inner product and $|| \cdot ||_2^2$ denotes the $L2$-norm.

B Additional experiments

B.1 Effect of transformer layer in the GA module

The GA module contains a transformer layer and top-down projections. We are interested in whether the performance gain is from the transformer layer or the top-down projections. For TDANet with only the GA module, we removed top-down projections from the GA module, keeping only the transformer layer. When the transformer layer is present, we take the output of it as the decoder input. When the transformer layer is absent, we use the top-layer features of the encoder as input of the decoder. Table 4 shows the experimental results. We found that the presence of the transformer layer does not significantly affect separation performance; both configurations yielded poor results compared with the configuration with top-down projections (SI-SNRi 12.4 dB, Table 1). These results further demonstrate the importance of top-down global attention.

B.2 Effect of multi-scale input to GA module

To examine the influence on separation performance between two different features as input to the GA module: multi-scale fused features $\mathbf{G}$ and top-layer features $\boldsymbol{F}_{S+1}$, we experimented with each of them separately. The experimental results are shown in Table 5. We chose $\mathbf{G}$ instead of $\boldsymbol{F}_{S+1}$

Table 4: Effect of transformer layer. "TL" denotes transformer layer in the GA module.

TL	SI-SNRi (dB)	SDRi (dB)	Params (M)	MACs (G/s)	CPU RTF (s)
$\surd$	**10.3**	**10.7**	2.3	4.6	0.77
$\times$	10.1	10.5	**0.7**	**2.5**	**0.65**

Table 5: Comparison over separation performance and model complexity between different multi-scale fused top-layer features used for input into the GA module.

Methods	SI-SNRi (dB)	SDRi (dB)	Params (M)	MACs (G/s)	CPU RTF (s)
$\mathbf{G}$	**13.2**	**13.5**	**2.3**	**4.7**	0.79
$\boldsymbol{F}_{S+1}$	12.8	13.1	2.3	4.7	**0.74**

because the former achieved better separation performance (0.4 dB ↑) without a remarkable increase in computational cost.

In addition, we investigated two different strategies for multi-scale feature fusion: summation and concatenation. Table 6 shows the results. Summation outperformed concatenation in terms of both separation performance and computational efficiency. Therefore, we employed summation operation as the fusion strategy.

Table 6: Comparison over separation performance and model complexity between different fusion strategies.

Fusion strategy	SI-SNRi (dB)	SDRi (dB)	Params (M)	MACs (G/s)	CPU RTF (s)
Summation	**13.2**	**13.5**	**2.3**	**4.7**	**0.79**
Concatenation	13.0	13.3	3.6	6.1	0.83

B.3 Effect of transformer layer components

The transformer layer contains two components: MHSA and FFN layers. We have investigated the effectiveness of MHSA and FFN layers on separation performance and inference efficiency in Table 7. We used the TDANet with both GA module and LA layers as the control model. When we remove MHSA and FFN layers from GA, the top-layer features of the encoder are used as top-down global attention. We observed that the these layers played a critical role in improving separation performance. When adding either MHSA or FFN layer, we observed that the impact of FFN (↑ 1.3 dB) on performance was more significant than that of MHSA (↑ 0.7 dB). One possible reason may be that the perceptual field of the top-layer features is already large enough so that the transformer structure, known to be good at modeling long sequences, does not improve separation performance as much.

Table 7: Comparison on the importance of MHSA and FFN layers.

MHSA	FFN	SI-SNRi (dB)	SDRi (dB)	Params (M)	MACs (G/s)	CPU RTF (s)
$\times$	$\times$	11.2	11.5	**0.2**	**2.6**	**0.68**
$\surd$	$\times$	11.9	12.2	1.3	3.6	0.73
$\times$	$\surd$	12.5	12.8	1.3	3.7	0.73
$\surd$	$\surd$	**13.2**	**13.5**	2.3	4.7	0.79