PAPEZ: RESOURCE-EFFICIENT SPEECH SEPARATION WITH AUDITORY WORKING MEMORY

Hyunseok Oh, Juheon Yi, Youngki Lee

Seoul National University

ABSTRACT

Transformer-based models recently reached state-of-the-art single-channel speech separation accuracy; However, their extreme computational load makes it difficult to deploy them in resource-constrained mobile or IoT devices. We thus present Papez, a lightweight and computation-efficient single-channel speech separation model. Papez is based on three key techniques. We first replace the inter-chunk Transformer with small-sized auditory working memory. Second, we adaptively prune the input tokens that do not need further processing. Finally, we reduce the number of parameters through the recurrent transformer. Our extensive evaluation shows that Papez achieves the best resource and accuracy tradeoffs with a large margin. We publicly share our source code at `https://github.com/snuhcs/Papez`.

***Index Terms*—** speech separation, auditory working memory, adaptive computation, transformer, deep learning.

1. INTRODUCTION

Speech separation serves as a preparatory stage for various downstream applications, e.g., speech recognition, speaker diarization, and machine translation. While Transformer-based models have recently achieved state-of-the-art separation performance [1, 2, 3], they are severely intensive in computation and memory, making it difficult to deploy them in resource-constrained mobile/IoT devices. For example, SepFormer [1] with 26M parameters costs 2.53 sec for inference on an 8kHz, 0.5-sec input with Samsung Galaxy S20 CPU; real-time mobile speech processing is thus infeasible.

In this paper, we propose Papez, a resource-efficient single-channel speech separation model. We design Papez with a couple of key observations regarding the inefficiencies in the state-of-the-art transformer-based models. First, we find that a widely-used dual-path process approach [1, 4] incurs unnecessary computational overhead [3]. Dual-path process alleviates the excessive processing load of elongated input sequences by chunking the sequence and modeling the intra-chunk and inter-chunk dependency separately [4, 5]. However, our in-depth analysis finds that only a few transformer layers of the inter-chunk transformer are utilized in effect. Second, prior models employ a fixed processing path regardless of the input content. Yet, a meaningful portion of an input signal can be pruned out during the computation since some segments of a speech mixture are much easier to separate than others, e.g., silence or single-speaker speech [6].

Leveraging our observations, we develop two key techniques to design Papez: *Auditory Working Memory (AWM) Transformer* and *Adaptive Token Pruning*. First, our *AWM Transformer* architecture augments the intra-chunk transformer with a small-sized short-term memory to replace the inter-transformer. In detail, the short-term memory captures and stores the global context needed for local intra-chunk processing. Note that our architecture is bio-inspired; the human brain has a functionality called AWM that temporarily stores audio features for auditory processing [7, 8]. Also, studies in NLP found that global tokens attending the whole sequence are effective in modeling a very long sequence [9, 10].

Second, our *Adaptive Token Pruning* technique context-adaptively prunes redundant input tokens that need no further processing. In detail, each token self-determines whether to stop or continue processing itself in every transformer layer probabilistically. This dynamically scales down the width (the number of tokens) and depth (layer iteration) of the transformer computation. Moreover, we optimize the latency and model size by (i) piggybacking the probability estimator to the feed-forward network (FFN) of the transformer layer (rather than using a dedicated estimator as in prior works [11, 12]), and (ii) employing the recurrent transformer architecture that shares weight across transformer layers.

Our extensive evaluation shows that Papez enables a significantly more efficient computation-accuracy tradeoff compared to prior models. Specifically, Papez achieves $3.6\times$, $4.14\times$ smaller parameters and $2.44\times$, $1.67\times$ faster inference latency than Tiny-Sepformer [13] and A-FRCNN [14] with 4.2dB and 1.1dB higher SI-SNR in WSJ0-2Mix dataset, respectively. We also reduce the model size of the state-of-the-art Sepformer by $17.7\times$ with minimal accuracy drop. Our techniques are widely applicable on other transformer-based models (e.g., Sepformer variants [2, 3, 13]) as well.

2. RELATED WORK

Separating each speech source from a single-channel mixture of multiple speakers is a fundamental yet difficult problem [5]. Initially, frequency-domain approaches separate the mixture in its time-frequency STFT representation [15].

The success of TasNet [16] has brought great interest in RNN-based time-domain approaches, which uses the non-STFT encoder to extract an effectively separable representation of the waveform. The problem is that encoded sequences get unbearably long to model with RNNs since a smaller sliding window of encoder led to higher performance [4, 17, 5]. Dual-path process addresses this by chunking the sequence

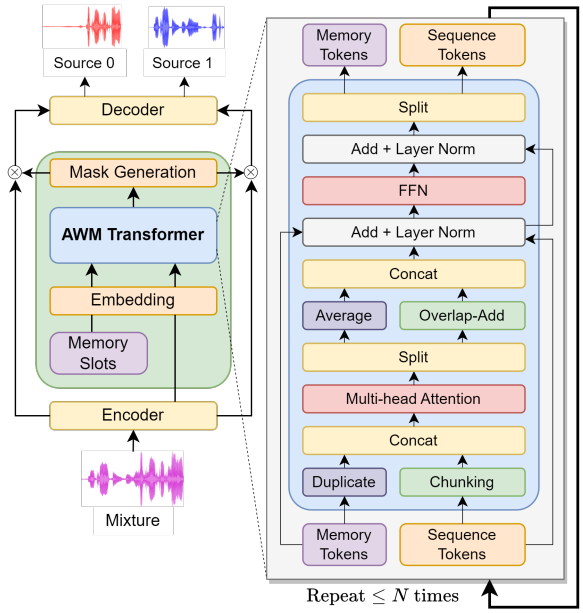

Fig. 1: Overall architecture of our Papez model. The iteration steps of the AWM Transformer layer are determined by our Adaptive Token Pruning technique.

and modeling intra-chunk and inter-chunk separately [4].

Unfortunately, RNN is limited due to its sequential nature; its computation cannot parallelize, and it relies on the inductive bias of temporal invariance [1, 18]. Recently, RNN-free transformer-based models (i.e., Sepformer [1] and its variants [2, 3]) achieved remarkable performance with three factors: (i) maximal connectivity of self-attention mechanism to model long-term dependency, (ii) no inductive bias unlike CNNs (translation invariance) or RNNs (temporal invariance) [18], and (iii) parallelization of computation [1]. Although these models are faster than RNN-Transformer hybrid models [5, 17], the dual-path process doubles the time complexity of the Transformer, which is already huge.

A few works focus on reducing the computational load and model size of speech separation models. ConvTasNet [19] replaces LSTM stack of TasNet [16] with a stack of 1-D dilated convolutions. A-FRCNN [14] fuses multi-scale features processed with CNN. Yet, CNN is not as effective as Transformer in modeling long-term dependency [20]. Tiny-sepformer [13] cuts down model size with parameter sharing [21]. However, it suffers from a significant accuracy drop of 4-5dB SI-SNRi in the WSJ0-2Mix dataset [22].

3. APPROACH

3.1. Architecture Overview

Figure 1 shows the overall model architecture of Papez. We take the time-domain masking approach [1, 19] composed of three modules: Encoder, Masking Module, and Decoder. First, the *Encoder* extracts a 2D spectrogram-like representation from the mixture signal. Second, the *Masking Module* estimates the mask over the 2D representation for each speaker. Finally, the *Decoder* uses the mask and 2D representation to reconstruct the clean speech of each speaker.

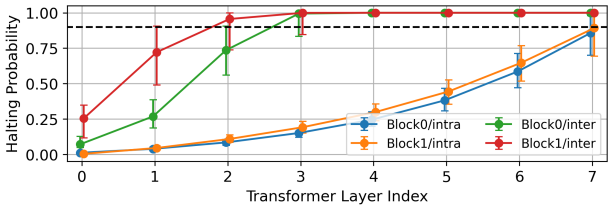

Fig. 2: Redundancy of input tokens in the Sepformer's dual-path process. The black dashed line indicates the threshold $P_{th} = 0.9$ of adaptively pruning the redundant token.

	Time Complexity	Params
Intra + Inter-T	$\mathcal{O}(N\sqrt{N}H^2)$	$8H^2$
Intra + $\frac{1}{S}$ Inter-T	$\mathcal{O}(\frac{N\sqrt{N}}{\sqrt{S}}H^2)$	$(4+\frac{4}{S})H^2$
Intra + AWM	$\mathcal{O}(\frac{N}{K}(K+M)^2H^2)$ $\approx 2NKH^2$	$4H^2$
Intra Only	$2NKH^2$	$4H^2$

Table 1: AWM vs. Inter-Transformer. N: # input tokens, M: # memory tokens, K: chunk size, H: token size, and S: depth ratio of intra- and inter-Transformer. $\approx$ when $M \ll K$.

Encoder and Decoder. The encoder is a two-layer downsampling 1D convolutional network, which is a sequence of 1-D convolution, instance normalization (IN), ReLU, and point-wise convolution. Similarly, the decoder is a two-layer upsampling convolutional network consisting of point-wise convolution, IN, ReLU, and 1-D transposed convolution.

Masking Module. An encoded mixture signal transforms into mask estimates through three stages. First, the *Embedding network* embeds the 2D signal representation into a sequence of tokens. Second, the *AWM transformer* process the token sequence. Finally, the *Mask Generation network* generates masks for each speaker. Embedding and Mask Generation networks are both a two-layer fully-connected feed-forward network with PReLU activation, and the $\tanh$ activation at the end of the Mask Generation network.

3.2. Auditory Working Memory (AWM) Transformer

We find that the inter-chunk transformer of the dual-path process is often redundant : as shown in Figure 2, when we apply our Adaptive Token Pruning (Section 3.3) to the Sepformer, most of the inter-transformer's tokens were quickly pruned after the first 2,3 layers. Similar observations were found in [3] as well. However, naïvely reducing the depth of inter-transformer does not change the asymptotic self-attention complexity $\mathcal{O}(N\sqrt{N})$ for input size N (See Table 1.).

In this light, we propose *Auditory Working Memory (AWM) Transformer*, a new transformer architecture augmented with a small-sized short-term memory to replace the compute-intensive inter-transformer. This is realized by special *memory tokens* distinct from input sequence tokens. In detail, the memory tokens are concatenated in front of every sequence chunk to globally attend the input sequence. After the multi-head attention mechanism, the memory tokens of each chunk are averaged to aggregate the global information. Right side of Figure 1 illustrates the detailed mechanism of

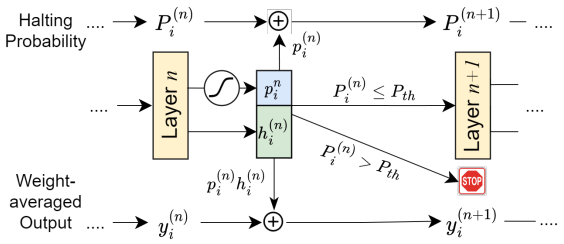

Fig. 3: Operation of Adaptive Token Pruning.

the AWM transformer layer. Note that chunking is only applied to multi-head attention, reducing the FFN's latency to nearly 50% since it halves the number of input tokens.

Our idea is bio-inspired; humans store auditory information like timbre, tone, and pitch [23] in a dedicated AWM for up to a few seconds [7], which is pivotal for auditory discrimination [24]. Also, the auditory cortex is interconnected with AWM areas (e.g., hippocampus), and the AWM is consistently maintained and retrieved along with auditory processing [8]. AWM has two advantages over the dual-path process. First, the time complexity is linear to input size, smaller than the transformer-based dual-path process's super-linear complexity as shown in Table 1. Second, it is fully parallelizable with intra-chunk processing.

3.3. Adaptive Token Pruning

In a mixture of speech, some segments are more straightforward to separate than others, which can be exploited for faster processing. For example, silent or non-overlapping speech segments are much easier to separate than overlapped speeches, which consist of 80% of audio inputs in a real-world scenario [6]. Also, mixtures from same-gender speakers are much more difficult to separate than different-gender case [22]. However, existing speech separation models employ a fixed processing path regardless of the input content, incurring unnecessary computation overhead.

Inspired by [11], we prune redundant tokens from the input sequence with the Adaptive Token Pruning technique. Specifically, each input token self-determines its redundancy and adaptively prunes itself (Figure 3). Let an input token sequence $\{h_i^{(0)}\}_{i=1}^{T}$ of each input token $h_i^{(0)} \in R^H$, its transformation $f_T : R^H \rightarrow R^H$ (in our case, a transformer layer), and a differentiable estimator $f_p : R^H \rightarrow (0, 1)$. The i-th output token $y_i^{(N)}$ is defined,

$$h_i^{(n)} = f_T\big(h_i^{(n-1)} | h_1^{(n-1)}, h_2^{(n-1)}, \cdots\big), \quad \text{if } P_i^{(n-1)} \le P_{th}$$

$$p_i^{(n)} = f_p(h_i^{(n)}), \quad \text{if } P_i^{(n-1)} \le P_{th}$$

$$P_i^{(n)} = \begin{cases} P_i^{(n-1)} + p_i^{(n)}, & \text{if } P_i^{(n-1)} \le P_{th} \\ 1, & \text{otherwise} \end{cases}$$

$$y_i^{(n)} = \begin{cases} y_i^{(n-1)} + p_i h_i^{(n)}, & \text{if } P_i^{(n-1)} \le P_{th} \\ y_i^{(n-1)} + (1 - P_i^{(n-1)}) h_i^{(n)}, & \text{otherwise} \end{cases}$$

where $P_{th} \in [0, 1]$ is the halting threshold, $P_i^{(0)} = 0$, and N is the maximum f_T iterations. So each sequence token processes f_T until its halting probability P_i reaches the threshold. This enables dynamic scaling of the transformer's *width* and *depth*, i.e., the number of tokens and the transformer iterations, respectively. The estimator f_p has sigmoid output activation, and f_p is jointly trained with the transformer f_T.

Piggybacked Probability Estimator. Prior works used a separate single-layer fully-connected network as the probability estimator f_p [11, 12]. However, it incurs additional latency since it cannot be computed with the f_T in parallel. Instead, we piggyback f_p into the FFN of the transformer and the embedding module. Hence, the input and output of the transformer layer become a sequence of pairs of a token and its halting probability, i.e., $\{(h_i^{(n)}, p_i^{(n)})\}_{i=1}^{T}$.

Recurrent Transformer and Time-step Encoding. To reduce the model size, we employ recurrent transformer [12] architecture, which iterates a single parameter-shared transformer layer instead of stacking it. This reduces the number of parameters to $\frac{1}{N}$ compared to the Transformer with depth N. Opposed to [12] which adds the fixed time-step encoding vector to the tokens, we make the time-step information learnable by embedding it in the trainable affine parameters of the transformer's layer normalization. Essentially, it is equivalent to sharing only the linear layer weights of the transformer.

4. EXPERIMENT

4.1. Experimental Setup

Datasets. We use two datasets for evaluation: (i) WSJ0-Mix [22] and (ii) LibriMix [6] (*train-360* for training, and *test* for evaluation). The signal-to-noise ratio (SNR) of mixture is randomly selected in [-5,5] dB range.

Metrics. We use scale-invariant Signal-to-Noise Ratio improvements (SI-SNRi) [16] and Signal-to-Distortion Ratio improvements (SDRi) [30] for separation accuracy. We report the number of parameters and inference latency on Intel Xeon Silver 4114 CPU and NVIDIA Titan XP GPU.

Training Details. We set the AWM size as 16, maximum depth 16, chunk size 150, 8 attention heads, embedding token size 256 and 1024 hidden nodes of FFN. The Encoder and Decoder have kernel size 16 and stride 8. The model is trained for 100 epochs on LibriMix and 200 epochs on WSJ0 dataset, with 16-bit mixed precision. Batch size is set to 1. We use SI-SNR with utterance-level Permutation Invariant loss [31]. We use the AdamW [32] optimizer with 10^{-4} learning rate, 10^{-4} weight decay, and exponential learning rate decay in a factor of 0.98. Maximum L_2 norm of gradient is clipped with 1. We also use Dynamic Mixing (DM) [28] for augmentation. The entire training takes 220 hours on a Titan RTX GPU.

4.2. Comparison with Prior Works

Table 2 shows that Papez is significantly more efficient than computation-intensive high-performance baselines [1, 4, 17, 5, 28]. Papez is $2.20 \sim 4.75\times$ and $2.67 \sim 20.95\times$ faster on GPU, CPU, respectively. The separation accuracy of Papez is noticeably higher than computation-efficient baselines [13, 19, 14]. The SI-SNRi of Papez has increased $1.1 \sim 4.2$dB on the WSJ0-Mix [22] and $0.6 \sim 5.1$dB on the LibriMix [6]. Interestingly, Papez is even faster than efficient baselines [13, 14] of $1.67 \sim 2.45\times$ on GPU and

Models	Libri2Mix [6]		WSJ0-2Mix [22]		Params	Latency (5s, 8kHz input)	
	SISNRi (↑)	SDRi (↑)	SISNRi (↑)	SDRi (↑)	(M)	CPU (s)	GPU (ms)
DPCL [22]	5.9*	6.6*	10.8	11.2	13.6	0.62	111.80
uPIT-LSTM [25]	7.6*	8.2*	9.8	10.0	92.7	0.56	110.44
Chimera++ [26]	6.3*	7.0*	11.5	11.8	32.9	0.69	152.36
BLSTM-TasNet [16]	7.9*	8.7*	13.2	13.6	23.6	1.68	335.58
Two-step TDCN [27]	12.0*	12.5*	16.1	-	8.6	0.67	156.11
DPRNN [4]	14.1*	14.6*	18.8	19.0	2.7	4.36	207.76
Sandglasset [5]	-	-	20.8	21.0	2.3	6.89	155.02
DPT [17]	16.2	16.8	20.2	20.6	2.6	24.73	336.74
Wavesplit [28]	**19.5**	**20.0**	**21.0**	**21.2**	29.0	9.15	274.87
Sepformer [1]	19.2	19.4	20.4	20.5	26.0	3.16	168.45
ConvTasNet [19]	12.2	12.7	15.3	15.6	5.6	**0.40**	**34.03**
A-FRCNN-16 [14]	16.7	17.2	18.3	18.6	6.1	1.95	124.52
A-FRCNN-16(sum) [14]	16.2	17.2	17.9	18.3	**1.7**	1.24	118.98
Tiny-Sepformer [13]	-	-	15.1	16.1	20.0	3.09	173.39
Tiny-Sepformer-S [13]	-	-	15.2	16.0	5.3	2.96	173.37
Papez	17.2	17.6	19.2	19.5	**1.47**	**1.18**	**70.88**
Papez + DM	**17.3**	**17.7**	**19.4**	**19.7**			

Table 2: Separation performance and computational efficiency of our **Papez** model compared with prior works. Latency of baselines measured from asteroid [29] toolkit with their code [1, 5, 14, 29] and ours [13]. (*) denotes results reported by [14].

	Method	WSJ0-2Mix		Params	Latency
		SISNRi(↑)	SDRi(↑)	(M)	(ms)
	Sepformer	20.4	20.5	26.0	168.31
(1)	+ Recurrence	17.1	17.4	2.25	163.67
(2)	+ Pruning	17.9	18.1	2.38	95.22
(3)	+ No Inter-T	15.6	16.1	1.52	82.97
(4)	+ AWM	19.0	19.3	1.52	86.18
(5)	+ Time Enc.	**19.4**	**19.7**	1.54	86.53
(6)	+ Piggyback	19.3	19.6	**1.47**	**80.06**
	$K = 250$	**19.3**	**19.6**	1.47	79.94
	$K = 150$	19.2	19.5	1.47	70.41
	$K = 100$	19.1	19.3	1.47	**67.28**
	$K = 50$	19.0	19.3	1.47	68.34
	$M = 16$	**19.3**	**19.6**	1.47	80.06
	$M = 8$	18.6	18.9	1.47	79.66
	$M = 4$	18.3	18.6	1.47	77.42
	$M = 2$	18.0	18.4	1.47	76.52

Table 3: Effect of our key techniques and hyperparameters. *No Inter-T* setup (3) removes the inter-transformer from (2). K is chunk size, and M is the number of AWM slots.

$1.05 \sim 2.61\times$ on CPU. Note that Tiny-Sepformer [13] focuses on reducing the model size of the Sepformer [1], and the latency rather increased. The model size of Papez is the smallest, which is $17.7\times$ smaller than the Sepformer baseline. Even if we scale down the transformer depth of the Sepformer to 3 or 4 with FFN size 2048, Papez is $1.26\times$, $1.65\times$ faster with $11.01\times$, and $14.59\times$ smaller model size, respectively.

4.3. Ablation Study

Table 3 shows that our techniques either contributes to efficiency ((1) $\sim$ (3), (6)) or separation accuracy ((2), (4), (5)). Note that our working memory technique greatly enhances the accuracy with minimal latency overhead of 3.8% (4). Also, our adaptive token pruning noticeably cuts down the latency to 58.1% while boosting the performance (2).

We also investigate the effect of two key hyperparameters: chunk size and AWM size. Table 3 shows that scaling down the chunk size notably decreases the latency, yet its impact on performance is marginal. The reason the latency increases from $K = 100 \rightarrow 50$ is that the latency proportion of the attention mechanism becomes insignificant with a small chunk size. In contrast, reducing the AWM size greatly diminishes the performance. This demonstrates that our AWM plays a pivotal role in speech separation performance.

5. CONCLUSION

We proposed Papez, an efficient and lightweight single-channel speech separation model. For computational efficiency, we substituted inter-transformer with small-sized working memory, and adaptively pruned redundant tokens from the input sequence. Experimental results on WSJ0-2Mix [22] and Libri2Mix [6] datasets demonstrate that our model is computationally efficient yet highly performant.

6. ACKNOWLEDGEMENT

This work was supported by the National Research Foundation of Korea(NRF) grant funded by the Korea government(MIST) (No. 2022R1A2C3008495).

7. REFERENCES

[1] C. Subakan, M. Ravanelli, S. Cornell, M. Bronzi, and J. Zhong, "Attention is all you need in speech separation," in *ICASSP 2021*, pp. 21–25.

[2] S. Lutati, E. Nachmani, and L. Wolf, "Sepit approaching a single channel speech separation bound," *Proc. Interspeech*, 2022.

[3] J. Rixen and M. Renz, "Qdpn-quasi-dual-path network for single-channel speech separation," *Proc. Interspeech*, pp. 5353–5357, 2022.

[4] Y. Luo, Z. Chen, and T. Yoshioka, "Dual-path rnn: efficient long sequence modeling for time-domain single-channel speech separation," in *ICASSP 2020*, pp. 46–50.

[5] M. Lam, J. Wang, D. Su, and D. Yu, "Sandglasset: A light multi-granularity self-attentive network for time-domain speech separation," in *ICASSP*, 2021, pp. 5759–5763.

[6] J. Cosentino, M. Pariente, S. Cornell, A. Deleforge, and E. Vincent, "Librimix: An open-source dataset for generalizable speech separation," *arXiv preprint arXiv:2005.11262*, 2020.

[7] T. Pasternak and M. Greenlee, "Working memory in primate sensory systems," *Nature Reviews Neuroscience*, vol. 6, no. 2, pp. 97–107, 2005.

[8] S. Kumar, S. Joseph, P. Gander, N. Barascud, A. Halpern, and T. Griffiths, "A brain system for auditory working memory," *Journal of Neuroscience*, vol. 36, no. 16, pp. 4492–4505, 2016.

[9] J. Ainslie, S. Ontañón, C. Alberti, V. Cvicek, Z. Fisher, P. Pham, A. Ravula, S. K. Sanghai, Q. Wang, and L. Yang, "Etc: Encoding long and structured inputs in transformers," in *EMNLP*, 2020.

[10] M. Zaheer, G. Guruganesh, K. Dubey, J. Ainslie, C. Alberti, S. Ontañón, P. Pham, A. Ravula, Q. Wang, L. Yang, and A. Ahmed, "Big bird: Transformers for longer sequences," *ArXiv*, vol. abs/2007.14062, 2020.

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

[12] M. Dehghani, S. Gouws, O. Vinyals, J. Uszkoreit, and L. Kaiser, "Universal transformers," in *ICLR*, 2018.

[13] J. Luo, J. Wang, N. Cheng, E. Xiao, X. Zhang, and J. Xiao, "Tiny-sepformer: A tiny time-domain transformer network for speech separation," *Proc. Interspeech*, 2022.

[14] X. Hu, K. Li, W. Zhang, Y. Luo, J. Lemercier, and T. Gerkmann, "Speech separation using an asynchronous fully recurrent convolutional neural network," *NeurIPS*, vol. 34, pp. 22509–22522, 2021.

[15] G. Logeshwari and G. S. Anandha Mala, "A survey on single channel speech separation," 2012.

[16] Y. Luo and N. Mesgarani, "Tasnet: time-domain audio separation network for real-time, single-channel speech separation," in *ICASSP 2018*, pp. 696–700.

[17] J. Chen, Q. Mao, and D. Liu, "Dual-path transformer network: Direct context-aware modeling for end-to-end monaural speech separation," *Proc. Interspeech*, pp. 2642–2646, 2020.

[18] T. Lin, Y. Wang, X. Liu, and X. Qiu, "A survey of transformers," *arXiv preprint arXiv:2106.04554*, 2021.

[19] Y. Luo and N. 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.

[20] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," *NIPS*, vol. 30, 2017.

[21] Z. Wu, Z. Liu, J. Lin, Y. Lin, and S. Han, "Lite transformer with long-short range attention," *ICLR*, 2020.

[22] J. Hershey, Z. Chen, J. Le Roux, and S. Watanabe, "Deep clustering: Discriminative embeddings for segmentation and separation," in *ICASSP 2016*, pp. 31–35.

[23] K. Schulze and B. Tillmann, "Working memory for pitch, timbre, and words," *Memory*, vol. 21, no. 3, pp. 377–395, 2013.

[24] Y. Zhang, D. Moore, J. Guiraud, K. Molloy, T. Yan, and S. Amitay, "Auditory discrimination learning: Role of working memory," *PloS one*, vol. 11, no. 1, pp. e0147320, 2016.

[25] M. Kolbæk, D. Yu, Z. Tan, and J. Jensen, "Multitalker speech separation with utterance-level permutation invariant training of deep recurrent neural networks," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 25, no. 10, pp. 1901–1913, 2017.

[26] Z. Wang, J. Le Roux, and J. Hershey, "Alternative objective functions for deep clustering," in *ICASSP 2018*, pp. 686–690.

[27] E. Tzinis, S. Venkataramani, Z. Wang, C. Subakan, and P. Smaragdis, "Two-step sound source separation: Training on learned latent targets," in *ICASSP 2020*, pp. 31–35.

[28] N. Zeghidour and D. Grangier, "Wavesplit: End-to-end speech separation by speaker clustering," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 29, pp. 2840–2849, 2021.

[29] M. Pariente, S. Cornell, J. Cosentino, S. Sivasankaran, E. Tzinis, Je. Heitkaemper, M. Olvera, Fabian-R. Stöter, M. Hu, J. Martín-Doñas, et al., "Asteroid: the pytorch-based audio source separation toolkit for researchers," *Proc. Interspeech*, 2020.

[30] E. Vincent, R. Gribonval, and C. Fevotte, "Performance measurement in blind audio source separation," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 14, no. 4, pp. 1462–1469, 2006.

[31] D. Yu, M. Kolbæk, Z. Tan, and J. Jensen, "Permutation invariant training of deep models for speaker-independent multi-talker speech separation," in *ICASSP 2017*, pp. 241–245.

[32] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," in *ICLR*, 2018.