NATIONAL RESEARCH UNIVERSITY
HIGHER SCHOOL OF ECONOMICS

Faculty of Computer Science
Bachelor's Programme "Applied Mathematics and Informatics"

UDC 004.832.25

Research Project Report on the Topic:

Context Mechanisms for Convolutional Models in Real-Time Audio Separation Problem

Submitted by the Student:

group #БПМИ226, 2rd year of study Ignatov Maxim Alexeevich

Approved by the Project Supervisor:

Kaledin Maxim Lvovich
Associate Professor
Big Data and Information Retrieval School, Faculty of Computer Science, HSE University

Moscow 2024

Contents

Abstract

This coursework delves into the implementation, enhancement and context examination of CNN models for the efficient separation of audio sources in real-time with a particular focus on maintaining the quality of audio metrics such as SI-SDR and PesQ. Utilizing the SpEx+ [1] model as a baseline. I have provided several versions of the model and conducted several experiments aimed at improving the model at our particular task. Models are trained on mixed LibriSpeech [7] dataset unlike the model given in the paper [1].

Аннотация

Курсовая работа будет направлена на улучшение и реализацию сверточных нейросетей для проблемы разделения аудио источников в режиме реального времени с особым акцентом на поддержание качества важных для чистоты аудио метрик, как SI-SDR и PesQ. В качестве бейзлайна была использована модель SpEx+ [1]. Данная работа приводит несколько версий данной модели, а также модели, полученные с помощью экспериментов для улучшения качества в задаче стриминга. Все модели были обучены с помощью семплов, полученных путем смешивания записей из датасета LibriSpeech [7].

Keywords

deep learning, speaker extraction, multi-scale, multi-task learning, source separation

1 Introduction

1.1 Problem

Speaker separation problem has different variations like speaker-independent blind source/speech separation (BSS) which aims to separate diverse sources of sound or speakers from a sound/speech mixture with unknown number of noise sources or speakers. Usually, you need to know exact number of audio sources in mixture to separate them. Target source separation takes a different strategy to the problem. It's trying to extract target speaker's voice from audio using reference speech of the target, thus you need to have more information about speaker than in blind source separation.

There are some various architectures that are used in this problem. Most recent ones are attention based transformers like Demucs[8] (Music Source Separation). Nevertheless CNN (or Encoder-CNN-Decoder) models like Conv-TasNet [5] and SpEx+ [1] demonstrate State-of-the-Art results. CNN models are going to be used because of the simplicity of their structure and the ability to easily compress them.

1.2 Problem statement

Given reference signal $r(t)$ and mixed signal $x(t)$ that is a linear combination of individual signals $s_i(x)$, including target signal. The task is an optimization of the loss function [1]

$$\mathcal{L}(\Theta \mid x, \hat{s}, s, I) = \mathcal{L}_{SI-SDR} + \gamma \mathcal{L}_{CE} \tag{1}$$

were Θ represents model parameters, $\hat{s}, s$ are the estimated and target signal respectfully. Their means are normalized to zero. I is a one-hot vector representing the true class labels for the target speaker, and γ is a scaling parameter.

$$\begin{gathered} \mathcal{L}_{SI-SDR} = -[(1-\alpha-\beta)\rho(s_1, \hat{s}) + \alpha\rho(s_2, \hat{s}) + \beta\rho(s_3, \hat{s})] \\ \rho(s_1, \hat{s}) = 20 \log_{10} \frac{\|(\hat{s}^T s / s^T s) \cdot s\|}{\|(\hat{s}^T s / s^T s) \cdot s - \hat{s}\|} \end{gathered} \tag{2}$$

s_1, s_2, s_3 are estimated signals from multiscale speech decoder with short, mid, long window size respectfully. $\mathcal{L}_{CE}$ is the cross-entropy loss for speaker classification, which is defined as

$$\mathcal{L}_{CE} = -\sum_{k=1}^{N_s} I_k \log_{10}(\sigma(W \cdot v)_k) \tag{3}$$

were N_s is a number of speakers in speaker classification, I_k is a true class label of a speaker. W represents a weight matrix, $\sigma(\cdot)$ is a softmax function and $\sigma(W \cdot v)$ represents the predicted probability.
Real-time audio separation specific in that the signal $s(t)$ is being provided by fragments of short lenght ($\sim$ 300 ms).

1.3 Applications

This task is not only critical for enhancing the clarity and quality of audio in communication technologies but also plays a significant role in various applications ranging from automated transcription services to assistive hearing devices. For example, imagine if voice control systems can response differently according to the voice signature of a speaker. Signal separation is being used in a lot of fields beyond sound. Some of them are Medical imaging (separating magnetic signals from magnetoencephalography and signals from external sources) and Image processing. The flexibility of the model architecture helps to project the solution to a more general class of problems.

2 Literature review

2.1 Analysis of current work

The majority of the previous methods have formulated the separation problem through the time-frequency representation (or spectrogram) of the mixed signal, which has several drawbacks, including the decoupling of the phase and magnitude of the signal before separation step and the long latency in calculating the spectrograms. Furthermore, accurate reconstruction of the phase of the clean sources is a non-trivial procedure, so usually T-F based networks are using phase of the mixed spectrogram or applying masks [2], [4] (those methods show small improvement of metrics, but at the cost of a noticeable complexity increase of the model).
Models like Conv-TasNet [5] and SpEx+ [1] propose a fully-convolutional time-domain solution. They don't separate phase from signal and uses 1-D CNN for waveform instead of 2-D for spectrogram. Another approach used in TasNet [6]. Deep LSTM networks have high computational cost and show a worse result comparing to CNN.

2.2 Specifics of the architecture

SpEx+ [1] consists of speech encoder, speaker encoder, speaker extractor, and speech decoder. The problem of not full time-frequency solution in SpEx [10] was fixed by weight-shared speech encoders, also called twin speech encoders.

2.2.1 Twin speech encoders

According to [1] weight sharing used to project mixture and target speech into common latent space. The speech encoders (and some more blocks) run separately, so it can be useful for paralleling the computations. Each encoder consists of parallel CNN blocks with different filter lengths with ReLU activation function.

2.2.2 Speaker encoder

Speaker encoder is designed to extract speaker embedding of the target speaker from the reference speech. It consists of 1-D CNN which takes coefficients from Twin speech encoders following by several residual network (ResNet) blocks. Then a 1-D CNN is used to project the representations into a same dimension as speech embedding together with a mean pooling operation.

ResNet blocks consists of two convolutional layers with 1×1 kernel and 1-D max-pooling layer. Batch Norm (BN) and PReLU are used for normalizing outputs of CNNs. After the second CNN, skip connection is used to add the inputs to the outputs of BN layer.

2.2.3 Speaker extractor and speech decoder

Speaker extractor uses several stacked TCN [5] blocks. First of every stacked TCN concatenates it's input with the embeddings from Speaker encoder. TCN structure consists of 1×1 CNN with PReLU activation and Layer normalization following by depth-wise separable convolution. This structure provides lots of possibilities for scaling the model.

2.3 Methods for weight reduction

Pruning and quantization are established techniques for enhancing the efficiency and reducing the storage demands of convolutional neural networks. Pruning eliminates weights that are close to zero in tensors ineffective connections among neurons in neighboring layers. Quantization decreases the weight precision by replacing them with numerically close values that occupy less storage space. There are more advanced methods like Knowledge Distillation [3]. The goal is to

teach lighter (student model) to reproduce the output of the bigger (teacher model). As already mentioned, our goal is to reduce weight while maintaining the normal performance of the model.

2.4 LSTM and Conv-LSTM

Long Short-Term Memory (LSTM) networks have become a staple in sequence modeling tasks, including audio processing. Their ability to learn long-term dependencies makes them well-suited for capturing temporal information in audio signals, which is crucial for tasks like source separation. In the context of real-time audio separation, LSTMs excel at remembering information through fragments of audio, allowing them to capture the evolution of individual sound sources and their relationships within a mixture. This is particularly helpful for separating sounds with overlapping frequency components.

However, Conv-LSTMs [9] is more suitable choice for this task because it replaces the standard LSTM's matrix multiplication with convolutional operations, altering the cell's internal processing. Furthermore, it allows variable input size due to the convolutional structure and reduce latency due to the smaller number of trainable parameters.

Equations for the single ConvLSTM cell looks like:

$$
\begin{aligned}
i_t &= \sigma\left(W_{xi} * X_t + W_{hi} * H_{t-1} + W_{ci} \circ C_{t-1} + b_i\right) \\
f_t &= \sigma\left(W_{xf} * X_t + W_{hf} * H_{t-1} + W_{cf} \circ C_{t-1} + b_f\right) \\
C_t &= f_t \circ C_{t-1} + i_t \circ \tanh\left(W_{xc} * X_t + W_{hc} * H_{t-1} + b_c\right) \\
o_t &= \sigma\left(W_{xo} * X_t + W_{ho} * H_{t-1} + W_{co} \circ C_t + b_o\right) \\
H_t &= o_t \circ \tanh\left(C_t\right)
\end{aligned}
\tag{4}
$$

Where $*$ denotes the convolution operation and $\circ$ denotes the Hadamard product.

At this moment, it is difficult to find examples of using Conv-LSTM in this particular task, which are confirmed by published articles.

3 Model Implementation

Implementation of the SpEx+ model was based on the corresponding paper [1] and information from HSE Deep Learning for Audio (DLA) course.

3.1 Pipeline

Open implementation of SpEx+ wasn't suitable for my task due to the differences in structure of the dataset and poor scalability of the trainer, logger and other important parts of pipeline. Implementation of an easily changeable pipeline for each experiment and dataset was a big part of this work. Final version has enough instruments to monitor all the metrics during training and validation. Also, pipeline uses different techniques for more consistent trainings like gradient accumulation and gradient clipping (good for LSTMs). All experiments were conducted on the Kaggle platform on single 16 Gb Nvidia Tesla P100.

4 Experemental Setup

4.1 Dataset

I simulated a two-speakers dataset LibriSpeech-mix at sampling rate of 16kHz based on original LibriSpeech dataset. The collected dataset contains 251 speakers and was divided to three sets: train set (30000 triplets), test set (2000 triplets) and development (validation) set (1000 triplets). Each triplet consists of target speaker utterance, reference speaker utterance and two-speaker mixed utterance. Going into details, in train and development set the silence was cut from the dataset in the samples, and the recordings themselves were cut to three seconds. Reference speech of the target speaker is a randomly selected utterance that is different from used in the mixture.

The small size of the validation sample is characterized by the fact that in conditions of the limited computing resources it was necessary to get the maximum out of the training epochs.

Actually, not all the data in the dataset is congeneric. Initially, the data from LibriSpeech is cut into chunks of the desired size, so the data from the ends of the record may have small amount of necessary information and high silence time which is harmful to the reference audio.

4.2 Encoder

This implementation of the SpEx+ using weight-shared encoder for mixed and reference waveform. Encoded embeddings from three encoders with different filter lengths then concatenated into one tensor. Padding is used in mid and long encoders to match the output shape of short decoder.

$$\begin{aligned} X &= [X_{short}, X_{mid}, X_{long}] = e(\theta \mid x, L_1, L_2, L_3) \\ Y &= [Y_{short}, Y_{mid}, Y_{long}] = e(\theta \mid y, L_1, L_2, L_3) \end{aligned} \tag{5}$$

Where encoder formula looks like:

$$e(\theta \mid x, L_1, L_2, L_3) = ReLu([W_{L_1}(\theta_1) * x, W_{L_2}(\theta_2) * p(x), W_{L_3}(\theta_3) * p(x)]) \tag{6}$$

$*$ denotes the 1-d convolutional operation, $p(\cdot)$ denotes padding operation.
All of my models use the same encoder parameters. Due to limited resources, I was unable to test the model with a different set of parameters. It may be beneficial to change their size due to the different sample rate in our dataset.

4.3 Feature extraction

In this part model calculates reference speech embeddings and embeddings of the mixed audio.

4.3.1 Reference embeddings

To calculate reference embeddings model used Res-Net blocks, it prevents model from gradient vanishing while at the same time, the architecture is good for highlighting differences between speaker embeddings (will be shown later). Every Res-Net block looks like:

$$f(x) = MaxPool(PReLU(BN(W_2 * BN(PReLU(W_1 * x))) + x)) \tag{7}$$

Where $PReLU(\cdot)$ is a parametric element-wise ReLU, $BN(\cdot)$ is a batch normalization function, $*$ is a 1-d convolutional operation. W_i is a 1×1 convolutional kernel.

4.4 Baseline Hyperparameters

All models were trained on 3-second long segments with Adam optimizer. The filter lengths of convolutions in speech encoder and decoder were $L_1 = 2.5ms, L_2 = 10ms, L_3 = 20ms$ for speech of 16kHz sampling rate, respectively. Baseline models uses 3 consecutive Res-Net blocks for calculating reference embeddings and 4 stacked TCN blocks with 8 TCN modules in each block. Loss hyperparameters are set for $\alpha = 0.1, \beta = 0.1, \gamma = 0.5$. All metrics are using only prediction from the short decoder ($s_1(t)$ according to [1]).

4.5 Baseline Trainings

I've trained several models with same architecture and different hyperparameters to get the best result for future tasks. Models in the following table are the best that I was able to get with my resources:

id	Description	Epoch Length	Total Epochs	lr	Si-SDR[1]	PesQ[1]
1	SpEx+	5000	11	$1e^{-4}$	5.059	1.461
2	SpEx+	5000	12	$1e^{-5}$	7.807	1.708
3	**Tuning pretrained # 2**	6000	20	$2.5e^{-5}$	9.163	1.853
4	Tuning pretrained # 1	2000	16	$1e^{-3}$	5.503	1.481

[1] Average score on test dataset.

Table 4.1: SpEx+ offline

Grad Accumulation parameter was set at 1. Changing other hyperparameters of optimizer and model did not bring any visible improvement in consistency and speed of the learning. Early stop was set at 3 epochs if the model's loss would stop improving on validation epoch, but during the whole training, model's metrics were increasing consistently. It can be concluded that if we continue training, we can achieve higher metrics.

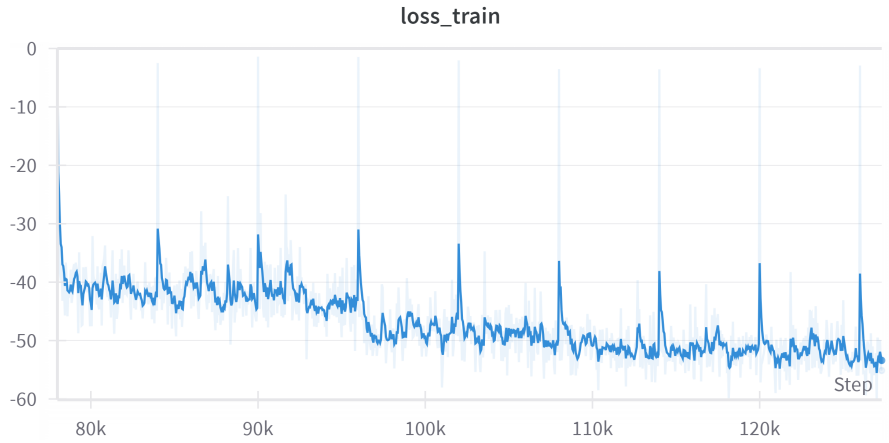


Figure 4.1: Training loss of the model # 3

5 Switching from an offline task to streaming

5.1 Reformulation of the optimization problem

Now mixed signal $x(t)$ is fed to the model in small chunks while reference signal $r(t)$ remains the same. Optimization problem can be formulated in two ways:

1) Optimization of the loss function (1) on every chunk. This problem is equivalent to improving the original model on short-length samples. The resulting loss is calculated as sum of the losses on each chunk:

$$\mathcal{L}(\Theta \mid x, \hat{s}, s, I) = \sum_{x=\bigcup_{i=1}^{N} x_i} \mathcal{L}(\Theta \mid x_i, \hat{s}_i, s_i, I) \tag{8}$$

2) Optimization of the loss function (1) on whole audio while predictions $\hat{s}_i(t)$ are made on small chunks and then concatenate to one audio $s(t) = \bigcup_{i=1}^{N} \hat{s}_i(t)$. This work reviews only this optimization problem.

5.2 Improving consistency of the predictions

During prediction on small chunks, values at adjacent ends between chunks may differ. To improve the quality of predictions model uses overlap $O = 10ms$ in audio and then concatenate predictions with fading on the ends.

5.3 Experiments and testing the basic SpEx+ model

Several models have been trained for this task. The table below shows the values for validating different models in the streaming task with $300ms$ chunks and overlap $O = 10ms$.

id	Description	Si-SDR[1]	PesQ[1]
3	Best SpEx+ on offline problem	3.383	1.486
5	SpEx+ trained purely on streaming	4.524	1.393
6	**Tuning # 3 on streaming task**	7.261	1.726

[1] Average score on test dataset.

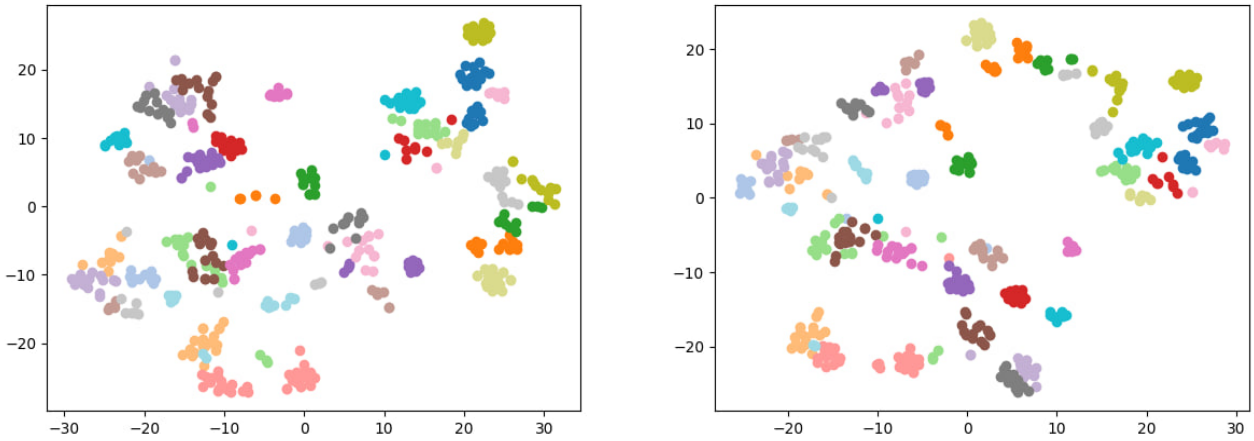
Table 5.1: SpEx+ streaming

Model # 5 was trained with the same hyperparameters as # 3, but on the streaming task. Similar to Model 3, its training can be continued to obtain better metrics.

Model # 6 shows the best result among all trained models on this task. The similarity of the problems helps to get the best result with the least learning time.

5.4 Speaker embedding investigation

To improve the quality of the model, it is worth knowing how model calculates embeddings of the audio. t-distributed Stochastic Neighbor Embedding (t-SNE) helps to understand how the model distinguishes between voices. The proximity of the points shows the proximity of the embeddings.



(a) model # 5 (b) model # 3

Figure 5.1: t-SNE for speaker embeddings using 200 samples of a test dataset

Both models distinguish well between different speakers. The points corresponding to different utterances of one speaker are closer to each other than to other points.

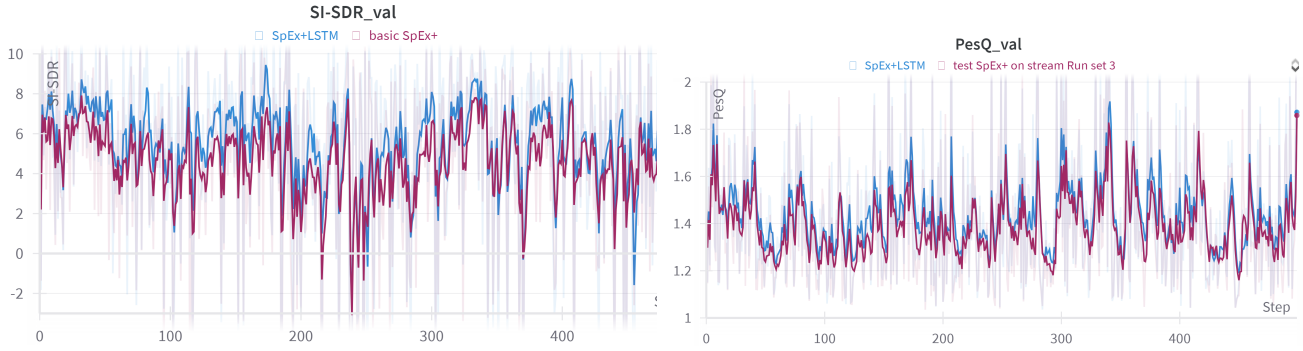
6 Context Mechanism

Because in the task of real-time source separation we get mixed data by chunks, we can use context from the previous chunks to separate necessary signal. To store the current state, we can use LSTM. I have embed an Conv-LSTM block into each set of stacked TCN blocks. Every first temporal convolutional network accepts speaker embedding and the learned representations over the mixture speech as input. Before getting into the TCN block, the input gets into the LSTM. I believe that this structure will help model to learn deeper features about target audio.
During the short training, SpEx+LSTM showed better results compared to the regular SpEx+, which was trained for the same amount of time.

id	Description	Si-SDR[1]	PesQ[1]
5	SpEx+ trained purely on streaming	4.524	1.393
7	SpEx+LSTM	5.396	1.440
8	**tuning # 7**	6.392	1.633

[1] Average score on test dataset.

Table 6.1: SpEx+ vs SpEx+LSTM



(a) SI-SDR

(b) PesQ

Figure 6.1: Plot of the metrics on the test dataset

7 Conclusions

This course work investigated the efficacy of State-Of-The-Art source separation model SpEx+ within the context of streaming applications. A new method has also been proposed to improve performance in this task. The current results have not shown much improvement in SI-SDR and PesQ metrics, nevertheless, it has been shown that the using of context mechanism and LSTM networks in Real-Time Audio Separation Problem has a positive effect on target metrics. Further work will be aimed at improving current models by fine tuning them on more powerful systems and a deeper study of the context mechanisms of nerual networks using RNN blocks and, in particular, LSTM.

References

[1] Meng Ge, Chenglin Xu, Longbiao Wang, Eng Siong Chng, Jianwu Dang, and Haizhou Li. "Spex+: A complete time domain speaker extraction network". In: *Proc. of INTERSPEECH*. 2020, pp. 1406–1410.

[2] Xiang Hao, Xiangdong Su, Radu Horaud, and Xiaofei Li. "Fullsubnet: A Full-Band and Sub-Band Fusion Model for Real-Time Single-Channel Speech Enhancement". In: *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2021, pp. 6633–6637. DOI: 10.1109/ICASSP39728.2021.9414177.

[3] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. *Distilling the Knowledge in a Neural Network*. 2015. arXiv: 1503.02531 [stat.ML].

[4] Yx Hu, Yun Liu, Shubo Lv, Mengtao Xing, Shimin Zhang, Yihui Fu, Jian Wu, Bihong Zhang, and Lei Xie. "DCCRN: Deep Complex Convolution Recurrent Network for Phase-Aware Speech Enhancement". In: Aug. 2020. DOI: 10.21437/Interspeech.2020-2537.

[5] Yi Luo and Nima Mesgarani. "Conv-TasNet: Surpassing Ideal Time–Frequency Magnitude Masking for Speech Separation". In: *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 27.8 (Aug. 2019), pp. 1256–1266. ISSN: 2329-9304. DOI: 10.1109/taslp.2019.2915167. URL: http://dx.doi.org/10.1109/TASLP.2019.2915167.

[6] Yi Luo and Nima Mesgarani. *TasNet: time-domain audio separation network for real-time, single-channel speech separation*. 2018. arXiv: 1711.00541 [cs.SD].

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

[8] Simon Rouard, Francisco Massa, and Alexandre Défossez. "Hybrid Transformers for Music Source Separation". In: *ICASSP 23*. 2023.

[9] Xingjian Shi, Zhourong Chen, Hao Wang, Dit-Yan Yeung, Wai-kin Wong, and Wang-chun Woo. *Convolutional LSTM Network: A Machine Learning Approach for Precipitation Nowcasting*. 2015. arXiv: 1506.04214 [cs.CV].

[10] Chenglin Xu, Wei Rao, Eng Siong Chng, and Haizhou Li. "SpEx: Multi-scale time domain speaker extraction network". In: *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 28 (2020), pp. 1370–1384.