Semantic Hearing: Programming Acoustic Scenes with Binaural Hearables

Bandhav Veluri*
Paul G. Allen School, University of Washington, Seattle, WA, USA
bandhav@cs.washington.edu

Malek Itani*
Paul G. Allen School, University of Washington, Seattle, WA, USA
malek@cs.washington.edu

Justin Chan
Paul G. Allen School, University of Washington, Seattle, WA, USA
jucha@cs.washington.edu

Takuya Yoshioka
Microsoft, One Microsoft Way, Redmond, WA, USA
tayoshio@microsoft.com

Shyamnath Gollakota
Paul G. Allen School, University of Washington, Seattle, WA, USA
gshyam@cs.washington.edu

Figure 1: Semantic hearing applications. a) Users wearing binaural headsets can attend to speech while blocking out only the vacuum cleaner noise, b) block out street chatter and focus on the sounds of birds chirping, c) block out construction noise yet hear car honks, and d) a meditating user could use headsets to block out traffic noise outside yet hear alarm clock sounds.

ABSTRACT

Imagine being able to listen to the birds chirping in a park without hearing the chatter from other hikers, or being able to block out traffic noise on a busy street while still being able to hear emergency sirens and car honks. We introduce *semantic hearing*, a novel capability for hearable devices that enables them to, in real-time, focus on, or ignore, specific sounds from real-world environments, while also preserving the spatial cues. To achieve this, we make two technical contributions: 1) we present the first neural network that can achieve binaural target sound extraction in the presence of interfering sounds and background noise, and 2) we design a training methodology that allows our system to generalize to real-world use. Results show that our system can operate with 20 sound classes and that our transformer-based network has a runtime of 6.56 ms on a connected smartphone. In-the-wild evaluation with participants in previously unseen indoor and outdoor scenarios shows that our proof-of-concept system can extract the target sounds and generalize to preserve the spatial cues in its binaural output.

Project page with code: https://semantichearing.cs.washington.edu

*Co-primary student authors

UIST '23, October 29–November 01, 2023, San Francisco, CA, USA

ACM ISBN 979-8-4007-0132-0/23/10.
https://doi.org/10.1145/3586183.3606779

CCS CONCEPTS

• **Computing methodologies** → **Machine learning**; • **Human-centered computing** → **Sound-based input / output**; • **Hardware** → **Emerging interfaces**.

KEYWORDS

Spatial computing, binaural target sound extraction, earable computing, noise cancellation, attention, causal neural networks

ACM Reference Format:
Bandhav Veluri, Malek Itani, Justin Chan, Takuya Yoshioka, and Shyamnath Gollakota. 2023. Semantic Hearing: Programming Acoustic Scenes with Binaural Hearables. In *The 36th Annual ACM Symposium on User Interface Software and Technology (UIST '23), October 29–November 01, 2023, San Francisco, CA, USA.* ACM, New York, NY, USA, 15 pages. https://doi.org/10.1145/3586183.3606779

1 INTRODUCTION

Over the past decade, we have witnessed an increase in the number of hearable devices like headsets, and earbuds, with millions of people using them worldwide [50]. Here, we introduce a new capability for hearable devices, which we call *"semantic hearing"*.

Consider a scenario where a user is wearing ear-worn devices on a beach and desires to listen to the calming sounds of the ocean while blocking out any human speech nearby. Similarly, while walking on a busy street, the user may wish to reduce all sounds except for emergency sirens; or while sleeping, they may want to listen to the alarm clock or baby sounds but not the noise from the street. In another scenario, the user may be on a plane and desire to hear human speech and announcements but not the sound of a crying baby. Or while hiking, the user may want to listen to the birds chirping

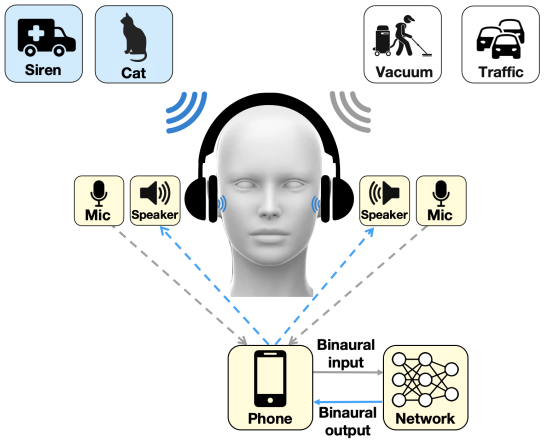
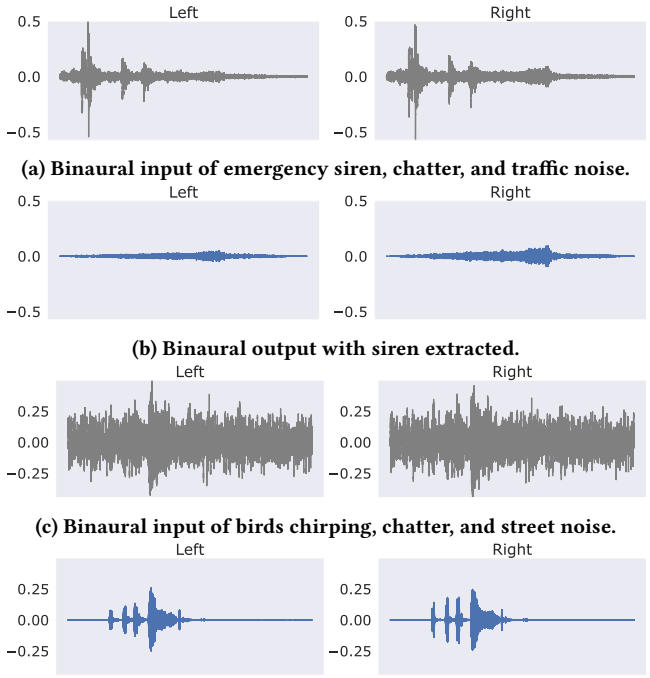


Figure 2: Semantic hearing architecture. The binaural input sounds are captured at a wired noise-canceling headset and sent to a phone, where we run on our sound extraction network. This extracts the binaural output that captures the target sounds (e.g., sirens and cat sounds) and suppresses noise and interfering sounds (e.g., vacuum and traffic noise). This binaural output is played back in real-time.



(a) Binaural input of emergency siren, chatter, and traffic noise.

(b) Binaural output with siren extracted.

(c) Binaural input of birds chirping, chatter, and street noise.

(d) Binaural output with birds chirping extracted.

Figure 3: Real-world binaural input and output recordings obtained with our semantic hearing system.

but not the chatter from other hikers (see examples in Fig. 1). These and other potential use cases require noise-canceling earphones for canceling all the sounds and then a mechanism for introducing *back* the desired sounds into the earphones. The latter, which is the focus of our work, requires programming the output acoustic scene in real-time by semantically associating the individual incoming sounds with user input to determine which sounds to allow in the hearable device and which sounds to block.

Animals have evolved over millions of years to focus on target sounds and the associated directions [32]. However, achieving this capability with in-ear devices like earphones and headsets is challenging for three key reasons.

- *Real-time requirements.* The sounds output by our design should be synced with the user's visual senses. This requires real-time processing that satisfies stringent latency requirements. Research on medical hearing aids and augmented audio shows that we need a latency of less than 20-50 ms [24, 59]. This requires identifying the target sounds using 10 ms or less of audio blocks, separating them from interfering sounds, and then playing them back, all on a computationally-constrained device like a smartphone.
- *Binaural processing.* Sounds arrive at the two ears with different delays and attenuations [64]. The physical separation between the two ears and the reflections/diffraction from the wearer's head, i.e., the head-related transfer function, provide cues for spatial perception. To preserve these cues, we need a binaural output to preserve or recover this spatial information for the target sounds across the two ears.
- *Real-world generalization.* While training and testing a neural network on synthetic data is common in audio machine learning research, designing a binaural target sound extraction network that generalizes to real-world hearable applications is challenging. This is because it is difficult to fully capture the complexity of real-world reverberations and head-related transfer functions (HRTFs) in simulations. We however require generalization to in-the-wild use in unseen acoustic environments across different users.

In this paper, we address the above challenges and demonstrate *semantic hearing*[1] with hearable devices. To achieve our goal, we make two key technical contributions. We design the first neural network capable of achieving binaural target sound extraction. Our network takes the two audio signals from the microphones at the two ears as binaural input and outputs two audio signals as binaural output, while preserving the directionality of the target sounds in the acoustic scene. To do this, we start with our recent single-channel (not binaural) transformer model for target sound extraction [66], which had neither real-world evaluation nor real-time smartphone operation. First, we optimize the network for real-time operations on smartphones. Then, we design a network that jointly processes the binaural input signals, allowing it to preserve the spatial information about the target sounds and output binaural audio (see §3.2). This joint processing is more effective at binaural target sound extraction and has half the computational cost of processing the binaural input signals separately.

We also design a training methodology that ensures our binaural network can generalize to real-world situations, such as reverberations, multipath, and HRTFs. Obtaining training data in fully natural environments can be difficult because we may capture mixtures but

[1]Our inspiration for the name 'semantic hearing' is directional hearing which is the ability to hear sounds from a specific direction [10, 16, 67]. Similarly, semantic hearing is the ability to hear the sounds that are specified by some semantic descriptions, such as sound classes.

lack access to the ground truth sounds needed for supervised learning. Moreover, training a network that can generalize to in-the-wild use with hearables requires that the training data capture reverberations, multipath, and head-related transfer functions across a large number of users. To achieve this, we synthesize our training data using multiple datasets. First, we use an HRTF dataset, which includes measurements from 41 users in non-reverberant environments. We convolve the room impulse responses with thousands of examples from 20 different audio classes to generate both our mixtures and the ground truth binaural audio. However, this does not capture the reverb and multipath in realistic environments. Therefore, we augment these synthesized mixtures with training data synthesized from three different datasets that provide binaural room impulse responses captured in real rooms. This facilitates our network to generalize to users and real-world environments that are not in the training dataset.

To demonstrate proof-of-concept, we augmented an off-the-shelf noise-canceling headset with commercial wired binaural earphones that provide access to data from both microphones. We implement our neural network on a connected smartphone and train it with 20 different sound classes, including sirens, baby cries, speech, vacuum cleaners, alarm clocks, and bird chirps. Our results are as follows.

- We achieve an average signal improvement of 7.17 dB across the 20 target sounds, in the presence of interfering sounds and urban background noise. Our real-time network has a 6.56 ms runtime on iPhone 11 for processing a 10 ms chunk of binaural audio.
- In-the-wild evaluation with participants in various indoor and outdoor scenarios with our hardware shows that our system can extract the target sounds (Fig. 3) and generalize to previously unseen participants, and environments, without requiring any training data collection with our hearable hardware.
- In a spatial hearing study where we played sounds from different directions in five previously unseen rooms, participants were able to predict the direction of the target sounds output by our system with 50th and 90th percentile errors of 22.5° and 45° respectively. These errors were similar for noise-free clean sounds.
- In a user study with 22 participants who spent over 330 minutes rating binaural data from real-world indoor and outdoor environments, our system achieved a higher mean opinion score and interference removal for the target sounds than the binaural input.

Contributions. We introduce the concept of semantic hearing, where we can program the binaural acoustic scene based on semantic sound descriptions. Our work makes five key contributions. 1) we present the first neural network to achieve binaural target sound separation and demonstrate that our network can run in real-time on smartphones, 2) we design a training methodology to generalize our system to unseen real-world environments, and users, 3) we implement a proof-of-concept with off-the-shelf hardware and show that our system achieves the above goals in real-world environments, 4) we highlight where our current system fails and opportunities for future research, and 5) by making our binaural models and datasets public, we hope to kickstart future research in the community towards further developing the concept of semantic hearing in practical hearable applications.

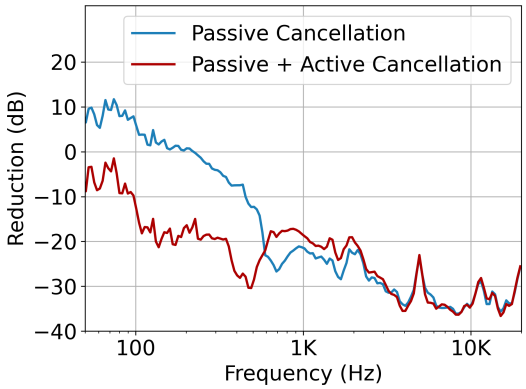


Figure 4: Noise reduction achieved with Sony WH-1000XM4 headphones – with and without active noise cancellation turned on – measured using an in-ear microphone inside the headphone cup. The reduction is measured relative to a microphone recording outside the ear cup. The spuriously large values at low frequencies (< 100 Hz) are due to the in-ear microphones picking up the wearer's blood pulse.

2 BACKGROUND AND RELATED WORK

Over the last decade, noise-canceling headsets and earbuds have undergone significant improvements, which now allow for more effective attenuation of *all* sounds in the environment. In fact, our experiments, where we play white noise to a human subject wearing a pair of Sony WH-1000XM4 headphones, show the impressive attenuation capabilities of these modern systems (Fig. 4). We identify this as an opportunity that provides us with an acoustic clean slate to introduce back target binaural sounds of interest from the environment. To the best of our knowledge, none of the prior work has explored semantic hearing capabilities for hearables. In the rest of this section, we describe related work in hearable systems, signal processing and machine learning for audio, and interaction tools.

Active noise cancellation and acoustic transparency. Active noise cancellation is a well-studied problem where outward-facing microphones are used to capture sounds [58]. An anti-noise signal is then transmitted to cancel *all* the external sounds and noise, which has more stringent delay requirements than semantic hearing. Traditional noise cancellation systems required bulky headsets. However in recent years lightweight in-ear earbud systems like the AirPods Pro can achieve reasonable noise-cancellation in many practical scenarios [1]. Semantic hearing leverages noise-cancelling earphones to cancel *all* sounds and then uses the mechanisms in this paper to program acoustic scenes in real-time.

The acoustic transparency mode for in-ear devices tries to imitate the sound response of an open-ear system by transmitting the appropriate signals into the ear canal [31]. Like active noise cancellation, this is agnostic to the sound classes. Adaptive transparency on Apple airpods is designed to automatically reduce the amplitude of loud sounds [3]. While related, this does not allow the user to pick and choose which sound classes to hear.

Speech systems. Prior systems have predominantly focused on improving the performance of speech-related tasks for in-ear devices (e.g., Airpods), telephony (e.g., Microsoft Teams), and voice

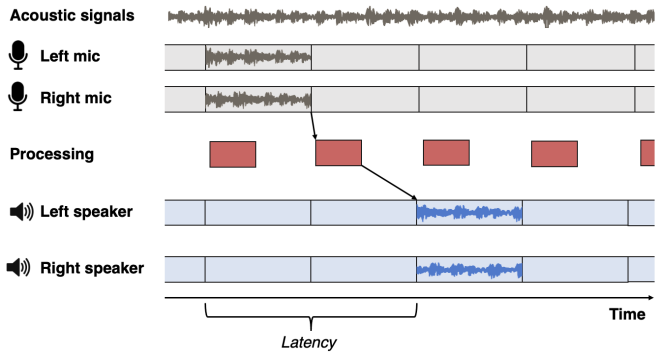


Figure 5: System requirements. Different components that contribute to latency in binaural target sound extraction.

assistants (e.g., Google Home). This includes speech enhancement [14, 41], target speech extraction [17, 22], and speech separation [37, 60]. Oftentimes, these systems collectively regard all non-speech sounds just as noise. In contrast, semantic hearing requires understanding the semantics of various natural and artificial sounds in real-time, in the presence of interfering sounds, and determining which sounds to allow and which to block, based on user input. Speech is one amongst many other sound classes in our system.

Neural networks for target sound extraction. Target sound extraction is the task of separating one or a limited number of target sounds from a mixture of sounds. Compared with speech systems, this is an underexplored problem in the audio machine learning community. However recent works have proposed neural networks that can achieve target sound extraction where clues about the target sound are provided either via audio [15, 21], images [19, 69], text [33, 35], onomatopoeic words [46], or a one-hot vectors [45]. All these models are designed for offline processing of audio clips, where the neural network has access to the entire audio file ($\geq$ 1 s) and hence cannot support our real-time hearable use-case.

The closest related work is our recent research on Waveformer [66], which introduces a neural network architecture for target sound extraction. Waveformer was shown to run in real-time on a desktop computer. Our work differs from [66] in two important dimensions. First, Waveformer is a single-channel model that operates on a single microphone. In contrast, our target use-case requires binaural processing across the two ears. As we show in §4.4, running the prior model independently on the two microphones is computationally expensive, failing to meet the real-time requirements on a smartphone. Second, all prior work in this domain was evaluated on synthetic datasets and has not been demonstrated on hardware in real-world scenarios. In contrast, we present the first binaural target sound extraction system that can run in real-time on smartphones. We designed a training methodology that allows our system to generalize to unseen indoor and outdoor real-world environments.

Hearable applications. Recent work has used in-ear sensors for health applications [11–13] and activity tracking [39, 52]. Prior work has also explored various interaction modalities like ultrasound sensing [68] and on-face interaction [70] for in-ear devices. The closest to our work is Clearbuds [14], which focuses on the task of enhancing the speech of the wearer using synchronized audio signals from two wireless earbuds. This prior work is focused on speech enhancement and is complementary to our system. Further, since the target application for [14] is telephony, it uses a 44.8 ms lookahead and has a latency of 109 ms.

Audio-based tools. Prior work has explored the use of sounds to perform activity recognition for wearables and smart home applications [28, 29, 34, 36, 43, 63, 71]. These systems operate on around 1s audio chunks as the target use cases do not have the O(10 ms) latency requirements of in-ear audio applications. Prior work has also designed interaction tools for audio editing [49, 55]. Our work is complementary in that it is focused on in-ear audio applications and semantic hearing that has more stringent latency requirements.

3 SEMANTIC HEARING

We first describe our system requirements and then present the network architecture we use for real-time binaural target sound extraction on smartphones. Next, we present our training methodology that generalizes our design to real-world use.

3.1 System Requirements

The goal of our design is to program the acoustic environment with imperceptible latency such that the target sound of interest is present but all other interfering sounds are suppressed. Given the stringent latency constraints, we cannot perform the necessary computation in the cloud but have to operate in real-time using computationally constrained devices like smartphones. Further, the target sounds generated by the model must originate from the same spatial directions as the real-world target sounds. Thus, our design must meet two key requirements: 1) real-time low-latency operation, and 2) binaural real-world generalization.

Real-time low-latency operation. Fig. 5 shows the different components that contribute to end-to-end latency in binaural acoustic processing systems. The first step is to feed the sound signals into two memory buffers of the binaural microphones. The acoustic data from the two microphones in each block is then fed into our neural network that outputs a block-length worth of binaural target sound data. This binaural output is then played back through the two speakers on the headset.

To ensure that the audio played through the headset is synced with the user's visual senses, we need this end-to-end latency to be less than 20-50 ms [24, 59, 67]. To achieve this, we need to reduce the buffer duration, the look-ahead duration and the processing time. This is challenging for multiple reasons. 1) A small buffer duration of say 10 ms means that the algorithm has only an 10 ms block of current data to not only understand the semantics of the acoustic scene but also separate the target sound from other interfering sounds. While we can use the acoustic signals that arrived prior to the current block, many of our target sounds (e.g., door knocks) are not continuous. Reducing the buffer size even further to say 2 ms can be challenging from an operating system perspective since it can increase the number of system calls. 2) While a large lookahead can provide more context for the neural network to extract the target sounds, meeting our end-to-end latency requirement reduces the leeway we have in terms of the available lookahead to a few milliseconds. 3) Real-time operation requires processing each acoustic block within the duration of the block itself. This means

that it should take less than 10 ms to process a 10 ms buffer [67]. This can be challenging since neural networks are not known for their lightweight computation. Further, since we cannot send the data to the cloud, the processing must be performed on-device on computationally-constrained devices like smartphones. In addition to all the above constraints, the operating systems also has I/O delays which for audio on iOS is on the order of 4 ms, depending on the buffer size [2].

Binaural real-world generalization. In real life, the target sounds experience reverberations and multipath propagation due to reflections from walls and other objects in the environment. Further, the human head and torso reflect and obstruct sounds. As a result the target sound arrives with different amplitudes and delays at the two ears. The differences in the received sounds across the two ears provide spatial awareness to humans. Thus, it is critical in our design to preserve these differences and play the target sounds with different amplitudes and delays through the two speakers of the headset. This is challenging since the target and interfering sounds can be at different positions and experience different reverberations and reflections from the head-related transfer function. Further, the multipath effects and reverberations are difficult to predict in real-world environments, let alone the fact that the head-related transfer functions can change across wearers.

3.2 Binaural target sound extraction network

We first explain the high-level framework for our binaural target sound extraction neural network. Then we explain the causal and streaming adaptation of this network. Finally, we provide a detailed description of the network architecture.

3.2.1 High-level framework. Consider $s \in R^{2\times T}$ to be the input binaural signal provided to the target sound extraction network. Since time-domain models also have been shown to be able to learn representations analogous to STFT features [38], our network operates on time-domain binaural signals. As shown in Fig. 6a, the signal is first mapped to a representation in a latent space, $x \in R^{D\times(T/L)}$, by using a 1D convolution layer with a kernel size $\geq L$ and a stride equal to L. D and L are tuneable hyperparameters of the model. D is the dimensionality of the model, having a significant effect on the parameter count, and consequently the computational and memory complexities. L determines the duration of the smallest audio chunk that can be processed with the model. The latent space representation x, is then passed to a mask generator, $\mathcal{M}$, which estimates an element-wise mask m as:

$$m = \mathcal{M}(x, q) \mid m \in R^{D\times(T/L)};\ q \in \{0,1\}^{N_c}, \tag{1}$$

where N_c is the total number of sound classes the model is trained for. The representation corresponding to the target sound is obtained by element-wise multiplication of the input representation, x, and the mask, m, as follows:

$$y = x \odot m \mid y \in R^{D\times(T/L)}. \tag{2}$$

The output audio signal $\hat{s} \in R^{2\times T}$ is then obtained by applying a 1D transposed convolution on y, with a stride of L.

In contrast to more complex binaural extraction frameworks proposed specifically for *speech*, where each channel is separately and parallely processed [23, 25], our design jointly processes the two channels for computational efficiency. In our experiments, we show that our simpler framework performs competitively with the prior parallel processing frameworks in terms of target sound extraction accuracy, even with a 50% lower runtime cost.

3.2.2 Streaming inference and causality. For real-time on-device operation, the model must output the audio corresponding to the target sound as soon as the input audio is received, i.e., within the latency requirements detailed in §3.1. Since the audio is fed to the model from the device buffers, the buffer size determines the duration of the audio chunk the model receives at each time step. Assuming the buffer size to be divisible by the stride size L, the audio chunk size can be represented as the number of strides, K. That is, the buffer size of an audio chunk of size K is equal to KL samples. Such a real-time setup means that the model only has access to the current and previous chunks, but not future chunks. *This requires the model to be causal with the time resolution of the buffer size, i.e., KL audio samples.* As a result, in the high-level framework described above, the input convolution, the mask estimation block, the element-wise multiplication, and the output transposed convolution must operate on one audio chunk at each time step.

The binaural target sound extraction framework described in §3.2.1 can be adapted to chunk-wise streaming inference as follows. Consider the input audio signal corresponding to the kth chunk to be $s_k \in R^{2\times KL}$. The input 1D convolution maps this audio chunk to its latent space representation, $x_k \in R^{D\times K}$. The mask estimation block is then used to estimate the mask corresponding to the target sound, based on the current chunk, as well as a finite number of the previous chunks:

$$m_k = \mathcal{M}(x_k, q, x_{k-1}, x_{k-2}, ...) \mid m_k \in R^{D\times K}. \tag{3}$$

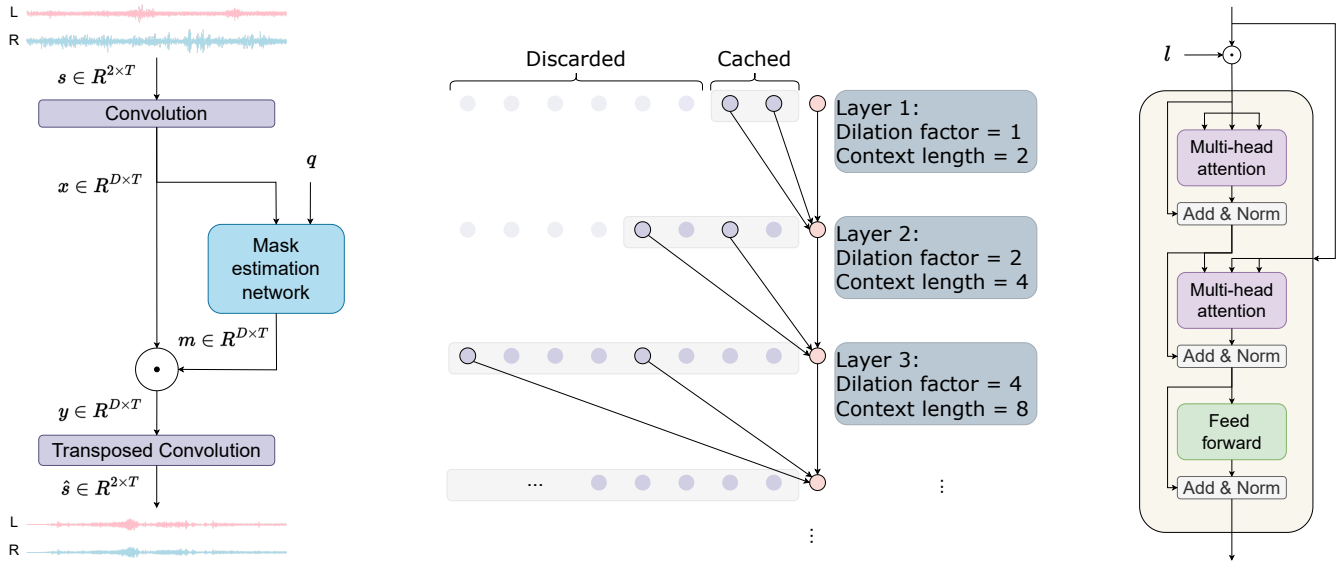
The previous chunks act as the audio context for the neural network, referred to as the receptive field of the model. A receptive field of 1-1.5s is shown to result in good performance [38]. The output representation of the current chunk corresponding to the target sound, $y_k \in R^{D\times K}$ can then be obtained as:

$$y_k = x_k \odot m_k. \tag{4}$$

The resulting output representation is then converted to the output signal $\hat{s}_k \in R^{2xKL}$ by applying the 1D transposed convolution.

3.2.3 Mask estimation network. Several architectures have been proposed in the literature for mask estimation such as Conv-TasNet [38], U-Net [30], SepFormer [60], ReSepFormer [61], and Waveformer [66]. Waveformer is an recently proposed efficient streaming architecture implementing chunk-based processing, which makes it suitable for our task. In this work, we propose a modified version of Waveformer to further increase efficiency without any loss in performance. The mask estimation network is an encoder-decoder neural network architecture, where the encoder is purely convolution-based and the decoder is a transformer decoder.

Different from Waveformer, in this work, we use the same dimensionality for both the encoder and decoder. This allowed us to use the standard transformer decoder [65], instead of a modified one used in the Waveformer. Waveformer proposes a smaller dimensionality for the decoder block, compared to the rest of the model. The transition between different dimensionality is achieved using projection layers (1D convolution layers with kernel size equal to 1).



(a) Binaural extraction framework **(b) Encoder** **(c) Decoder**

Figure 6: Binaural target sound extraction network architecture. a) Our high-level binaural extraction framework. Mask estimation network is an encoder-decoder architecture operating on latent space representation of binaural signals to extract mask for target sound based on the query vector q. b) and c) show the encoder and decoder architectures used in the mask estimation network. The encoder processes the previous input context and does not consider the label embedding. Decoder first conditions the encoded representation with the label embedding, l, and then generates the mask corresponding to the target sound using the conditioned representation.

This however breaks the residual paths and the result might affect the gradient flowing back, which is mitigated in the Waveformer using a long residual connection bypassing the decoder. For our binaural application, however, we found that different dimensionality is not necessarily providing gains warranting the complexity of the projection layers and the long residual connection.

Encoder. Mask estimation in Eq. 3, involves processing many previous chunks in addition to the current chunk to obtain the mask corresponding to the current chunk. Repeated processing of the entire receptive field for each iteration could become intractable for a real-time on-device application. To mitigate this inefficiency, while achieving a large receptive field, our mask estimation network implements a Wavenet [47] style dilated causal convolutions for processing the input and previous chunks. In this work, for efficient on-device inference, we implemented the dynamic programming algorithm proposed in Fast Wavenet [48]. As shown in Fig. 6b, higher efficiency is achieved by reusing the intermediate results computed in the previous iterations. The encoder function $\mathcal{E}$ processes the input chunk x_k and an encoder context ξ_k to generate the encoded representation of the input chunk:

$$e_k, \xi_{k+1} = \mathcal{E}(x_k, \xi_k) \mid e_k \in R^{D\times K} \tag{5}$$

The size of the context ξ_k depends on the hyperparameters of the encoder. In our implementation, the encoder is comprised of a stack of 10 dilated causal convolution layers. The kernel size of all layers is equal to 3, and the dilation factor is progressively doubled after each layer starting with 1, resulting in dilation factors $\{2^0, 2^1, ..., 2^9\}$. Since the kernel size is equal to 3, the context needed for each dilated convolution layer is twice the layer's dilation factor. As long as this context is saved after each iteration, and padded with the input chunk in the next iteration, the intermediate results corresponding to the previous chunks do not have to be recomputed. Thus the size of the context ξ_k is equal to $2 \times \sum_{i=0}^{9} 2^i = 2046$.

Decoder. The query vector q is first embedded into the embedding space using a linear layer to generate a label embedding $l \in R^D$. The mask corresponding to the target sound m_k is estimated using a transformer decoder layer [65], represented here as $\mathcal{D}$. The encoded representation is first conditioned with the label embedding l by an element-wise multiplication. The encoded representation and the conditioned encoded representation are first concatenated in the time dimension, with those from the previous time step, before processing with the transformer decoder layer $\mathcal{D}$. The encoded representation from the previous time step, e_{k-1}, acts as the decoder context. The mask estimation can be written as:

$$m_k = \mathcal{D}(\{l \cdot e_{k-1}, l \cdot e_k\}, \{e_{k-1}, e_k\},) \tag{6}$$

where $\{\}$ represents concatenation in the time dimension. As shown in Fig. 6c, the transformer decoder $\mathcal{D}$ first computes the self-attention result of the conditioned encoded representation $\{l \cdot e_{k-1}, l \cdot e_k\}$ using the first multi-head attention block, followed by cross-attention between the self-attention result and the unconditioned encoded representation $\{e_{k-1}, e_k\}$ using the second multi-head attention block. A feed-forward block along with residual connection generates the final mask corresponding to the target sound.

Table 1: Number of raw audio files collected for training/testing/validation from each dataset for our potential target classes. The total number of mixtures we generate using these for training, testing and validation are 100k, 10k, and 1k, respectively.

Dataset	Alarm clock	Baby cry	Birds chirping	Car horn	Cat	Rooster crow	Typing	Cricket	Dog	Door knock
FSD50K	34/4/4	30/8/4	65/52/8	36/21/4	73/39/9	23/9/3	78/68/9	61/14/7	109/33/13	65/33/8
ESC-50	24/8/8	24/8/8	24/8/8	24/8/8	24/8/8	24/8/8	24/8/8	24/8/8	24/8/8	24/8/8
MUSDB18	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0
DISCO	0/0/0	95/53/11	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0
Total	58/12/12	149/69/23	89/60/16	60/29/12	97/47/17	47/17/11	102/76/17	85/22/15	133/41/21	89/41/16

Dataset	Glass breaking	Gunshot	Hammer	Music	Ocean	Singing	Siren	Speech	Thunderstorm	Toilet flush
FSD50k	212/31/24	169/67/20	88/39/10	0/0/0	86/19/10	134/76/19	16/5/3	494/109/56	122/9/14	112/21/13
ESC-50	24/8/8	0/0/0	0/0/0	0/0/0	24/8/8	0/0/0	24/8/8	0/0/0	24/8/8	24/8/8
MUSDB18	0/0/0	0/0/0	0/0/0	581/358/62	0/0/0	480/291/54	0/0/0	0/0/0	0/0/0	0/0/0
DISCO	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0	0/0/0
Total	236/39/32	169/67/20	88/39/10	581/358/62	110/27/18	614/367/73	40/13/11	494/109/56	146/17/22	136/29/21

3.3 Training for real-world generalization

We first describe our audio class dataset curation and then present our training methodology to generalize to real-world scenarios.

3.3.1 Picking audio classes. Our main goal is to create a system that efficiently handles target sounds encountered in real-world situations. By focusing on practical applications, we identify a manageable set of target sound classes to extract. However, in reality, we come across a wide range of background sounds, many of which are not part of our target sound classes. To curate our dataset of sound classes, we follow the AudioSet ontology [20], which provides a comprehensive and structured representation of the relationships between various sound classes. The ontology arranges the sound classes as nodes in a graph and groups them into seven main sound categories. Each sound class node has a unique AudioSet ID and may contain one or more child nodes that represent more specific sound classes. For example, the "Hands" sound class has two children, namely "Finger Snapping" and "Clapping." In the rest of this section, we describe how we pick our target sound classes as well as the interfering classes.

• **Target sound classes.** We first consider various indoor and outdoor scenarios where the system is likely to operate, such as beaches, parks, streets, living rooms, offices, and cafes. Based on these scenarios, we identify potential sound sources that are prevalent in such locations, such as human speech, dogs, cats, birds, sea waves, and music. We then compile a list of sound classes associated with these selected target sounds and map each of these classes to a label in the AudioSet ontology. We eventually selected 20 sound classes that we felt human listeners could distinguish with reasonably high accuracy.

• **Other sound classes.** In the real world, the interfering sounds and noise often do not belong to our 20 target sound classes. To create a neural network that can generalize to interference from these sounds, we need a diverse set of interfering sound classes in our dataset.[2] However, this poses several challenges. Firstly, these sounds can come from a very large variety of sources, making it infeasible to exhaustively enumerate all of them. Secondly, since we want to use them as interfering signals, we must ensure that these sound classes do not overlap with our set of target classes. To overcome these constraints, we use the AudioSet hierarchical structure and our set of 20 target classes to generate a large set of 141 other sound classes. Specifically, we can define this set as the nodes that are neither a more specific nor a more general instance of any target (or known) class, according to the AudioSet hierarchy. In other words, by considering the AudioSet ontology as a directed acyclic graph with edges from each sound class node towards its child nodes, we define unknown sound classes as the set of AudioSet nodes that are disconnected from all target sound class nodes.

3.3.2 Audio dataset curation. Given the sets of the target and other sound classes, we next obtain labeled audio recordings for each of the sound classes. The challenge is that, we cannot rely on only a single general-purpose audio-tagging datasets as was done in prior single-channel work [66], This is because such datasets do not contain audio samples of all 20 target sound classes, and may contain a limited number of audio samples from the other sound classes. So, we combined audio samples from four different datasets: FSD50K [18] (general-purpose), ESC-50 [51] (environmental sounds), MUSDB18 [53] (music and vocals) and noise files for the DISCO [44] dataset (noise sounds). Since each dataset uses different class names, we standardized the class labels into the AudioSet labels by mapping every class in each dataset to the semantically closest label in the AudioSet ontology, if any. For FSD50K and MUSDB18, we performed additional dataset-specific pre-processing procedures. Specifically, since our goal is to create binaural mixtures of individual sources from multiple directions, we excluded audio samples from FSD50K that were already mixtures of multiple distinct sound sources. For MUSDB18, we extract and split audio into vocal and instrumental streams and assign them the AudioSet labels "Singing" and "Melody," respectively.

We divide the resulting audio samples into 15 second segments and discard all silent ones. We split each dataset into mutually exclusive training, testing, and validation sets and then combined them into our final dataset. For both the FSD50K and MUSDB18, we sample the training and validation audio files from the development split (90-10 split), and the testing samples from the evaluation split. For the ESC-50 dataset, we use the first three folds for training, the fourth fold for validation and the fifth for testing. For the DISCO noise dataset, the audio samples for each sound class is split into train, test and validation sets (60-33-7) before combining with the rest of the datasets. The final combined dataset consists of 20 target classes, distributed as shown in Table 1 and 141 other sound classes.

[2]Note that the target sound classes can also be interfering with each other.

3.3.3 Binaural data synthesis. The procedure above describes how we sample single channel sound classes from various audio datasets. However, our goal is to create binaural mixtures that (1) are representative of spatial sounds perceived by a diverse set of listeners, and (2) capture the idiosyncrasies of real world reverberant environments. For this purpose, we use a pre-existing dataset of 43 human head-related transfer function (HRTF) measurements (CIPIC [8]) to address the first challenge. We also augment this with three datasets of measured (SBSBRIR [57], RRBRIR [26]) and simulated (CATT RIR [27]) reverberant binaural room impulse responses (BRIRs) to address the second requirement. We split each dataset across rooms and listeners into train, test and validation (70-20-10) sets. We ensure that no BRIR subjects or rooms are sampled across different sets. For each sample during training, we randomly choose one of the datasets and sample a single room and participant from its training set. Then, to create a binaural mixture with K sources, we independently pick a source direction for each of the K sources, out of all source directions available for this room and this participant subject in the dataset. Note that since the source directions are independently picked, two different sound sources might end up being at the same direction from the wearer. We then obtain a set of $2K$ room impulse responses $h_{1,L}, h_{1,R}, h_{2,L}, \ldots, h_{K,R} \in \mathbf{R}^N$, where N is the length of the room impulse response. Hence, for a training sample with input audio signals of length T samples long, denoted by $x_1, x_2, \ldots, x_n \in \mathbf{R}^T$, we can compute the sound received at the left and right ears, s_L and s_R as, $s_L = \sum_{k=1}^{K} x_k * h_{k,L}$ and $s_R = \sum_{k=1}^{K} x_k * h_{k,R}$. Here '$*$' represents the convolution operation. The synthesized binaural audio mixtures are sampled at 44.1 kHz. If room impulse response in our HRTF dataset has a different sampling rate, we resample the signal before and after convolving.

3.3.4 Training procedure. We used the Scaper toolkit [56] to synthesize binaural mixtures dynamically on the fly during training. For training and validation, our binaural mixtures consist of two randomly picked target classes, each with an 5-15 dB SNR relative to the background sounds, and 1-2 other classes that each have a 0-5 dB SNR relative to the background sounds. We also use background sounds sourced from the TAU Urban Acoustic Scenes 2019 dataset [42] in our mixtures. Each mixture is 6 seconds long. Sounds from the target and other background classes are between 3 to 5 seconds long, while background urban sounds persist for the entire duration of the mixture. In addition to the mixture, we also synthesize ground truth signals y_L and y_R, respectively at the left and right channels, for each chosen target sound source t as, $y_L = x_t * h_{t,L}$ and $y_R = x_t * h_{t,R}$.

The network is then trained to produce a pair of left and right channel target sound estimates $\hat{y}_L$ and $\hat{y}_R$. To preserve the spatial cues, such as interaural time differences (ITD) and interaural level differences (ILD), we use the sample-sensitive and scale-sensitive signal-to-noise ratio (SNR) loss function, applied independently and then average the left and right SNRs to obtain the loss function:

$$SNR(\hat{x}, x) = 10 \log \left(\frac{||x||^2}{||x - \hat{x}||^2} \right)$$

$$L = -\left(\frac{1}{2} SNR(\hat{y_L}, y_L) + \frac{1}{2} SNR(\hat{y_R}, y_R) \right).$$

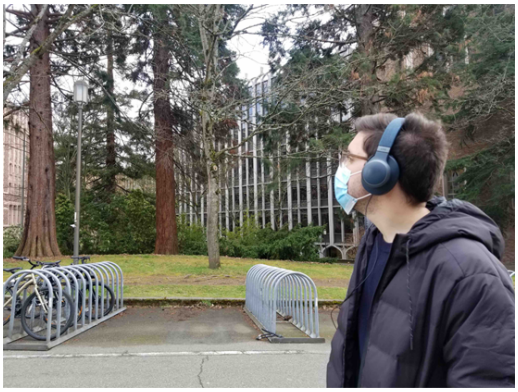


Figure 7: A participant in our in-the-wild evaluation where the target sound was birds chirping in the presence of urban environment noises. The participants could move their head freely and the target sound source could also be mobile.

Finally, we train our transformer model for 80 epochs, with an initial learning rate of 5e-4. After completing 40 epochs, we halve the learning rate if there is no improvement in the validation SNR for more than five epochs. We emphasize here that the *training data do not include any measurements with our binaural hardware* and the results we report in this paper evaluate generalization to our hardware, unseen users and environments.

4 RESULTS

We first describe our setup for real-world evaluations and then present our binaural network benchmarks.

Hardware prototype. Our hardware setup includes a pair of SonicPresence SP15C binaural microphones that are wired to capture high-quality recordings. We use an iPhone 12 to process the recorded data and output the audio through noise-canceling headphones like JBL Live 650BTNC and the NUBWO gaming headsets. We use a lightning-to-aux adapter to connect the headphones to the iPhone over a wire. We also use a USB hub to connect both the microphones and the headphones to the smartphone.

Participants. We recruiting 9 individuals (3 female, 6 male) across our in-the-wild and spatial cues evaluations. We also invited 22 participants (6 female, 16 male) for our online hearing study.

4.1 In-the-wild evaluation

To evaluate the proposed system in real-life scenarios, we conduct in-the-wild experiments to assess the effectiveness of our system.

In-the-wild scenarios. 5 individuals (3 female and 2 male) wore our hardware and collect sounds in the real world. These experiments were conducted in typical application settings: offices, living rooms, streets, rooftops, parks, and restrooms. Since some of the sound classes were relatively less common, our in-the-wild experiments had a subset of classes which most commonly appeared in our recordings: alarm clock, car horn, door knock, speech, computer typing, hammer, birds chirping, and music. The position and movement of the sound sources were uncontrolled and reflective of real-world scenarios, where the sound sources could be mobile. Furthermore, in all experiments, participants had complete freedom

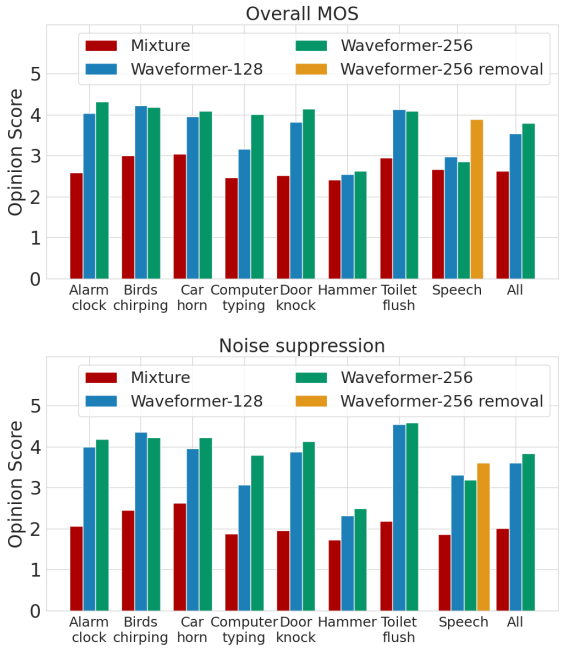
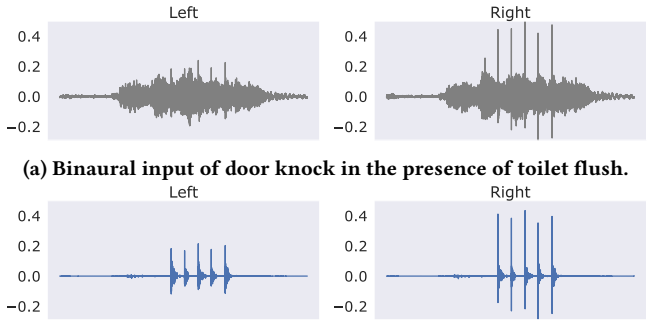


Figure 8: In-the-wild evaluation results for (a) mean opinion score (MOS) and (b) noise suppression across various classes that occurred in real-world data collection.

to move their heads, causing the sound source positions relative to the microphones to vary over time (Fig. 7). Thus, our in-the-wild evaluation captured both mobile wearers as well as mobile sound sources that naturally occured in real-world scenarios (e.g., cars moving or birds that fly).

Evaluation procedure. Unlike with our simulated training data, we do not have clean, sample-aligned ground truth signals to objectively compare the binaural outputs of our system with. Hence, we conduct a listening study to compute a mean opinion score (MOS) regarding the sound extraction accuracy. This metric is crucial to evaluate the perceptual quality of our algorithm for end-users, although it has often been omitted in prior non-speech sound extraction research. We invited 22 participants (6 female, 16 male, mean age 34.6) to the online listening study. The study consists of 16 sections. In each section, the participants evaluated the quality of 3 or 4 5.0-8.5 second audio samples. The audio samples played at each section were in-the-wild recordings processed in the following three ways for the same target label: (1) the original recording, (2) the output of our 128-dimensional binaural network, (3) the output of our 256-dimensional binaural network. For the subset of the evaluations that involved speech as the target sound, we also included an additional fourth audio sample that was obtained by extracting of the interfering class (e.g., door knocks) and then subtracting it from the input recording to estimate the target speech.

We conducted a pre-screening process to ensure that the participants used suitable binaural headsets. This involved playing two white noise samples, one exclusively from the left channel and one exclusively from the right channel. The participants were instructed to confirm that they heard the sounds only from the



(a) Binaural input of door knock in the presence of toilet flush.

(b) Binaural output with door knock extracted.

Figure 9: Qualitative result with a real-world recording.

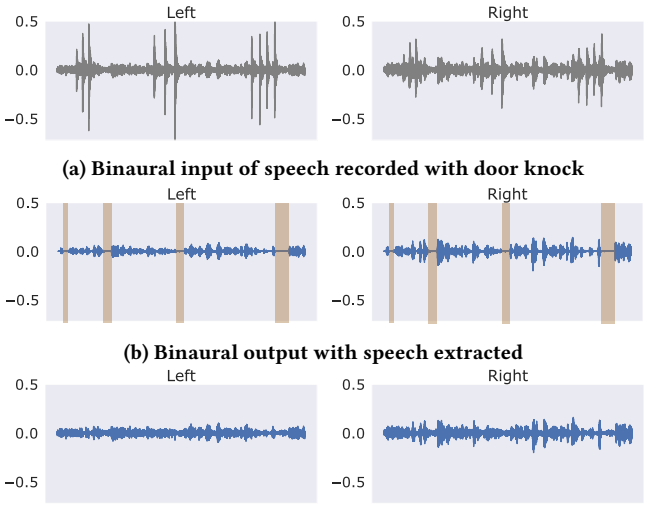
correct channels. 11 of our participants used headphones, and 11 used earbuds during our online user study.

We measured the sound extraction quality based on both interference suppression and overall mean opinion score (MOS), as they are often included in speech enhancement quality assessment:

(1) **Noise suppression**: *How INTRUSIVE/NOTICEABLE were the BACKGROUND sounds? 1 - Very intrusive, 2 - Somewhat intrusive, 3 - Noticeable, but not intrusive, 4 - Slightly noticeable, 5 - Not noticeable*
(2) **Overall MOS**: *If the goal is to focus on the <TARGET> sounds, how was your OVERALL experience? 1 - Bad, 2 - Poor, 3 - Fair, 4 - Good, 5 - Excellent*

Results. In Fig. 8, we present the results of the user evaluations for the interference sound suppression and overall quality improvement of our system for different target sound labels. The results demonstrate the system's capability to significantly reduce unwanted background sounds, as indicated by an increase in the overall noise suppression score from 2.01 (corresponding to *2 - Somewhat intrusive*) to 3.61 (between *3 - Noticeable, but not intrusive* and *4 - Slightly noticeable*) with the 128-dimensional model, and to 3.84 (slightly worse than *4 - Slightly noticeable*) with the 256-dimensional model. We also observed a similar trend in the overall MOS improvement, with an improvement from 2.63 for the input signal to 3.54 and 3.80 after processing with the 128-dimensional and 256-dimensional models, respectively. Figs. 3d and 9 also show that our network preserves the timing of the target sounds and can silence out noise outside the target sound duration.

The results also offer interesting insights at a per-class level. In general, the 128-channel model performs only slightly worse than the 256-channel model for almost all classes, except for the "Computer typing" class, where the gap in the overall MOS between the two models is almost 0.84 MOS points. This is likely due to a particularly noisy recording taken near a running generator, where the 128-channel model created faint, unpleasant artifacts that were not observed with the 256-channel model. However, both models performed poorly in the "Hammer" class, where the target hammer sound was recorded in the presence of interfering music. Although the network correctly silenced the time segments that did not contain the hammer sounds, there was a noticeable residue from the music when there was a hammer sound, which the listeners found intrusive. Another important finding from the study is the significant improvement obtained by removing interfering signals from



(a) Binaural input of speech recorded with door knock

(b) Binaural output with speech extracted

(c) Binaural output with door knock removed

Figure 10: Extracting speech as a target here causes momentary periods of excessive signal attenuation (highlighted in b) as the network tries to remove door knocks and background sounds. However, if we extract and then subtract door knock sounds, the background noise is still faintly present, and the resulting signal sounds less harsh.

the input recording when the target is speech. By removing short-length sounds such as door knocks from the recorded signal instead of extracting the speech directly (see Fig. 10), we were able to increase the overall MOS by 0.91 points. Finally, it's worth noting that these in-the-wild results were obtained from the models trained solely on synthesized data, without any training on data collected from our hardware or for the participants.

4.2 Evaluating user-perceived spatial cues

We present experiments conducted in five ordinary, reverberant rooms to evaluate the ability of our design to preserve or recover user-perceived spatial cues. As with the in-the-wild evaluation, our training data had no samples either from our hardware or the tested real-world environments.

Data collection. We collected real-world audio recordings of our target sounds from known directions. To achieve this, five participants (3 male, 2 female) were fitted with binaural microphones and seated on a rotating chair positioned at the center of a large, printed semicircular protractor measuring 70×36 inches, as shown in Fig. 11. The protractor was lined at regular 22.5° intervals (nine lines total) for precise rotational measurement. A Sony SRS-XB10 loudspeaker was placed on a fixed tripod at the 90° line of the protractor to emit different sound signals. To control the angle-of-arrival of the sound signal relative to the listener, the participants were asked to rotate the chair and align themselves with one of the protractor's lines.

The data was collected in 9 stages. In each stage, the user is rotated towards a different angle. The first stage starts with the participant facing the 180° line. After completion of each stage, the participant rotates 22.5° clockwise to the next marked angle. At

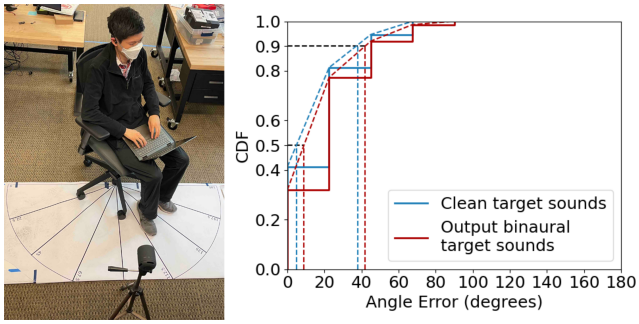
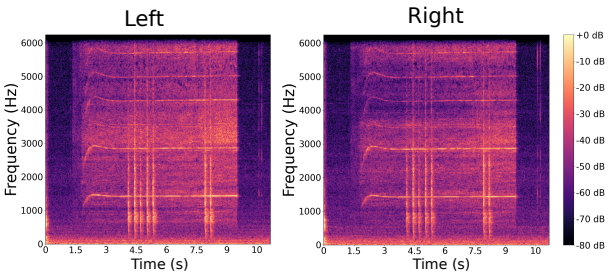


Figure 11: Spatial cue evaluation. (left) the evaluation setup, and (right) the CDF of the error between the ground truth source direction and the user-perceived source direction after listening to the isolated clean target sounds as well as network output binaural target sounds. The dashed lines are interpolated CDFs used to compute the interpolated median and 90th percentile error.

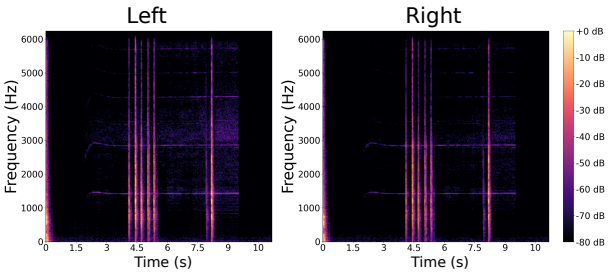
each stage, the loudspeaker plays four 5-second audio samples: (1) white noise, (2-3) two test samples belonging to the target sound classes, and (4) a test sample belonging to the interfering other sound classes. Across all stages of data collection, the chosen audio samples comprise exactly 9 test samples from 9 distinct interfering other sound classes and 6 test samples from 6 distinct target sound classes. Notably, each test sample from the target classes is recorded for 3 different relative angles.

Evaluation procedure. Since our goal is to develop a system that accurately preserves the spatial cues perceived by human listeners, we design a user study to compute the perceived angle-of-arrival for the target binaural sounds output by our system. To this end, based on the collected audio recordings, we first create sound mixtures by sampling two audio clips from the target classes, and 1-2 clips from the interfering other classes. The mixtures are generated using Scaper. We choose the reference loudness of the background to be -50 LUFS, and we set the SNR of the target class sounds to 15-25 dB and that of the interfering other class sounds to 0-10 dB. We process each mixture by choosing a target class and running the mixture through our network.

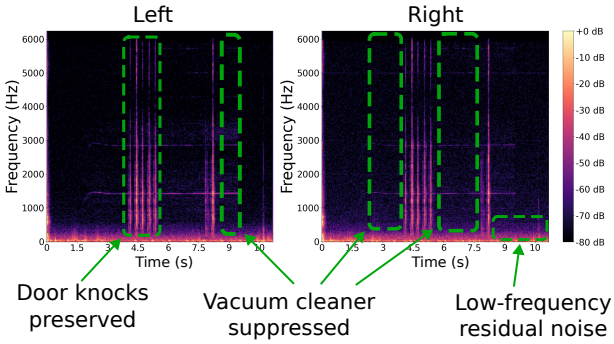
We play the recordings of the individual clean target sounds with no interference, as well as the network output samples estimating these target sounds from the created mixtures, to the same set of participants via a pair of binaural earphones. Since the perceived spatial cues rely heavily on anthropometric features, all the sound signals played to a given participant originated from the binaural data obtained from that same participant in the data collection step. Prior to listening to each sample, participants are informed of the target class they should be localizing. After listening, they are asked to predict the direction of the sound source. To prevent the participants from associating each output sample with its corresponding individually-recorded target sound, the samples are played in a random order. To help the participants establish an orientation reference, we play back the white noise samples for each angle in the increasing order at the start of the evaluation. Additionally, in cases of uncertainty between two specific source angles, the participants are allowed to re-listen to the white noise samples recorded for these angles. The study lasted around 20 minutes per user.



(a) Sound recorded outside the headphone cups.



(b) Semantic hearing output played by headphones.



(c) Sound recorded inside the headphone cups.

Figure 12: Spectrograms of binaural recordings showing results from our end-to-end experiment with a wearable headset. Here, we extract door knock sounds in an environment with a nearby active vacuum cleaner.

Results. We compare the errors between the ground truth source directions and the users' perceived arrival directions obtained for both the clean interference-free target sound recordings as well as the binaural target sound signals generated by our system for the mixture signal input. Our findings, as illustrated in Fig. 11, show that the mean angle error slightly increases from 18° to 23.25°. Additionally, we observe that the interpolated 50th and 90th percentile errors also increase marginally from 5° to 9° and from 38° to 42°, respectively. This demonstrates that our model preserves the spatial cues of the target sounds in its output and has a negligible impact on how users perceive the source directions.

4.3 Integration with noise canceling headsets

So far, we have treated semantic hearing and active noise cancellation as two separate systems that function independently. In practice, however, the end-to-end system requires a few additional considerations. Firstly, many active noise cancellation systems rely on a recorded signal inside the ear cup to adaptively silence the noise signals and achieve adaptive noise cancelation. Hence, the audio we play back to perform semantic hearing may influence the noise cancellation algorithm. Secondly, active noise cancellation systems are not perfect, and they may still let some sounds through. To address these concerns, we record data while a user is utilizing our end-to-end system in real time. The user wears a pair of Sony WH-1000XM4 headphones with active noise cancellation enabled. In addition to the outer microphones used to capture external sounds to process, they also wear binaural microphones inside the earcups to record the sound produced by the active noise cancellation and semantic hearing systems together, i.e. as heard by the user. The user chooses to listen to the sound of door knocks as a vacuum cleaner is turned on nearby. For this experiment alone, we run our semantic hearing algorithm on the audio recorded from the outer microphones on a laptop with an Intel Core i5 CPU. The processed audio is played back through the headphones.

Fig. 12(a)-(c) shows the spectrograms for three binaural signals: the signal recorded at the outer microphone, the signal played through the headphones, and the signal recorded inside the earcups. We demonstrate that while the recordings from the inside earcups are slightly noisier, we clearly see that the system can suppress the unwanted sounds (vacuum cleaner), while preserving the target sounds (door knocks). This demonstrates the feasibility that such a system can coexist with active noise cancellation systems. We note that to mitigate residual noises, the semantic hearing subsystem may have to integrate the residual audio from noise cancelling headphones to adapt the playback signal to the residual noise as well. However, this comes with stricter latency requirements and thus we leave it for future work.

4.4 Benchmarking the neural network

In-the-wild evaluation with human evaluators is closest to real-world use. It is however hard to objectively compare different models due to the lack of ground-truth signals, as well as due to the challenges in obtaining a large amount of test data necessary for the statistical significance of smaller performance gaps. To address these practical limitations, we also evaluate our model on an extensive reverberant binaural testset comprising 10000 mixture and ground-truth pairs. We synthesized the benchmarking dataset to mimic real-world situations following the approach in §3.3.

To evaluate the performance of our binaural extraction model, as shown in Table 2, we compare the following three binaural target sound extraction frameworks.

- **Dual-ch.** This is the dual-channel architecture we proposed in §3.2.1 for efficient binaural target sound extraction. In this framework, the binaural signal is converted into a combined latent space representation before the mask estimation. Since both left and right channels are combined into a common representation, a single instance of the mask estimation network is used for estimating the mask corresponding to the target sound. We consider our mask estimation architecture with both $D = 128$ and $D = 256$.
- **Parallel.** This is the binaural framework proposed in [25] that implements parallel processing of the left and right channels, along with some cross-communication between channels. The binaural

Table 2: Performance and efficiency comparison of different binaural target sound extraction frameworks and mask estimation architectures on a large test dataset across 20 target classes generated using the approach described in §3.3.

Binaural framework	Mask estimator	Params (M)	SI-SNRi (dB)	ΔITD (μs)	ΔILD (dB)	Runtime (ms)
Dual-ch	Ours ($D = 128$)	0.52	7.17	87.77	0.88	6.56
	Ours ($D = 256$)	1.74	7.41	85.16	0.87	12.54
Parallel	Ours ($D = 128$)	0.86	7.24	81.72	1.08	13.35
	Conv-TasNet	2.33	4.43	670.05	-	15.58
Single-ch	Ours ($D = 256$)	1.68	7.43	79.70	1.32	22.19
	Vanilla Waveformer ($D = 256$) [66]	1.69	7.37	85.33	1.27	25.85

framework in [25] is originally proposed for binaural speech separation. We implemented this framework for both our mask estimation network with $D = 128$ and Conv-TasNet [38]. We include Conv-TasNet as it is one of the most widely used signal enhancement model architectures. We choose a configuration of Conv-TasNet that resulted in similar runtime to that of our model and trained both models with our training dataset.

• **Single-ch.** In addition to the above two binaural extraction frameworks, we also evaluate and compare the performance with a single-channel extraction baseline. Since the target sound extraction models we consider are sample-aligned, models trained with monaural inputs and outputs can be independently applied to the left and right channels. Similar to the *Parallel* case, this also involves two instances of the mask estimation network. However, by contrast, the model parameters applied to the left and right channels are the same and there is no cross-communication between the channels. We implement the best configuration of our model ($D = 256$) so that this serves as a strong baseline.

For each model, we compare the performance in terms of the signal quality, the accuracy in spatial cues, and the on-device runtime requirement. We measure the signal quality using the scale-invariant signal-to-noise-ratio [54] improvement (SI-SNRi) of the output compared to that of the mixture, computed with respect to the ground-truth. The SI-SNRi results are averaged over the entire testset, across the left and right channels. Following [25], the spatial cue accuracy is measured using the difference in the interaural time differences (ITDs) and interaural level differences (ILDs) between the output binaural signal and the ground-truth binaural signal, denoted as ΔITD and ΔILD. We compute ITD using cross-correlation, limiting them to ±1 ms, as was done in [40]. The model runtimes are measured on iPhone 11, by converting them to ONNX format [9] and then executing them using ONNX Runtime for iOS. The runtimes are measured for computing a 10 ms output chunk averaged over 100 runs. Therefore, the runtime must be less than 10 ms for deployment, which our dual-channel model with $D = 128$ meets.

In our experiments, we observed that the causal Conv-TasNet converges to the local minima of generating a constant zero signal when trained only with the SNR loss. This phenomenon is also observed in [66], which suggested training Conv-TasNet with 90% SNR + 10% SI-SNR loss. The likely cause for this is, unlike the speech datasets that Conv-TasNet is originally designed for, sound datasets have a significant amount of silence, causing the Conv-TasNet optimization process to converge to generating a zero signal. On the other hand, using a loss of 90% SNR + 10% SI-SNR in the binaural case, caused one of the channels to output a very low-amplitude signal relative to the other channel as SI-SNR is insensitive to the signal gains. We confirmed that the signal is spectrally meaningful even though the magnitude is wrong. As a result, only SI-SNRi and ΔITD results are meaningful for the Conv-TasNet model. ΔILD computation resulted in infinity, so we omit it in our table.

In Table. 2, we observe that the dual-channel framework is competitive with the parallel and single-channel frameworks in terms of SI-SNRi, while outperforming in ΔILD. With regard to ΔITD, it resulted in a very marginal increase. These results intuitively make sense because the dual-channel framework has a sample-aligned common representation for both left and right channels. As a result, it can maintain the relative amplitudes of the left and right channels. On the other hand, the parallel and single-channel frameworks have separate branches that independently process different channels, facilitating maintaining the sample alignment with the respective channels. This phenomenon is more notable for the single-channel framework, where the SI-SNRi and ΔITD are promising but the ΔILD is poor, as there is no cross-communication between the left and right channel processings. Finally, we note that our dual-channel framework requires only a little more than 50% of the runtime required by their parallel or single-channel counterparts, making it a good practical choice for our semantic hearing system. Finally, we note that our dual-channel framework uses 240 MFLOPS, while vanilla Waveformer uses 357 MFLOPS across the two microphones.

For our causal model, the receptive field is exclusively the past audio. Hence, it has no effect on the algorithmic latency. The algorithmic latency of our model is the sum of chunk size, KL, and the lookahead of the input convolution, L, where L is the stride of the input convolution (§3.2.2). Table 2 uses stride $L = 32$ samples and $K = 13$, resulting in a chunk size of $KL = 416$ samples and lookahead $L = 32$ samples. This is equivalent to 9.4 ms and 0.7 ms, respectively. Table 3 shows the performance of our binaural model with various chunk sizes to understand the effect of algorithmic latency on performance. The results show that our model achieves reasonable performance with an algorithmic latency as low as 1.4 ms. Thus with ASIC implementations, such as those in hearing aids, we could envision ultra-low-latency semantic hearing systems.

During our in-the-wild evaluations, users freely moved their heads and encountered mobile sources (eg. sirens). The model also

Table 3: Effect of algorithmic latency on the performance. *Proposed system with end-to-end latency ~20 ms.

Chunk size (samples)	Algorithmic latency (ms)	SI-SNRi (dB)
32	1.4	6.59
128	3.6	6.83
256	6.5	7.18
416*	10.1	7.42

Table 4: Comparison of performance in the presence of relative angular motion between listener and sound sources. Dual-ch model with $D = 256$ is used for this evaluation.

Angular velocity (°/s)	Reverb.	SI-SNRi (dB)	ΔITD (μs)	ΔILD (dB)
30	No	7.95	34.26	0.58
	Yes	7.88	103.45	0.43
60	No	7.91	49.49	0.57
	Yes	7.98	98.23	0.49
90	No	7.87	58.67	0.54
	Yes	8.00	99.83	0.43

performed robustly without glitches during evaluations by human testers. The model adapted quickly to relative motion because it outputs small chunks (<10ms) while updating its internal state. The model can also utilize spatial positions in the trajectory that have better level differences between L and R channels. In addition to that qualitative evaluation, Table 4 provides a quantitative comparison of the performance for different amounts of relative angular motion between the listener and sound sources. For this comparison, we use the dual-ch model with $D = 256$ dimensions. We simulate motion using Steam Audio SDK [7], which simulates binaural motion given an HRTF file in the SimpleFreeFieldHRIR format [6]. We performed controlled experiments with different angular velocities in both anechoic and reverberant environments, with sources moving from a random position on an arc with the given angular velocity. We used the CIPIC [8] HRTF dataset for anechoic simulations and RRBRIR [26] BRIR dataset for reverberant simulations as they provide impulse responses in the SimpleFreeFieldHRIR format. We synthesize the binaural audio in frames of 1024 samples by convolving with an interpolated impulse response using bilinear interpolation at every frame. Since the ILD and ITD are now time-varying, we compute the ΔILD and ΔITD in chunks of 250 ms, discarding any chunks where the clean signal is silent on both channels, and take the mean across the remaining chunks. We observed that in the presence of motion, SI-SNRi and ΔILD are marginally better as the model is able to better leverage the level differences between L and R at different relative angular positions while achieving lower ΔITD in the anechoic case and slightly higher ΔITD in reverberant scenarios.

4.5 Proof-of-concept user interface

Finally, a natural question is: how does the user pick between classes? To answer this question, we prototyped an iOS app with three different user interfaces for sound selection: Speech, Text and Toggle switch grid of sounds (Fig. 13), and evaluated their accuracy, speed, and ease of use.

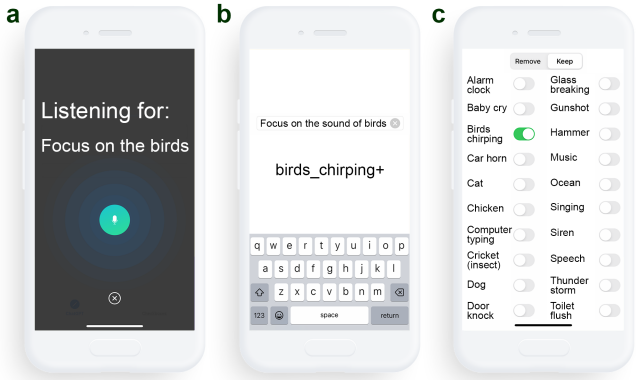


Figure 13: User interface designs for target sound selection on a smartphone. Each design uses a different input method to capture the user's intent: (left to right) speech, text and toggle switches. The first two interfaces use ChatGPT API to convert natural language to class inputs for our system.

For the speech and text interfaces, our goal was to investigate if the ChatGPT API for phones [4] could be used to convert natural language (*I want to listen to ambulance sounds*) into known sound class inputs to our system (*siren+*). To do this, we initialized ChatGPT using the prompt: *Here is a list of sound classes: ['alarm_clock', 'baby_cry', [...] I will provide you a sentence that involves keeping or removing one of these sound classes. I want you to output the sound class from the list that most closely matches (semantically) the sounds in the sentence. If there are no classes that are sufficiently close, output 'na'. Please do not output any other characters. If you find a close class and if the sentence involves keeping sound from this label, append a '+' to your output, otherwise, append '-'. For example, if I say 'mute cat' you should say 'cat-'. If I say 'mute cow' you should say 'na'.*

Ten participants were presented each of them with ten scenes of the following form: *Your infant's cries break the silence. Your phone plays a melody. The rustle of the wind is audible.* We asked participants to select a single sound event to add or remove, and convey their intent to the app with the three user interfaces (UIs). To evaluate the accuracy of each interface, we compared the sound event selected through each UI with our best interpretation of what the user said. Agreement rates were 92% for Speech and Text, and 93% for Toggle switch. For Speech and Text, disagreement was due to confusion by ChatGPT (e.g. *The toilet is too loud* mapped to *toilet_flush+*), or when ChatGPT would map selected sounds not in the dataset to a similar sound in the dataset (e.g. *wind* and *fountain sounds* were mapped to *ocean*). For the Toggle switch, disagreement occurred when the intended sound class could not be found.

The mean time taken to convey intent was shortest for Speech (5.5 ± 1.0s), then Toggle (6.3 ± 3.3s) and longest for Text (8.3 ± 3.7s). Preference ratings (1=very unlikely, 5=very likely) were highest for Speech (4.0 ± 1.1), then Text (2.9 ± 1.2), and lowest for Toggle (2.7 ± 1.4). These findings suggest that from an user interface perspective, Speech would be a practical interface choice and would scale better than the Toggle interface as the number of supported classes increases. One participant noted that they would prefer the Text interface when using the system in a public setting even if it took a longer time to input their intent.

5 LIMITATIONS AND DISCUSSION

As shown in Table. 1, we have an imbalance in the number of examples across classes. For instance, the "speech" class has 494 training examples, while "car horn" has only 60 training examples. Collecting more examples across all classes can potentially improve the performance. Finally, some classes may be inherently harder to separate. For example, music and human speech share many characteristics, including vocal sounds and harmonicity. Thus, despite having a larger number of training examples, it is difficult for our model to perform tasks such as separating the speech of the person around the wearer in the presence of background music that also has vocals. Similarly, it can be challenging to separate music from other classes like alarm clock sounds or bird chirping. Additionally, our training methodology does not utilize any real-world data with our hardware. Nevertheless, our real-world testing results demonstrate the generalization capabilities to our hearable hardware as well as unseen real-world environments. However, it is still possible that collecting training data in the real-world scenarios as well as with actual hardware can help improve the system performance.

Another limitation is the form factor of the hearable hardware we used in our evaluations where we used binaural earphones in addition to a noise canceling headset. The form factor could be simplified if we used a single device for recording and playback. Currently, there are commercial noise cancelling headsets that provide user access to the microphone data, such as the Sennheiser AMBEO Smart Headset, which we found after our evaluations. Our system implemented on such a device would have fewer wires and would directly connect to the smartphone at a single point, without the need for an additional pair of binaural earphones.

Binaural target sound extraction can also be used to subtract the target sounds and play the residual sounds into the ear. Fig. 10 shows the results for subtracting a target sound (e.g., computer typing or hammer) to focus on the human speech. This can be beneficial when the user knows the specific type of environmental noise that they feel annoying (e.g., computer typing in an office room) as this approach would remove only the specified noise and thus allow the user to focus on the speech and the other sounds in the environment.

For proof-of-concept demonstration, we have implemented our neural network on a connected smartphone. While wired headsets connected to the smartphones are an important use case for practical applications and can benefit from our implementation, extending our system to wireless headsets requires integrating computation with the headset hardware itself. This is likely feasible given the ultra-low power multicore embedded GPUs that are being designed for wearable devices [5]. Further, with recent developments in custom silicon for on-chip deep learning for speech and natural language processing [62], it is likely that commercial hearable devices for semantic hearing would use such custom silicon to reduce both the power consumption of the wearable device and the end-to-end latency.

6 CONCLUSION

This paper takes an important first step towards realizing real-time programming of acoustic scenes on binaural hearable devices using the semantic description of sounds. At its core are two key technical contributions: 1) the first binaural target sound extraction neural network. Our network can run in real-time, using 10 ms or less of audio blocks, while preserving the spatial information, and 2) a training methodology that allows our system to generalize to unseen real-world environments. In-the-wild experiments with participants show that our proof-of-concept hardware-software system can preserve the directions of the target sounds and separate these sounds in real-time from both the background noise and other sounds in the environment.

REFERENCES

[1] 2023. Apple AirPods. https://www.apple.com/airpods/. (2023).
[2] 2023. Audio Latency Meter for iOS. https://onyx3.com/LatencyMeter/. (2023).
[3] 2023. Customize Transparency mode for AirPods Pro. https://support.apple.com/guide/airpods/customize-transparency-mode-dev966f5f818/web. (2023).
[4] 2023. GPT models. https://platform.openai.com/docs/guides/gpt. (2023).
[5] 2023. GPU-WEAR, Ultra-low power heterogeneous Graphics Processing Units for Wearable/IoT devices. https://cordis.europa.eu/project/id/717850. (2023).
[6] 2023. SimpleFreeFieldHRIR. https://www.sofaconventions.org/mediawiki/index.php. (2023).
[7] 2023. Steam Audio SDK. https://valvesoftware.github.io/steam-audio/. (2023).
[8] V.R. Algazi, R.O. Duda, D.M. Thompson, and C. Avendano. 2001. The CIPIC HRTF database. In *Proceedings of the 2001 IEEE Workshop on the Applications of Signal Processing to Audio and Acoustics (Cat. No.01TH8575)*. 99–102. https://doi.org/10.1109/ASPAA.2001.969552
[9] Junjie Bai, Fang Lu, Ke Zhang, et al. 2019. ONNX: Open Neural Network Exchange. https://github.com/onnx/onnx. (2019).
[10] Luca Brayda, Federico Traverso, Luca Giuliani, Francesco Diotalevi, Stefania Repetto, Sara Sansalone, Andrea Trucco, and Giulio Sandini. 2015. Spatially selective binaural hearing aids. In *Adjunct Proceedings of IMWUT/ISWC*.
[11] Nam Bui, Nhat Pham, Jessica Jacqueline Barnitz, Zhanan Zou, Phuc Nguyen, Hoang Truong, Taeho Kim, Nicholas Farrow, Anh Nguyen, Jianliang Xiao, Robin Deterding, Thang Dinh, and Tam Vu. 2021. EBP: An Ear-Worn Device for Frequent and Comfortable Blood Pressure Monitoring. *Commun. ACM* (2021).
[12] Justin Chan, Nada Ali, Ali Najafi, Anna Meehan, Lisa Mancl, Emily Gallagher, Randall Bly, and Shyamnath Gollakota. 2022. An off-the-shelf otoacoustic-emission probe for hearing screening via a smartphone. *Nature Biomedical Engineering* 6 (10 2022), 1–11. https://doi.org/10.1038/s41551-022-00947-6
[13] Justin Chan, Sharat Raju, Rajalakshmi Nandakumar, Randall Bly, and Shyamnath Gollakota. 2019. Detecting middle ear fluid using smartphones. *Science Translational Medicine* 11 (05 2019), eaav1102. https://doi.org/10.1126/scitranslmed.aav1102
[14] Ishan Chatterjee, Maruchi Kim, Vivek Jayaram, Shyamnath Gollakota, Ira Kemelmacher, Shwetak Patel, and Steven M Seitz. 2022. ClearBuds: wireless binaural earbuds for learning-based speech enhancement. In *ACM MobiSys*.
[15] Marc Delcroix, Jorge Bennasar Vázquez, Tsubasa Ochiai, Keisuke Kinoshita, Yasunori Ohishi, and Shoko Araki. 2022. SoundBeam: Target sound extraction conditioned on sound-class labels and enrollment clues for increased performance and continuous learning. In *arXiv*.
[16] Simon Doclo, Sharon Gannot, Marc Moonen, Ann Spriet, Simon Haykin, and KJ Ray Liu. 2010. Acoustic beamforming for hearing aid applications. *Handbook on array processing and sensor networks* (2010), 269–302.
[17] Sefik Emre Eskimez, Takuya Yoshioka, Huaming Wang, Xiaofei Wang, Zhuo Chen, and Xuedong Huang. 2022. Personalized speech enhancement: New models and comprehensive evaluation. In *IEEE ICASSP*.
[18] Eduardo Fonseca, Xavier Favory, Jordi Pons, Frederic Font, and Xavier Serra. 2022. FSD50K: An Open Dataset of Human-Labeled Sound Events. (2022). arXiv:cs.SD/2010.00475
[19] Ruohan Gao and Kristen Grauman. 2019. Co-separating sounds of visual objects. In *IEEE /CVF ICCV*.
[20] Jort F. Gemmeke, Daniel P. W. Ellis, Dylan Freedman, Aren Jansen, Wade Lawrence, R. Channing Moore, Manoj Plakal, and Marvin Ritter. 2017. Audio Set: An ontology and human-labeled dataset for audio events. In *IEEE ICASSP*.
[21] Beat Gfeller, Dominik Roblek, and Marco Tagliasacchi. 2021. One-shot conditional audio filtering of arbitrary sounds. In *ICASSP*. IEEE.
[22] Ritwik Giri, Shrikant Venkataramani, Jean-Marc Valin, Umut Isik, and Arvindh Krishnaswamy. 2021. Personalized percepnet: Real-time, low-complexity target voice separation and enhancement. In *arXiv*.
[23] Rongzhi Gu, Jian Wu, Shi-Xiong Zhang, Lianwu Chen, Yong Xu, Meng Yu, Dan Su, Yuexian Zou, and Dong Yu. 2019. End-to-End Multi-Channel Speech Separation. In *arXiv*. arXiv:cs.SD/1905.06286

[24] Rishabh Gupta, Rishabh Ranjan, Jianjun He, Woon-Seng Gan, and Santi Peksi. 2020. Acoustic transparency in hearables for augmented reality audio: Hear-through techniques review and challenges. In *Audio Engineering Society Conference on Audio for Virtual and Augmented Reality*.

[25] Cong Han, Yi Luo, and Nima Mesgarani. 2020. Real-time binaural speech separation with preserved spatial cues. In *arXiv*. arXiv:eess.AS/2002.06637

[26] IoSR-Surrey. 2016. IoSR-surrey/realroombrirs: Binaural impulse responses captured in real rooms. https://github.com/IoSR-Surrey/RealRoomBRIRs. (2016).

[27] IoSR-Surrey. 2023. Simulated Room Impulse Responses. https://iosr.uk/software/index.php. (2023).

[28] Dhruv Jain, Kelly Mack, Akli Amrous, Matt Wright, Steven Goodman, Leah Findlater, and Jon E. Froehlich. 2020. HomeSound: An Iterative Field Deployment of an In-Home Sound Awareness System for Deaf or Hard of Hearing Users. In *ACM CHI*.

[29] Dhruv Jain, Hung Ngo, Pratyush Patel, Steven Goodman, Leah Findlater, and Jon Froehlich. 2020. SoundWatch: Exploring Smartwatch-Based Deep Learning Approaches to Support Sound Awareness for Deaf and Hard of Hearing Users. In *ACM SIGACCESS ASSETS*.

[30] Andreas Jansson, Eric J. Humphrey, Nicola Montecchio, Rachel M. Bittner, Aparna Kumar, and Tillman Weyde. 2017. Singing Voice Separation with Deep U-Net Convolutional Networks. In *ISMIR*.

[31] Wenyu Jin, Tim Schoof, and Henning Schepker. 2022. Individualized Hear-Through For Acoustic Transparency Using PCA-Based Sound Pressure Estimation At The Eardrum. In *ICASSP*.

[32] Gabriel Jorgewich-Cohen, Simon Townsend, Linilson Padovese, Nicole Klein, Peter Praschag, Camila Ferrara, Stephan Ettmar, Sabrina Menezes, Arthur Varani, Jaren Serano, and Marcelo Sánchez-Villagra. 2022. Common evolutionary origin of acoustic communication in choanate vertebrates. *Nature Communications* 13 (10 2022). https://doi.org/10.1038/s41467-022-33741-8

[33] Kevin Kilgour, Beat Gfeller, Qingqing Huang, Aren Jansen, Scott Wisdom, and Marco Tagliasacchi. 2022. Text-Driven Separation of Arbitrary Sounds. In *arXiv*.

[34] Gierad Laput, Karan Ahuja, Mayank Goel, and Chris Harrison. 2018. Ubicoustics: Plug-and-Play Acoustic Activity Recognition. In *ACM UIST*.

[35] Xubo Liu, Haohe Liu, Qiuqiang Kong, Xinhao Mei, Jinzheng Zhao, Qiushi Huang, Mark D Plumbley, and Wenwu Wang. 2022. Separate What You Describe: Language-Queried Audio Source Separation. In *arXiv*.

[36] Hong Lu, Wei Pan, Nicholas D. Lane, Tanzeem Choudhury, and Andrew T. Campbell. 2009. SoundSense: Scalable Sound Sensing for People-Centric Applications on Mobile Phones. In *ACM MobiSys*.

[37] Jian Luo, Jianzong Wang, Ning Cheng, Edward Xiao, Xulong Zhang, and Jing Xiao. 2022. Tiny-Sepformer: A Tiny Time-Domain Transformer Network for Speech Separation. In *arXiv*.

[38] Yi Luo and Nima Mesgarani. 2019. Conv-tasnet: Surpassing ideal time–frequency magnitude masking for speech separation. *IEEE/ACM transactions on audio, speech, and language processing* (2019).

[39] Dong Ma, Andrea Ferlini, and Cecilia Mascolo. 2021. OESense: Employing Occlusion Effect for in-Ear Human Sensing. In *MobiSys*.

[40] Tobias May, Steven van de Par, and Armin Kohlrausch. 2011. A Probabilistic Model for Robust Localization Based on a Binaural Auditory Front-End. *IEEE Transactions on Audio, Speech, and Language Processing* 19, 1 (2011), 1–13. https://doi.org/10.1109/TASL.2010.2042128

[41] Emma McDonnell, Soo Hyun Moon, Lucy Jiang, Steven Goodman, Raja Kushalnaga, Jon Froehlich, and Leah Findlater. 2023. "Easier or Harder, Depending on Who the Hearing Person Is": Codesigning Videoconferencing Tools for Small Groups with Mixed Hearing Status. In *ACM CHI*.

[42] Annamaria Mesaros, Toni Heittola, and Tuomas Virtanen. 2018. A multi-device dataset for urban acoustic scene classification. In *DCASE*. https://arxiv.org/abs/1807.09840

[43] Vimal Mollyn, Karan Ahuja, Dhruv Verma, Chris Harrison, and Mayank Goel. 2022. SAMoSA: Sensing Activities with Motion and Subsampled Audio. *IMWUT* (2022).

[44] Furnon Nicolas. 2020. Noise files for the DISCO dataset. (2020). https://github.com/nfurnon/disco.

[45] Tsubasa Ochiai, Marc Delcroix, Yuma Koizumi, Hiroaki Ito, Keisuke Kinoshita, and Shoko Araki. 2020. Listen to What You Want: Neural Network-based Universal Sound Selector, In arXiv. *arXiv e-prints*. arXiv:eess.AS/2006.05712

[46] Yuki Okamoto, Shota Horiguchi, Masaaki Yamamoto, Keisuke Imoto, and Yohei Kawaguchi. 2022. Environmental Sound Extraction Using Onomatopoeic Words. In *IEEE ICASSP*.

[47] Aaron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu. 2016. WaveNet: A Generative Model for Raw Audio. In *arXiv*. https://doi.org/10.48550/ARXIV.1609.03499

[48] Tom Le Paine, Pooya Khorrami, Shiyu Chang, Yang Zhang, Prajit Ramachandran, Mark A. Hasegawa-Johnson, and Thomas S. Huang. 2016. Fast Wavenet Generation Algorithm. In *arXiv*. arXiv:cs.SD/1611.09482

[49] Amy Pavel, Gabriel Reyes, and Jeffrey P. Bigham. 2020. Rescribe: Authoring and Automatically Editing Audio Descriptions. In *ACM UIST*.

[50] Mike Peterson. 2021. Apple AirPods, Beats dominated audio wearable market in 2020. https://appleinsider.com/articles/21/03/30/apple-airpods-beats-dominated-audio-wearable-market-in-2020. (2021).

[51] Karol J. Piczak. 2015. ESC: Dataset for Environmental Sound Classification. In *ACM Multimedia*.

[52] Jay Prakash, Zhijian Yang, Yu-Lin Wei, Haitham Hassanieh, and Romit Roy Choudhury. 2020. EarSense: Earphones as a Teeth Activity Sensor. In *MobiCom*.

[53] Zafar Rafii, Antoine Liutkus, Fabian-Robert Stöter, Stylianos Ioannis Mimilakis, and Rachel Bittner. 2017. MUSDB18 - a corpus for music separation. (2017).

[54] Jonathan Le Roux, Scott Wisdom, Hakan Erdogan, and John R. Hershey. 2018. SDR - half-baked or well done?. In *arXiv*.

[55] Steve Rubin, Floraine Berthouzoz, Gautham J. Mysore, Wilmot Li, and Maneesh Agrawala. 2013. Content-Based Tools for Editing Audio Stories. In *ACM UIST*.

[56] Justin Salamon, Duncan MacConnell, Mark Cartwright, Peter Li, and Juan Pablo Bello. 2017. Scaper: A library for soundscape synthesis and augmentation. In *WASPAA*. https://doi.org/10.1109/WASPAA.2017.8170052

[57] Darius Satongar, Yiu W Lam, Chris Pike, et al. 2014. The Salford BBC Spatially-sampled Binaural Room Impulse Response dataset. (2014).

[58] Sheng Shen, Nirupam Roy, Junfeng Guan, Haitham Hassanieh, and Romit Roy Choudhury. 2018. MUTE: Bringing IoT to Noise Cancellation. In *ACM SIGCOMM*. https://doi.org/10.1145/3230543.3230550

[59] Michael A Stone and Brian CJ Moore. 1999. Tolerable hearing aid delays. I. Estimation of limits imposed by the auditory path alone using simulated hearing losses. *Ear and Hearing* 20, 3 (1999), 182–192.

[60] Cem Subakan, Mirco Ravanelli, Samuele Cornell, Mirko Bronzi, and Jianyuan Zhong. 2021. Attention is all you need in speech separation. In *IEEE ICASSP*.

[61] Cem Subakan, Mirco Ravanelli, Samuele Cornell, Frédéric Lepoutre, and François Grondin. 2022. Resource-Efficient Separation Transformer. In *arXiv*.

[62] Thierry Tambe, En-Yu Yang, Glenn Ko, Yuji Chai, Coleman Hooper, Marco Donato, Paul Whatmough, Alexander Rush, David Brooks, and Gu-Yeon Wei. 2022. A 16-nm SoC for Noise-Robust Speech and NLP Edge AI Inference With Bayesian Sound Source Separation and Attention-Based DNNs. *IEEE Journal of Solid-State Circuits* (2022). https://doi.org/10.1109/JSSC.2022.3179303

[63] Noriyuki Tonami, Keisuke Imoto, Ryotaro Nagase, Yuki Okamoto, Takahiro Fukumori, and Yoichi Yamashita. 2022. Sound Event Detection Guided by Semantic Contexts of Scenes. In *arXiv*. arXiv:cs.SD/2110.03243

[64] Vesa Valimaki, Andreas Franck, Jussi Ramo, Hannes Gamper, and Lauri Savioja. 2015. Assisted Listening Using a Headset: Enhancing audio perception in real, augmented, and virtual environments. *IEEE Signal Processing Magazine* 32, 2 (2015), 92–99. https://doi.org/10.1109/MSP.2014.2369191

[65] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention Is All You Need. In *arXiv*. https://doi.org/10.48550/ARXIV.1706.03762

[66] Bandhav Veluri, Justin Chan, Malek Itani, Tuochao Chen, Takuya Yoshioka, and Shyamnath Gollakota. 2023. Real-Time Target Sound Extraction. In *IEEE ICASSP*.

[67] Anran Wang, Maruchi Kim, Hao Zhang, and Shyamnath Gollakota. 2022. Hybrid Neural Networks for On-Device Directional Hearing. *AAAI* (2022). https://ojs.aaai.org/index.php/AAAI/article/view/21394

[68] Yuntao Wang, Jiexin Ding, Ishan Chatterjee, Farshid Salemi Parizi, Yuzhou Zhuang, Yukang Yan, Shwetak Patel, and Yuanchun Shi. 2022. FaceOri: Tracking Head Position and Orientation Using Ultrasonic Ranging on Earphones. In *ACM CHI*.

[69] Xudong Xu, Bo Dai, and Dahua Lin. 2019. Recursive visual sound separation using minus-plus net. In *IEEE/CVF ICCV*.

[70] Xuhai Xu, Haitian Shi, Xin Yi, WenJia Liu, Yukang Yan, Yuanchun Shi, Alex Mariakakis, Jennifer Mankoff, and Anind K. Dey. 2020. EarBuddy: Enabling On-Face Interaction via Wireless Earbuds. In *CHI*. https://doi.org/10.1145/3313831.3376836

[71] Koji Yatani and Khai N. Truong. 2012. BodyScope: A Wearable Acoustic Sensor for Activity Recognition. In *UbiComp*.