WavePurifier: Purifying Audio Adversarial Examples via Hierarchical Diffusion Models

Hanqing Guo[1,2], Guangjing Wang[1], Bocheng Chen[1], Yuanda Wang[1], Xiao Zhang[3], Xun Chen [4], Qiben Yan[1], Xiao Li[1]
[1]Michigan State University, [2]University of Hawaii at Manoa, [3]Duke University, [4]Samsung Research America

{guohanqi, wanggu22, chenboc1, wangy208, qyan}@msu.edu, xiao.zhang@duke.edu, xunchen@outlook.com, lxiao@cse.msu.edu

ABSTRACT

In this paper, we propose WavePurifier, an audio purification framework to defend against audio adversarial attacks. Audio adversarial attacks craft adversarial examples or perturbations to attack the automated speech recognition (ASR) models. Although existing defense mechanisms can detect such attacks and raise alarms, they fail to recover or maintain benign commands. Consequently, this leads to the denial of users' benign commands. Different than existing defenses, WavePurifier aims to purify adversarial examples, thereby rectifying the user's benign commands. We find that the forward diffusion process of the diffusion model effectively eliminates perturbations, whereas the reverse diffusion process restores benign speech. Based on this, we develop a hierarchical diffusion model to defend against audio adversarial examples. This model is capable of purifying different spectrogram bands to varying degrees. To validate the performance of WavePurifier, we purify the adversarial examples from 3 different adversarial attacks in 140 distinct settings. In total, we collect 78,864 diffused spectrograms and 21,000 purified audios. Then, we evaluate WavePurifier on 2 different ASR models, 4 commercial speech-to-text APIs, 2 real-world attack scenarios, and compare them against 7 existing defense approaches. Our result shows that WavePurifier is a universal framework, demonstrating adaptability across diverse attacks with the same hyperparameters. Notably, WavePurifier outperforms existing methods with the lowest character error rate (CER), word error rate (WER), and a high purification success rate against different attacks.

ACM MobiCom '24, November 18–22, 2024, Washington D.C., DC, USA

ACM ISBN 979-8-4007-0489-5/24/11.
https://doi.org/10.1145/3636534.3690692

CCS CONCEPTS

• **Security and privacy → Security services**.

KEYWORDS

Speech Recognition Services; Purification; Diffusion Model.

ACM Reference Format:
Hanqing Guo[1,2], Guangjing Wang[1], Bocheng Chen[1], Yuanda Wang[1], Xiao Zhang[3], Xun Chen [4], Qiben Yan[1], Xiao Li[1]. 2024. WavePurifier: Purifying Audio Adversarial Examples via Hierarchical Diffusion Models. In *The 30th Annual International Conference on Mobile Computing and Networking (ACM MobiCom '24), November 18–22, 2024, Washington D.C., DC, USA.* ACM, New York, NY, USA, 15 pages. https://doi.org/10.1145/3636534.3690692

1 INTRODUCTION

Voice communication is a type of human-computer interaction that enables hands-free control and benefits visually impaired users. With recent advancements in Artificial Intelligence (AI), Automatic Speech Recognition (ASR) has become increasingly prevalent in our daily lives. These ASR models have been integrated into Intelligent Voice Control devices to provide voice assistant services like Siri [43], Google Assistant [14], and smart speakers like Google Home [15] and Amazon Echo [3]. Additionally, companies are increasingly relying on ASR-powered intelligent interactive voice systems to handle customer service inquiries and improve support efficiency. As the deployment of ASR systems grows, their security concerns are attracting heightened attention from researchers.

Audio adversarial attacks: The adversarial attack was discovered and demonstrated in image recognition tasks [13, 41]. Similarly, ASR systems are also susceptible to adversarial attacks. The audio adversarial examples are crafted to fool the ASR system, for example, a benign command "open the door" with crafted perturbations could be transcribed to "browse to evil dot com". Prior studies demonstrate the audio adversarial attacks in white-box scenario [5, 8], black-box scenario [2, 42], over-the-air scenario [6, 7, 28, 35, 48, 50], and human-in-the-loop scenario [20].

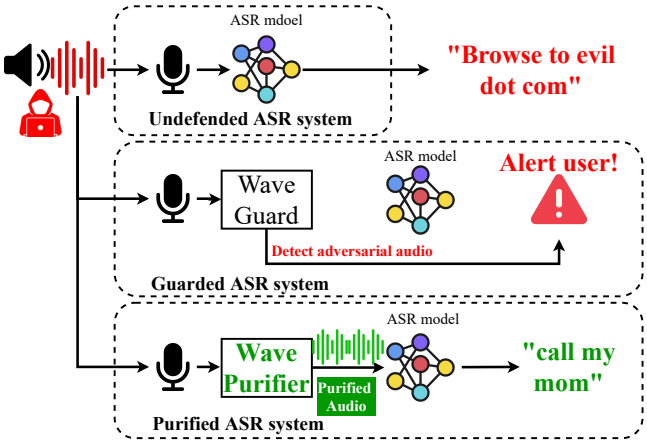


Figure 1: Demonstration of an undefended ASR system, a guarded ASR system, and a purified ASR system.

Prior defenses: To defend against adversarial attacks in the image domain, researchers have proposed to use image transformations [17, 29] such as format compression, quantization, and smoothing to disrupt the imperceptible perturbations. However, such defenses may cause image quality degradation and become ineffective for adaptive attacks (e.g., the adversary has the knowledge of the defenses, so he/she can craft robust AEs to bypass the static defenses). Inspired by the image transformation defenses, a recent study [23] proposed to detect the audio adversarial examples by signal processing methods. Since the audio AEs are sensitive to the audio transformation, by comparing the transcriptions before and after the audio transformation, they can detect adversarial audio. Although such defense has demonstrated satisfactory detection accuracy, it does not ensure audio quality, fails to restore benign commands, and is ineffective against strong perturbations. More recently, researchers proposed AudioPure [47], which leverages the diffusion model to purify audio. Even though this method shows great success in recovering the perturbed single-word speech, however, they did not demonstrate their capability for purifying real-world long speech commands, and their purification performance against advanced existing speech adversarial attacks (e.g., SpecPatch [20], C&W [5]), which target ASR systems (e.g., Deepspeech [21], Google Lingvo [36]), remains unknown.

WavePurifier: In this work, we propose WavePurifier, an end-to-end framework that purifies the incoming audio commands, to defend against audio adversarial attacks, and recovers the users' original commands. This is important in many scenarios, for example, if the adversary plays the perturbation to alter the user's commands [19, 20], our defense can not only detect the potential threat of the attack but also repair the users' benign speech. This framework can also be used for audio datasets maintenance, to examine and restore the clean audio samples.

Fig. 1 depicts a defense scenario of WavePurifier. For a benign command "call my mom", the adversary adds perturbation to change the ASR transcription into "browse to evil dot com". An unprotected ASR system will recognize the target command, while WaveGuard [23] will detect the audio adversarial example and alert the user. In contrast, WavePurifier will purify the audio adversarial example and produce clean audio that recovers the benign command for the ASR model. We leverage the *diffusion model* to achieve our goal.

In this paper, we make the following contributions.

- **Purification Framework:** We design the first audio AE purification framework for purifying audio adversarial examples against ASR systems. Our framework leverages the hierarchical diffusion models to purify audio across various frequency bands, followed by the application of denoising techniques to remove any residual noise. Our demo and code is available at https://wavepurifier.github.io.
- **Theoretical Proof:** We provide the theoretical proof to demonstrate that the diffusion model can be used to purify audio adversarial examples. We extend the theorem to include additional factors that 1) clarify the rationale for using the diffusion model to neutralize attacks, and 2) reveal the identification of the most effective purification steps.
- **Comprehensive Evaluation:** We reproduce three recent audio adversarial attacks and generate 300 audio AEs. We purify them with 140 different settings for each attack, resulting in 78,000 diffusion-purified images and 21,000 purified audios. We compare WavePurifier with seven existing defenses and conclude that our framework outperforms existing defense methods on purification tasks, with a high purification success rate and low character error rate.

2 BACKGROUND

2.1 Audio Adversarial Attacks

Audio adversarial attack aims to craft an AE $x_0+\delta$, in order to deceive the ASR model $f(\cdot)$ to make false prediction [5]. For a targeted attack, the adversary can specify the prediction result as y_t. For an untargeted attack, the adversary's goal is to mistranscribe the benign audio into random sentences. The generation of AE can be formulated as an optimization problem as follows:

$$minimize \quad \mathcal{L}(x_0+\delta) := \mathcal{D}(f(x_0+\delta), y_t). \tag{1}$$

The goal of Eq. (1) is to minimize $\mathcal{L}(x_0+\delta)$ under the constraint that $||\delta||_2 < \epsilon$, where $\mathcal{L}(\cdot)$ denotes the loss function,

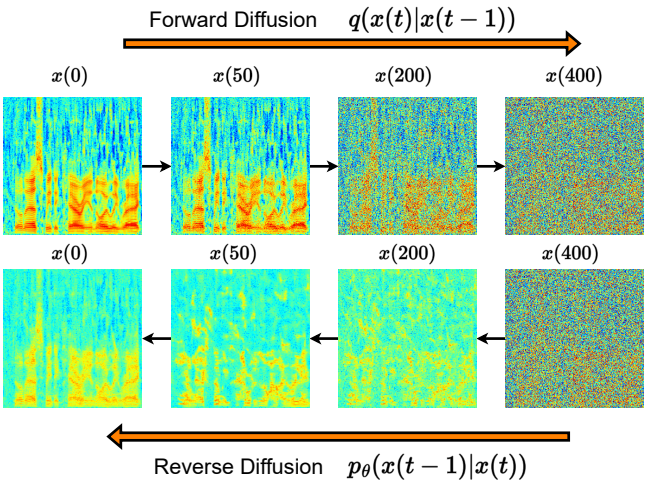


Figure 2: Depiction of the diffusion model for forward and reverse the process.

which uses a distance function $\mathcal{D}(\cdot)$ to measure the disparity between $f(x_0+\delta)$ and y_t, $||\cdot||_2$ is the L2 norm, and ϵ is used to control the amplitude of perturbation.

2.2 Diffusion Model

The diffusion model is a generative model to generate high-quality images. Prior studies have demonstrated that the diffusion model beat GANs in image generation [10, 44] and achieved great success in audio synthesis [24]. Recently, OpenAi and stability.ai have released their AI creative tool DALL-E2 [33] and Stable Diffusion [40], both powered by the diffusion model. Fundamentally, the diffusion models are composed of two processes.

Forward diffusion: The forward diffusion is to gradually add Gaussian noise to the real data. Given an original sample $x(0)$ with distribution $q(x)$, the single-step forward diffusion process can be defined as follows:

$$x(t) = x(t-1) + \mathcal{N}(0,1) \tag{2}$$

where $x(t)$ is the original sample at forward step t, and $x(t-1)$ is the previous sample of $x(t)$. $\mathcal{N}(0,1)$ denotes the Gaussian distribution. The conditional probability distribution can be defined as follows:

$$q(x(t), x(t-1)) = \mathcal{N}(x(t); \sqrt{1-\beta_t}x(t-1), \beta_t I) \tag{3}$$

Where $\beta_t \in (0,1)$ is the step size. It means that when giving the $x(t-1)$, the transition from $x(t-1)$ to $x(t)$ can be represented by a Gaussian distribution, with the mean as $\sqrt{1-\beta_t}x(t-1)$ and variance as $\beta_t I$. We can find that when t becomes larger and larger, the $x(t)$ will be equivalent to an isotropic Gaussian distribution.

Reverse diffusion: The reverse diffusion is to recover the data from a noise distribution. Theoretically, the reverse conditional probability distribution can be represented as follows:

$$p_\theta(x(t-1)|x(t)) = \mathcal{N}(x(t-1); \mu_\theta(x(t),t), \Sigma_\theta(x(t),t)) \tag{4}$$

In this equation, the mean $\mu_\theta(x(t),t)$ and variance $\Sigma_\theta(x(t),t))$ of the Gaussian distribution are unknown, so they need to be learned from the training process. Specifically, the model learns the joint distribution p_θ from $x(T)$ to $x(0)$. Usually, prior works [38, 44] solve it with the reverse-time stochastic differential equation (SDE), and a parameterized neural network to estimate the gradient of the forward diffusion. Once the diffusion model is well-trained, it can generate images from noise because it learns the probability distribution, so it is capable of producing a high-quality image $x(0)$ from the Gaussian noise $x(T)$.

In Fig. 2, We train a diffusion model using spectrogram data, so it learns to reverse the transition probability in order to generate a spectrogram. In the first row, we display the original spectrogram $x(0)$ with increasing amounts of noise added using $q(x(t), x(t-1))$. It is observed that even after adding noise in 50 and 200 steps, the general shape of the spectrogram can still be recognized, though with blurred energy boundaries. However, when the forward diffusion step reaches 400, the spectrogram transforms into noise. Next, we perform the reverse diffusion process from the noise-like $x(400)$. By providing the learned $p_\theta(x(t-1)|x(t))$, our diffusion model re-generates new spectrograms based on the noise distribution. From $x(400)$ to $x(200)$, we can find the dense noise has been greatly reduced, replaced by the light-weight noise. Besides, some of the formants have been exposed, showing clear energy in $x(200)$. When we keep running the reverse program, we find the noise is further suppressed, and more spectrogram details are revealed. It is worth noting that the reverse process will introduce a new random feature, which might create a unique spectrogram that is different from the original one (e.g., $x(0)$). For example, in the final recovered $x(0)$, we can find it includes the high-frequency energy that is not shown in the previous $x(50)$. Besides that, it contains more noise compared to the original spectrogram $x(0)$. Based on this preliminary study, we realized that the diffusion model can be used to produce spectrograms from noise, however, the quality of the generated audio is not guaranteed because it might introduce more noise degrading the audio's acoustic feature.

2.3 Threat Model

Our defense aims to purify audio adversarial examples and recover their benign transcriptions. By doing so, we can defend against both untargeted audio adversarial attacks and the targeted attack.

Attacker's Capability: We assume the attacker can craft the audio adversarial examples using multiple approaches [5, 20, 35] to target different ASR models (e.g., Deepspeech [21], Lingvo [9]). We also assume that generating each audio adversarial example can be accomplished with a limited amount of computational resources. We believe this is a sensible assumption because if significant time and computational resources were required for each adversarial example, the attack would become impractical and inefficient.

Defender's Capability: To defend against the audio adversarial attack, we assume the defender can obtain the adversarial examples before it proceeds to the ASR system. Unlike prior defenses that require plenty of adversarial samples for adversarial training [16, 30], we assume the defender does not have knowledge of the audio attacks, neither possess many attack samples. Besides, the defender does not obtain the benign transcription dataset and the attack target. Moreover, the defender does not know the protected ASR systems, including the model architecture, the parameters, and the input and output of the ASR systems.

Defense Scenarios: WavePurifier is a generalized purification framework that can be adopted on smart speakers. For all the incoming audio samples, WavePurifier generates the purified audio. For benign commands, WavePurifier can retain the same audio quality and the benign transcription. For the adversarial audios, WavePurifier can disrupt the target label and recover the benign transcription. Finally, the secure usage of smart speakers can be guaranteed.

3 SYSTEM DESIGN

3.1 WavePurifier Pipeline

Our purification framework is composed of three major steps: prepare, purify, and rebuild. In the initial phase, we obtain a pure speech dataset from available sources and fine-tune a pre-trained diffusion model for generating audio. Next, in the second stage, we purify the waveform by processing the chunked spectrogram of the audio with the fine-tuned diffusion model. The forward diffusion adds random noises to the input, resulting in a noisy spectrogram $x_a(t^*)$, followed by the reverse diffusion to recover the benign semantics of the noisy spectrogram. Since the diffusion model is trained by clean audio samples, intuitively, it will only generate clean audio samples. By leveraging this property, the recovered and clean spectrograms $x_a(\hat{t^*})$ are produced. For the rebuild stage, we concatenate the recovered spectrograms together to form a complete spectrogram of the audio. Then, combined with the original phase information, a clean audio waveform is created. To further eliminate the residual noises introduced by the forward diffusion, we add an extra denoise function in the audio reconstruction module. Finally, WavePurifier outputs the purified audio to complete the whole process.

3.2 Prepare the Purification

Diffusion model training: As described in Section 2.2, the diffusion model generates samples by reversing a gradual noising process [10]. More specifically, it starts from noise $x(T)$ and gradually removes the noise to produce less-noisy samples $x(T-1), x(T-2)$ until $x(0)$. Each timestep t is a certain noise level, and $x(t)$ is a combination of $x(0)$ with noise ϵ_t. The goal of the training diffusion model is to learn probability p_θ with a noisy $x(t)$. From Eq. 4, the mean $\mu_\theta(x(t), t)$ and the variance $\Sigma_\theta(x(t), t))$ are the parameters to learn. To train this model, we provide a clean audio spectrogram $x(0)$, a timestep t, and noise ϵ to forward diffuse it to a noisy sample $x(t)$ in the following equation:

$$q(x(t)|x(0)) = \sqrt{\bar{\alpha}_t}x(0) + \epsilon\sqrt{1-\bar{\alpha}_t}, \epsilon \sim \mathcal{N}(0, I) \quad (5)$$

Where $\alpha_t = 1 - \beta_t$ and $\bar{\alpha_t} = \Pi_{s=0}^{t}\alpha_s$. Once the noisy sample $x(t)$ is obtained, the diffusion model needs to learn the term $u_\theta(x(t), t)$ and $\Sigma_\theta(x(t), t))$ for the reverse process. Prior studies [10, 22] prove that directly parameterizing $\mu_\theta(x(t), t)$ is inefficient, instead, they train a model $\epsilon_\theta(x(t), t)$ to predict the ϵ from Eq. 5. Basically, the essential idea is that if the diffusion model can estimate the forward diffusion noise at every step, then it is capable of performing the reverse noise reduction. Formally, the loss function to train the $\epsilon_\theta(x(t), t)$ is listed as follows:

$$L_{\text{simple}} = E_{t\sim[1,T]}||\epsilon - \epsilon_\theta(x(t), t)||^2 \quad (6)$$

Where E is the expectation for the L_2 distance between real noise and the estimated noise on all steps. Once the L_{simple} is converged, the model $\epsilon_\theta(x(t), t)$ can be used to derive the $\mu_\theta(x(t), t)$:

$$\mu_\theta(x(t), t) = \frac{1}{\sqrt{\alpha(t)}}(x(t) - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon_\theta(x(t), t)) \quad (7)$$

As for the other critical factor $\Sigma_\theta(x(t), t))$, prior researches [10, 22] find that it can be fixed into a constant. Now that the mean and the variance have been derived, the diffusion model can be used to denoise the samples from the previous step, as illustrated in Eq. 4, furthermore, it can generate clean audio spectrograms $x(0)$ from pure noise distribution $x(T)$.

Audio data preparation: Since the diffusion model is widely used to produce high-quality images, we transfer the audio data to the image to fit the input requirements. For a raw audio waveform $v \in [-1, 1]^N$, we run the STFT(Short Time Fourier Transform) to convert the waveform to a spectrogram $S \in \mathbb{R}^{T\times F}$ and a phase $P \in \mathbb{R}^{T\times F}$. The STFT is described as follows:

$$S(m, \omega) = |\sum_{n=0}^{N} v[n]w[n-m]e^{-j\omega n}|, \quad (8)$$

Where m is the time index, ω is the frequency index, N is the waveform length and w is the window for convolution

operation. The size of the spectrogram depends on the duration of v, the STFT window length, STFT hop length, and the number of FFT points. Since existing open-source pretrained diffusion models only support the square input (e.g., 64*64, 128*128, 256*256), we have to split the complete spectrogram into a square shape, resulting in a series of chunk spectrograms. In our design, if the diffusion model requires an input shape as $M * M$, we will run the STFT with a $2 * M$ points FFT, resulting in a spectrogram shape as $T * M$. If the T is shorter than M, then we will pad it with zeros. Otherwise, we divide the spectrogram through time dimension by M, which generates T/M chunk spectrograms, with each having shape as $M * M$. Note that the last spectrogram chunk will retain the size of $M * M$ by borrowing the previous chunk.

3.3 Theory of Purification

The goal of WavePurifier is to purify audio adversarial examples to recover the benign commands. To achieve this goal, there are two factors to be considered. First, the perturbation of the audio adversarial example needs to be removed, consequently, the adversary's target will not appear in the ASR predictions. Second, the benign speech needs to be retained, indicating the benign command needs to be correctly recognized. Motivated by the forward and reverse process of the diffusion model, we assume the forward process can remove the adversarial perturbations and the reverse process can recover the benign speech. Based on these assumptions, we propose the first research questions:

RQ1: Why can forward diffusion remove perturbations?

To answer the research question, we define some notations. For the adversarial spectrogram, we denote it as $x_a(0)$, and its corresponding benign spectrogram as $x(0)$. For the forward process, it gradually adds noises to form different noisy outputs. We use $x_a(t)$ to represent the adversarial spectrogram and add t steps noise, also, we use $x(t)$ to denote the noisy benign spectrogram at step t. To measure the distance between $x_a(t)$ and $x(t)$ with increasing t, we can find whether forward diffusion can remove perturbations. Let y_t denote the distance of them as the following equation:

$$y_t = ||x_a(t) - x(t)|| \tag{9}$$

To understand how y_t changes with increasing t, we can refer to Theorem 3.1 proposed by prior study [31], which is formulated as follows:

$$\frac{\partial D_{KL}(x_a(t)||x(t))}{\partial t} \leq 0 \tag{10}$$

According to their finding, the KL divergence of the adversarial $x_a(t)$ and benign $x(t)$ monotonically decreases when increasing t. This is because the partial derivative of KL divergence with respect to t is less than or equal to zero, meaning that as t increases, $D_{KL}(x_a(t)||x(t))$ decreases, indicating a reduction in the difference between $x_a(t)$ and $x(t)$.

However, the previous finding only indicates the KL divergence, not explaining the value of y_t. To explicitly answer **RQ1**, we give the following proof: From Eq. 5 and replace $\bar{\alpha}_t$ to $\alpha(t)$, we can represent $x(t)$ as follows:

$$x(t) = \sqrt{\alpha(t)}x(0) + \sqrt{1-\alpha(t)}\epsilon \tag{11}$$

Specifically, when applying the VP-SDE [38] solver, the $\alpha(t)$ can be represented as $\alpha(t) = e^{-\int_0^t \beta(s)ds}$, where $\beta(t)$ represents a time-dependent noise scale. Next, we assume the $x_a(0) = x(0) + p$, where p is the adversarial perturbation, then the adversarial audio after forward diffusion can be represented as:

$$x_a(t) = \sqrt{\alpha(t)}(x(0) + p) + \sqrt{1-\alpha(t)}\epsilon \tag{12}$$

By adopting Eq. 12 and Eq. 11 into y_t, we can derive the following result:

$$\begin{aligned} y_t &= ||\sqrt{\alpha(t)}(x(0) + p) + \sqrt{1-\alpha(t)}\epsilon - \\ &\quad (\sqrt{\alpha(t)}x(0) + \sqrt{1-\alpha(t)}\epsilon)|| \\ &= ||\sqrt{\alpha(t)}p|| \end{aligned} \tag{13}$$

According to this equation, the y_t is only affected by $\alpha(t)$ and the perturbation p. Knowing that $\alpha(t) = e^{-\int_0^t \beta(s)ds}$, the greater t results greater integral value $\int_0^t \beta(s)ds$, and lead to smaller $\alpha(t)$ because the larger negative variable $-\int_0^t \beta(s)ds$ in exponential function. When t is infinitely large, $\alpha = e^{-\infty} = 0$, and y_t equals zero, which implies the adversarial sample and the benign sample have the same value after forward diffusion.

Observation 1: Forward diffusion can narrow down the difference between adversarial and benign samples, and this is affected by the magnitude of the perturbation p and the diffusion step t. If the diffusion step is infinitely large, in the best-case scenario, adversarial samples would be identical to benign ones.

According to Observation 1, we conclude that forward diffusion can remove perturbations. Next, we propose the second research question:

RQ2: Can the reverse diffusion recover the benign speech?

To answer this question, we define $\hat{x(t)}$ as the recovered benign spectrogram since step t, and the $\hat{x_a(t)}$ is the recovered adversarial spectrogram from forward diffusion at step t. To investigate the recovery quality for the adversarial sample, we can measure the distance between $\hat{x_a(t)}$ and the benign sample $x(0)$. Based on the Theorem 3.2 of the previous study [31], the distance can be represented as follows:

$$||\hat{x_a(t)} - x(0)|| \leq ||p|| + \sqrt{e^{2\gamma(t)} - 1}C_\delta + \gamma(t)C_s \tag{14}$$

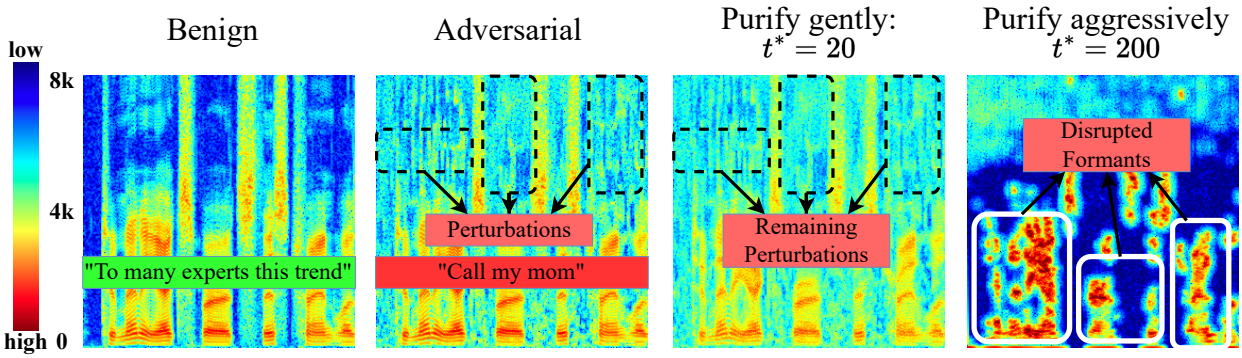


Figure 3: Demonstration of WavePurifier.

Where C_s and C_δ are positive constant, and $\gamma(t) = \int_0^t \frac{1}{2}\beta(s)ds$. In their finding, the distance between the recovered adversarial sample and the original benign sample are bound by three factors: the perturbation p, the term $\sqrt{e^{2\gamma(t)} - 1}C_\delta$ and the $\gamma(t)C_s$. As $\gamma(t)$ increases with increasing forward step t, the second and third terms will increase consequently. Therefore, the upper bound of $||\hat{x_a}(t) - x(0)||$ is getting higher, resulting in a larger distance between the recovered spectrogram and the original one.

Observation 2: The forward diffusion strategy favors using a larger value of t to eliminate the perturbation, whereas the reverse diffusion approach opts for a smaller value of t to ensure the quality of the recovered audio. This creates a conflict when selecting the diffusion step size t for both tasks. Given the conflicting perspectives on selecting the diffusion step size, we hypothesize that a trade-off option can be identified to achieve a balance between the two objectives. Building on this premise, we propose a third research question:

RQ3: Does an optimal purify step exist?

First, we visualize the aforementioned conflict in choosing the diffuse step. In Fig. 3, we take audio that was transcribed as "To many experts, this trend is inevitable" as benign audio, and craft an audio adversarial examples targets to "call my mom". To observe the perturbation clearly, we adjust the perturbation distortion tolerance to make the generated perturbation apparent. From the first figure, we find the benign speech is clear and the noise energy is low in some high-frequency bands. After adding the perturbations, the adversarial sample shows extra energy (labeled as perturbations) on the spectrogram. By using a small forward diffusion step t followed by a reverse step, we obtain a spectrogram that has been gently purified but still retains some perturbations. This indicates that the purification process has not been successful in completely removing the perturbations. On the other hand, when a larger t is used for more aggressive purification, most of the noise is removed, as shown in the last figure. However, this approach also affects the low-frequency speech details, leading to disruptions in the formants. Thus, it becomes evident that a random choice of the diffusion step can result in either incomplete removal of perturbations or compromise of the benign speech components. To find such optimal purification step t, we formulate the following objective function:

$$minimize(||x_a(t) - x(t)|| + ||\hat{x_a}(t) - x(0)||) \tag{15}$$

The objective of this function is to determine the optimal value of t that can effectively eliminate the perturbations in the adversarial sample while maintaining a close similarity to the original benign sample. Let y represent our objective goal, combine with Eq. 13 and Eq. 14, we have the following equation:

$$\begin{aligned} y &= ||x_a(t) - x(t)|| + ||\hat{x_a}(t) - x(0)|| \\ &\leq \sqrt{\alpha(t)}p + ||p|| + \sqrt{e^{2\gamma(t)} - 1}C_\delta + \gamma(t)C_s \end{aligned} \tag{16}$$

Next, we compute the partial derivation $\partial y/\partial t$ to search for the optimal t, to simplify the computation complexity, we replace the $\leq$ with $=$ for minimizing the upper bound of y.

$$\frac{\partial y}{\partial t} = \frac{\partial}{\partial t}\sqrt{\alpha(t)}p + \frac{\partial}{\partial t}||p|| + \frac{\partial}{\partial t}\sqrt{e^{2\gamma(t)} - 1}C_\delta + \frac{\partial}{\partial t}\gamma(t)C_s \tag{17}$$

For the first term of the above partial derivation:

$$\frac{\partial}{\partial t}\sqrt{\alpha(t)}p = \frac{1}{2}p\alpha(t)^{-\frac{1}{2}}\alpha(t) * -1\frac{\partial}{\partial t}\int_0^t \beta(s)ds \tag{18}$$

From the Leibniz integral rule [25],

$$\begin{aligned} \frac{\partial}{\partial t}\int_0^t \beta(s)ds &= \beta(t)\frac{dt}{dt} - \beta(0)\frac{d0}{dt} + \int_0^t \frac{\partial\beta(s)}{\partial t}ds \\ &= \beta(t) - 0 - 0 \\ &= \beta(t) \end{aligned} \tag{19}$$

So Eq. 18 can be rewritten as:

$$\frac{\partial}{\partial t}\sqrt{\alpha(t)}p = -\frac{1}{2}p\alpha(t)^{\frac{1}{2}}\beta(t) \tag{20}$$

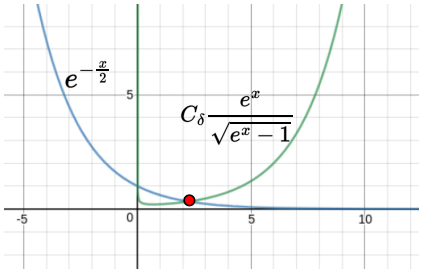


Figure 4: Intersection of the two functions

As for the second term in Eq. 17:

$$\begin{aligned}\frac{\partial}{\partial t}\sqrt{e^{2\gamma(t)}-1}C_\delta &= \frac{C_\delta}{2}(e^{2\gamma(t)}-1)^{-\frac{1}{2}}e^{2\gamma(t)}\frac{\partial}{\partial t}2\gamma(t)\\ &= \frac{C_\delta}{2}(e^{2\gamma(t)}-1)^{-\frac{1}{2}}e^{2\gamma(t)}\beta(t)\end{aligned} \tag{21}$$

Next, we do the partial derivation for the third term in Eq. 17:

$$\frac{\partial}{\partial t}\gamma(t)C_s = \frac{C_s}{2}\beta(t) \tag{22}$$

After simplifying the three terms, we let the $\partial y/\partial t = 0$. Then the Eq. 17 can be rewritten as follows:

$$\begin{aligned}&\frac{1}{2}p\alpha(t)^{\frac{1}{2}}\beta(t) = \frac{C_\delta}{2}(e^{2\gamma(t)}-1)^{-\frac{1}{2}}e^{2\gamma(t)}\beta(t)+\frac{C_s}{2}\beta(t)\\ &p\alpha(t)^{\frac{1}{2}} = C_\delta(e^{2\gamma(t)}-1)^{-\frac{1}{2}}e^{2\gamma(t)}+C_s\end{aligned} \tag{23}$$

Since $\alpha(t) = e^{-\int_0^t \beta(s)ds}$ and $e^{2\gamma(t)} = e^{\int_0^t \beta(s)ds}$, we denote $\int_0^t \beta(s)ds = x$, and the equation is transformed to:

$$pe^{-\frac{x}{2}} = C_\delta\frac{e^x}{\sqrt{e^x-1}}+C_s \tag{24}$$

According to the above equation, if we find the x that satisfies left and right equal to each other, we can find the optimal diffusion step t to minimize the objective score y. In Fig. 4, we draw two lines of the function $y = e^{-\frac{x}{2}}$ and $y = 0.1 * \frac{e^x}{\sqrt{e^x-1}}$. We found that these two lines have one intersection point (the red circle). This intersection proves that there exists a t that makes the derivation $\partial y/\partial t = 0$. In that case, the objective goal will be achieved by correctly selecting the diffusion step.

Observation 3: There exist and only exist one t to achieve minimal y for $t \in [0, +\infty)$. The value of the optimal t is determined by the $x = \int_0^t \beta(s)ds$.

3.4 Hierarchical Purification

In the previous subsection, we formulate an objective function to optimize the purification goal and prove the existence of the optimal diffusion step t^*. However, the objective function might work well on image purification (DiffPure [31]) and single-word speech purification (AudioPure[47]), but its performance on the long audio is questionable due to two critical reasons: (1) The image classification/speech classification model treats every pixel in the image equally, instead, the ASR model focuses more on the low-frequency formants, and neglect the importance of the high frequency; (2) The image adversarial examples/single word audio adversarial example typically add perturbation with a global view, in contrast, the long audio adversarial perturbation is crafted on the waveform vector instead of the 2D image, which is not considered the distribution on its spectrogram, therefore causing the perturbation assigned differently at a different frequency.

Therefore, we propose the hierarchical diffusion framework. In this model, we aim to purify the spectrogram diversely in terms of different frequency bands. In our design, we do not assign a global diffusion step t for the complete spectrogram, instead, we design a tuple of t to represent the diffusion step at different spectrogram bands. Namely, the $t^* = (t_s, t_m, t_l)$.

Where t_l denotes the diffusion step for the 4k-8k spectrogram, t_m means the diffusion step for 2k-4k spectrogram, and t_s is the diffusion step for 0-2k spectrogram. By optimizing the three t individually, we can achieve a fine-tuned purified performance. Consequently, the $x(\hat{t^*})$ can be represented by the concatenation of the three purified results:

$$x(\hat{t^*}) = x(\hat{t_s})_{F<2k}||x(\hat{t_m})_{2k<F<4k}||x(\hat{t_l})_{F>4k} \tag{25}$$

Fig. 5 illustrates the hierarchical purification framework. Giving adversarial audio, we first process the audio waveform to multiple fixed-size spectrograms (described in Section 3.2). Next, for every spectrogram, we perform the forward diffusion process three times, at each run t_l, t_m, t_s steps. Followed by the reverse process that starts at the t_l, t_m, t_s, resulting in three recovered spectrograms. For the rebuild stage, we collect the corresponding components of the three recovered spectrograms to form the hierarchical purified spectrogram. Combined with the original phase data, we transform the spectrogram into audio. In the end, we apply an extra denoise algorithm to remove the residual noise introduced by the diffusion process and generate the purified audio, which can be correctly recognized by the ASR model.

RQ4: How to determine the hierarchical purify step?

To find the optimal purification step for different frequencies, we use a data-driven method. First, we let t_s, t_m, t_l share the same value t_{sub}. By iterating the t_{sub} and calculating the smallest y, we find the best t_sub that achieves the optimal purification goal. Next, we initialize $t_s, t_m, t_l = t_{sub}$, and start to search the optimal purification steps for each frequency. In practice, the optimize goal $y = ||x_a(t)-x(t)||+||x_a\hat{(t)}-x(0)||$ is meaningless in a speech recognition task, we revise it to

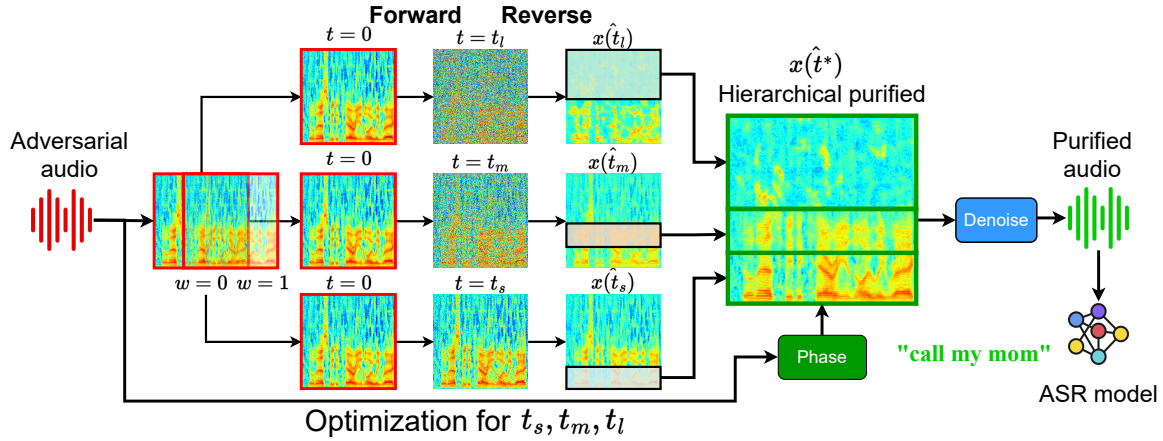


Figure 5: WavePurifier system architecture.

the following format:

$$y = CER(f(x_a(\hat{t})), benign) - k * CER(f(x_a(t)), target) \quad (26)$$

We use $f(\cdot)$ to denote the ASR model, and $f(x)$ is the recognized transcription for spectrogram x. k is a constant to control the weights of those two terms. Since our ultimate goal is correctly recognizing the benign commands, and discarding the target commands, we replace the L_2 distance of the spectrogram with the CER(character error rate) metric. In Section 3.3, we prove the existence of the t for optimizing the prior objective function, however, the t value cannot be derived because the noise is random. For the new objective function, we assume the ASR model can be queried, and the output of the ASR model is available. Once the optimal t is determined, WavePurifier can be adopted on different ASR models without knowledge of those models. To compute the minimal y, we iterate the t to compute the CERs. We denote t^* as the global optimal purify step. Next, we start the fine-tuned search by assigning t^* as the start point of t_s, t_m, and t_l. In conclusion, by repeatedly executing the steps that are close to t^* for the hierarchical values of t_s, t_m, and t_l, we can determine the most effective hierarchical purification step. The optimized purifying steps are then allocated to WavePurifier to safeguard against unforeseen adversarial attacks.

4 EVALUATION

4.1 Experiment Settings

4.1.1 Target ASR models. We choose two target ASR models to evaluate our defense. The first one is the Mozilla DeepSpeech [21] model. This model is composed of multiple layers of recurrent cells and handles the sequential audio data. It takes audio samples as input and recognizes them as sentences with the CTC decoder. More specifically, we use DeepSpeech version 0.4.1 as the victim model. The second ASR model is Google Lingvo [36], which is powered by the Tensorflow framework for building sequence models. We set up this model via docker from a GitHub repository [9].

4.1.2 Audio adversarial attacks. We select three recent audio adversarial attacks on the ASR model. We reproduce all the audio adversarial attacks based on their description with different languages and ASR frameworks. The attacks are listed as follows:

- **C&W:** This attack was introduced by Carlini and Wagner [5] in 2018. They proposed this attack against the DeepSpeech model. This attack assumes the adversary has complete knowledge of the victim model, then he/she crafts the perturbation by optimizing the CTC loss between the target transcription and the model output.
- **Qin-I:** This is a white-box imperceptible audio attack against the Lingvo model that was proposed by [35] in 2019. Compared to the prior C&W attack, this attack leverages the acoustic feature to formulate an objective function, therefore constraining the perceptual level of the adversarial perturbation.
- **SpecPatch:** This is a white-box audio adversarial attack proposed in 2022 [20]. The authors craft a spectrogram patch to attack the DeepSpeech model, then they demonstrate this attack under the Human-in-the-loop scenario. This attack is proven to be robust to different sentence contexts and environments.

4.1.3 Audio adversarial defenses. We compare our defense with seven defense mechanisms, including three defenses from WaveGuard [23], two defenses from the conventional denoise methods, and two diffusion-based methods from AudioPure [47]. The attacks are listed as follows:

- **Down-Up sampling (Ds)**: This method uses a resampling process to defend against audio AE.
- **Linear Predictive Coding (LPC)**: This method changes the linear filter of the waveform to reconstruct audio, therefore defending against audio AE.
- **Quantization-Dequantization (Quant.)**: This method transforms the quantization setting of the provided audio to generate new audio. The first three defenses are used in WaveGuard [23].
- **Stationary Noise Reduction (SNR)**: This is a noise reduction algorithm that suppresses the noise by a pre-defined noise threshold.
- **Adaptive Noise Reduction (ANR)**: Different from SNR, ANR Continuously updates the estimated noise threshold over time.
- **DiffSpec**: This method is proposed by AudioPure [47]. It purifies the single **mel-spectrogram** by image diffusion model.
- **Diff Wave**: This method is also proposed by AudioPure, which purifies the waveform by using the diffwave model [24]. As they assume all the input speech samples have a fixed length, we separate our long audio into multiple chunks and reproduce their work by feeding the chunk audio data.

4.1.4 Attack and defense settings. **Dataset:** Different from AudioPure [47] which focuses on command classification tasks (recognizing single words) on a short command dataset, we use TIMIT dataset that includes 6,300 audios, and each audio is around 5 seconds. To simulate the attack, we choose the target sentence from ok-google.io, which provides commonly used commands on Google Assistant. To construct the audio adversarial example set, we randomly select 100 pairs of benign audio and target sentences and produce 300 audio adversarial examples.

Attack performance: We achieve 100% attack success rate for C&W attack. For the Qin-I attack, we have a 96% attack success rate. In the SpecPatch attack, we get an 87% success rate.

Diffusion model settings: We choose the pre-trained guided diffusion [10] model as our diffusion model. More specifically, we download checkpoints of 64×64, 128×128, 256×256 models, and fine-tune them with the corresponding shape of the spectrogram. The performance of diffusion models that fine-tuned different shapes is described in Section 4.

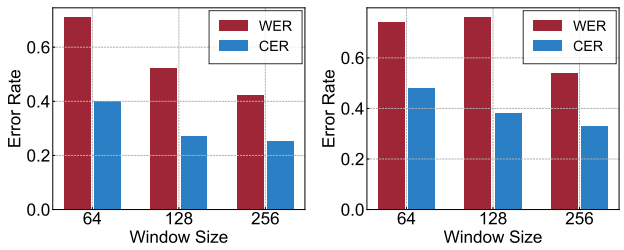
Experiment platform: All the experiments are conducted on a desktop with Intel i7-7700k CPUs, 64GB RAM, and NVIDIA 1080Ti GPU, running 64-bit Ubuntu 18.04 LTS operating system. For the C&W attack and SpecPatch attack, it uses Python 3.6 Tensorflow 1.14. The Qin-I attack uses Python 2.7, Tensorflow 1.13.1. The guided diffusion model uses Python 3.6 and Pytorch 1.10.1. The detailed setups can be found in [9, 34].

Metrics: We use Character Error Rate (CER), Word Error Rate (WER), and Purify Success Rate (PSR) to measure the performance. **CER**: This metric denotes the character error rate. Specifically, we use $CER(F, T)$ to denote the CER between **F**orward diffused transcription and the **T**arget transcription, and $CER(P, B)$ represents the CER between **P**urified transcription and the **B**enign transcription. Based on Eq. 26, we expect a low $CER(P, B)$ and high $CER(F, T)$ because our goal is to recover the benign transcription and disrupt the malicious transcription. **WER**: This metric evaluates the word-level error rate between the purified transcription and the benign transcript. **PSR**: We define the Purification Success Rate (PSR) as the rate at which purification is successful. We consider purification successful if the $CER(P, B)$ is below a specified threshold. Based on our experience with speech recognition models, we set this threshold at 0.25. Typically, an effective model has a CER between 0.05 and 0.2 (e.g., OpenAI's Whisper [39]). If WavePurifier achieves a CER around 0.25, once applying Text Auto-correction, which provides a $\tilde{2}5\%$ reduction in CER [26], can lower the overall CER below 0.2, which is acceptable for current speech recognition standards.

4.2 Choosing the Purify Granularity

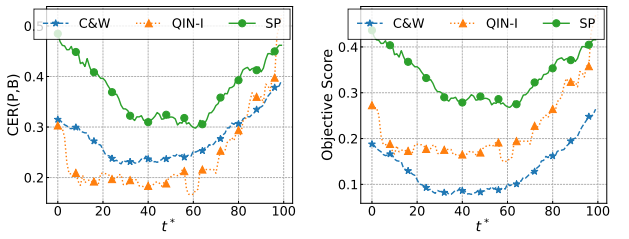
Before conducting the purification, we need to determine the window size of the diffusion model. Based on the pre-trained diffusion model, it contains different generating image shapes such as 64×64, 128×128, 256×256. By choosing different diffusion models, we can control the purification granularity. For small granularity, the purification model is fast but loses global vision to generate the complete spectrogram. However, for large granularity, the diffusion model might neglect the details in the generated spectrogram. Therefore, the first evaluation focus is on selecting the purification granularity and the diffusion model.

We purified the spectrograms with 50 steps and 80 steps, then re-constructed the purified spectrograms to form the purified audio. We calculate the WER and CER based on the transcription of purified audio and benign transcript. According to Fig. 6, we observe the WER and CER experience a steady drop with increasing window size. The results show consistent trends for different diffusion steps $t = 50$ (Fig. 6(a)) and $t = 80$ (Fig. 6(b)), which indicates that the global semantic information is more important than the details in the spectrogram. Besides, we find that the CER is 0.25 when choosing the 256×256 diffusion model, which proves the potential of adopting the diffusion model to purify audios. Based on this observation, we decide to use 256×256 granularity to run the rest evaluations.



(a) Purify C&W attack with t=50 (b) Purify C&W attack with t=80

Figure 6: Purify with different granularity



(a) Reverse diffuse to recover benign (b) Objective scores VS purify steps

Figure 7: Global optimization for purifying steps

4.3 Searching the Purify Steps

Global optimal searching: The global optimal searching follows the design in DiffSpec [47]. As the prior work did not evaluate the performance on practical audio adversarial attacks, we test this defense on three attacks (C&W [5], QIN-I [35], and SpecPatch [20]) with 100 steps. For each t, we perform the complete forward and reverse process. In total, we generate 78,864 purified spectrograms.

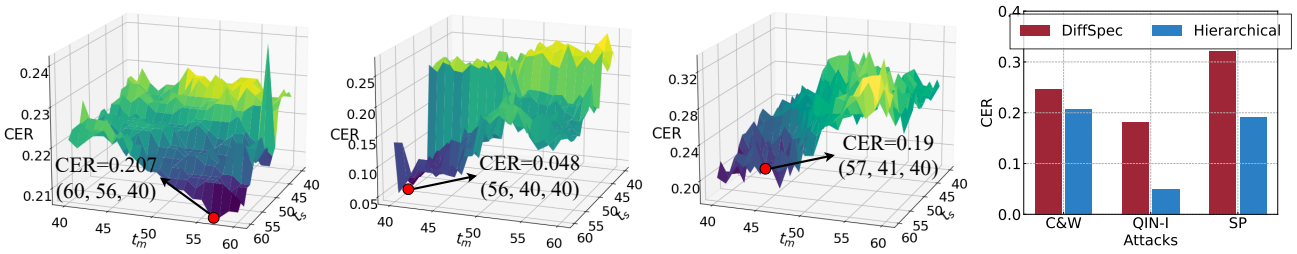
Fig. 7(a) demonstrates that all three attacks have similar trends. The CER decreases before $t < 40$, remains steady between $40 < t < 60$, and increases gradually after $t > 60$. It is surprising to find that different audio adversarial attacks experience similar CER changes for diffusion purification. Observing the CER of the target and benign transcript, we combine them together to visualize the objective score in Eq. 26. We choose $k = 0.1$ since our primary goal is to recover the benign commands. The average objective score for using different t to purify is present in Fig. 7(b). In this result, the optimal global t is uncovered. While $t = 40$, the objective function achieves the lowest point for all three attacks.

Hierarchical optimal searching: After obtaining the global optimal diffusion step, we assign the t_l equals to 40 and fix it to fine-tune the ts and tm with the range from 40 to 60. For every AE in each attack, we purify it $20 * 20 = 400$ times and compute the average $CER(P, B)$. We plot the 3D surface figure to demonstrate the CER change along with different t_s and t_m in Fig. 8. For C&W attack in Fig. 8(a), the optimized CER reaches to 0.207 with $t_s = 60$, $t_m = 56$, and $t_l = 40$. In comparison, the purification of QIN-I attack achieves the best performance, with the lowest CER 0.048 when $(t_s, t_m, t_l) = (56, 40, 40)$. Similar to the QIN-I attack's result, the hierarchical purification on the SpecPatch attack also prefers the large diffusion step on low frequency while keeping the small diffusion step on a high frequency. By conducting the hierarchical optimal searching, we find that the low-frequency data need to be purified intensively, while the middle and high-frequency spectrogram are less important, so they are encouraged to purify gently. To further visualize the advantage brought by the hierarchical diffusion strategy, we compare the performance change between the DiffSpec (global purify) and the hierarchical purify. In Fig. 8(d), we find that our hierarchical scheme improves the purify performance on all three attacks, achieving less CER compared to the global purification.

4.4 Overall Performance

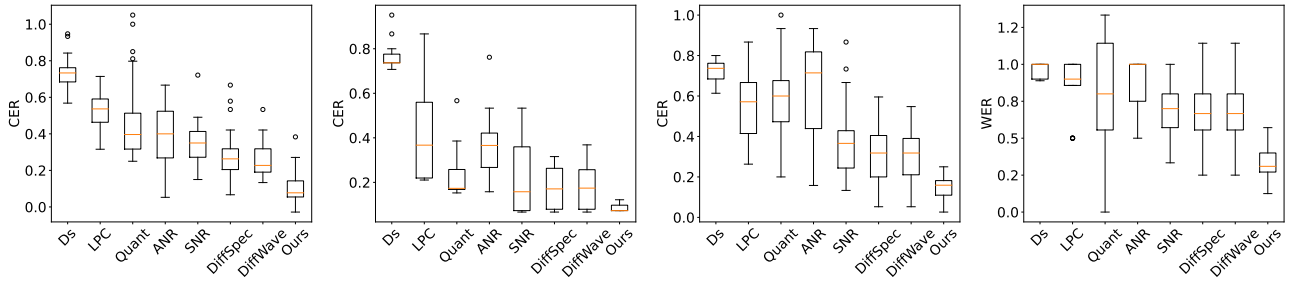
As noted earlier, different attacks require different optimal settings for effective defense. To create a more generalized defense solution, we compromise on some attack-specific optimizations to apply uniform purification parameters across all attacks. Specifically, we treat WavePurifier like a black-box purification framework, applying the same hierarchical settings $(t_s, t_m, t_l) = (56, 40, 40)$ to all incoming audio. This configuration was chosen because it achieves the best average CER, as shown in Fig. 8. Unless otherwise specified, the following experiments use this generalized setting, assuming the defender has no prior knowledge of the specific attacks.

Comparison with other approaches: We compare our work with seven defenses from WaveGuard [23] and AudioPure [47]. We feed AEs to the seven defenses and collect the output audio. Then, we use the ASR model to recognize the transformed audio and compute the CER. Meanwhile, we purify those AEs with WavePurifier, and get the CER result for our methods. The result is presented in Fig. 9. From Fig. 9(a), we realize that our CER distribution is further dropped after adding the noise reduction algorithm, achieving an average 0.1 CER for the C&W attack. For comparison, the signal processing methods (Ds, LPC, Quant, SNR, ANR) are not performing well for transcription recovery. The prior diffusion purification work [47] DiffSpec and DiffWave show potential to purify speech, but WavePurifier, which benefits from the hierarchical design, outperforms them with lower CER. As for purifying the QIN-I attack shown in Fig. 9(b), our method outperforms all existing studies, with the CER distributed closely at 0.05. For the SpecPatch attack, all the defense methods receive a high CER due to the robustness of the adversarial patch. However, WavePurifier only has slight performance degradation, with average CER



(a) Hierarchically purify CW attack (b) Hierarchically purify QIN-I attack (c) Hierarchically purify SP attack (d) DiffSpec VS Hierarchy purify

Figure 8: Hierarchical optimization for purifying steps



(a) CER of purifying C&W attack (b) CER of purifying QIN-I attack (c) CER of purifying SpecPatch attack (d) WER of purifying SpecPatch attack

Figure 9: Comparisons with other purification methods.

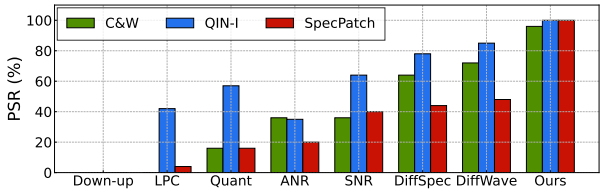


Figure 10: Purification success rate

as 0.2. In the end, we evaluate the WER of WavePurifier and the other methods to purify the SpecPatch attack and present the result in Fig. 9(d). Compared to the existing studies, WavePurifier achieves the best WER, with around 0.3. We also compare WavePurifier with other methods in the Purify Success Rate. By calculating the percentage of the successfully purified audio samples, we present the result in Fig. 10. It is evident that WavePurifier achieves the best purify success rate for all of the three attacks, with 98%, 100%, and 100% Purify Success Rate. In contrast, the other audio transformation and noise reduction algorithms at most reach to 64% success rate.

Is the CER improvement helpful for smart devices? As we observed that WavePurifier effectively purifies audio and achieves low CER/WER with speech recognition models, it raises the question of whether these improvements in CER/WER can enhance voice understanding in real-world scenarios, such as with voice assistants. To address this question, we find that voice assistants comprehend our speech not only through speech recognition models but also by utilizing NLP-based auto-correlation techniques. Therefore, they can infer the user's intention even when there are minor errors in the transcription [4]. While most commercial voice assistants do not release their NLP-based auto-correlation approaches(e.g., context-based corrections, spelling correction algorithms, or machine learning models), we implement three common correction approaches as add-on techniques to evaluate if WavePurifier helps to purify and understand voice commands in a real-world scenario.

Autocorrect [32] is a probabilistic model for spell correction. It utilizes a large corpus of text to build a frequency distribution of words and applies edits (insertions, deletions, substitutions, and transpositions) to generate possible corrections for a misspelled word. The correct word is chosen by maximizing the probability, combining the likelihood of the correction appearing in the corpus and the likelihood of the error occurring given the correction. Symspellpy [12] is another spelling correction approach that utilizes the Symmetric Delete Spelling Correction algorithm. Different from the traditional methods that require handling various types of edits (deletes, transposes, replaces, and inserts), Symspellpy only requires the number of deletes to pre-compute and store potential misspellings. This approach allows for incredibly fast spell correction, as it reduces the number of calculations. RapidFuzz [1] is a fast string matching library for Python

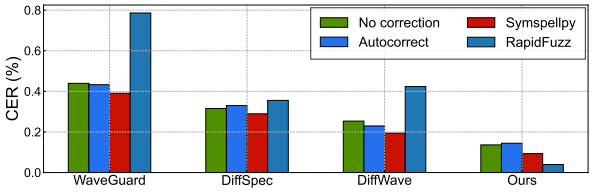


Figure 11: Post Auto-correction CERs

and C++, which uses string similarity calculations to find the most matched string from a pre-defined set. Different from the autocorrection tools [12, 32], the RapidFuzz can return "NOT RECOGNIZED" if the given transcription is not similar to any pre-defined voice commands.

In our setting, we apply different defenses (WaveGuard-lpc10, DiffSpec, DiffWave, WavePurifier) to defend AEs from C&W attacks. Next, we feed the defended audio to the speech recognition model (DeepSpeech) and collect the recognized transcriptions. Once we have the recognized transcriptions, we use three add-ons (Autocorrect, Symspellpy, and RapidFuzz) to correct them. Note that we construct a voice command set for RapidFuzz, which only contains common voice commands such as "open the door, close the window, play some music". To understand the impact of auto-correction on various audio defenses, we conducted a benchmark test by feeding 10 adversarial examples (derived from 2 benign voice commands and targeting 5 attack commands) into the audio defenses and collecting their transcriptions from a speech recognition model. We then processed the transcriptions under three aforementioned auto-corrections and calculated the average CER with benign voice commands. The results are shown in Fig. 11. We observe that spellcheck-based methods like Autocorrect and Symspellpy have difficulty correcting the transcription and achieve only minimal reduction in CERs. This may be due to: 1) the significant edit distance between the recognized transcription and the benign command; and 2) the low occurrence frequency of benign commands in the corpus. For the RapidFuzz, all existing audio defenses with RapidFuzz experience higher CERs. This issue arises from "NOT RECOGNIZED", where the transcription does not match any commands in the pre-defined set, and RapidFuzz returns null. In contrast, our approach achieves the lowest CER before auto-correction and could be further optimized with auto-corrections, and reach to CER as low as 5%. This experiment shows that the CER improvement in WavePurifier can help the smart devices purify the attack and correctly understand commands.

5 DISCUSSION

5.1 Time budgets

We test the average running time for the ASR model and our purification framework on two platforms: the desktop and the high-end server. The desktop is a commodity device and the server is running 8x RTX A6000 (48G) GPU. We report the time budgets in Table 1, with the left number showing the running time on a desktop and the right number displaying the running time on a server. Since we consider purifying a complete command in a real-world setting, the average duration of input audio is 5 seconds.

Time budgets on the desktop: In this setting, every query of the ASR model takes 4 to 9 seconds. In comparison, the forward and reverse diffusion process takes 8.77 seconds on average. In total, the WavePurifier completes the purification step in 8.9 seconds, which is comparable to the time cost on the ASR model.

Time budget on the server: With the same length of audio input, the server can process the ASR model and Diffusion model with x10 speed. Specifically, each audio can be recognized with 500ms and can be purified at a similar time. This result implies the possibility of deploying WavePurifier on the cloud server of voice assistant, for example, purifying the voice commands before recognizing it.

5.2 Understand the defense

WavePurifier leverages a diffusion model for speech purification, offering distinct advantages over other machine learning approaches like autoencoders and GANs. Unlike autoencoders, which often reduce the dimensionality of data, our diffusion model preserves the original data dimensions throughout the purification process. This ensures that the full complexity of the speech signal is maintained, allowing for the retention of benign components while effectively eliminating adversarial perturbations. In contrast to GANs, which generate data in a single pass, our step-by-step generation process provides finer control over purification, reducing the risk of losing important details. Additionally, the hierarchical nature of the diffusion process provides fine-grained control over purification depth, enabling the model to adaptively remove noise across different frequency bands. This approach ensures that essential speech features are retained while minimizing distortions. Moreover, the diffusion model's iterative refinement capability contributes to achieving a lower Character Error Rate (CER) and higher Purification Success Rate (PSR), as demonstrated in our experiments. The results suggest that WavePurifier is particularly well-suited to outperform other generative models, especially in challenging scenarios involving complex, real-world audio adversarial attacks.

5.3 Defending against adaptive attackers

Adaptive attackers who are knowledgeable of our defense can construct attacks that are robust to our defense by incorporating the gradients of our model into the attacking process. However, directly tracking the gradients in WavePurifier (256×256 input, 3 diffusion models) caused the Out-of-memory issue on our 1080Ti with 11GB GPU, a recent study [11]

also found a similar phenomenon when they retrieve the gradients from the diffusion models. Their experiment shows that it takes several hours to attack one batch of 8 CIFAR10 images on a high-end GPU(e.g., RTX 2080Ti with 11GB memory). In our case, Such significant computation cost does not align with our assumptions about the attacker's capabilities because we assume the attacker can generate AEs with a limited amount of computational resources, otherwise this attack is impractical. In fact, prior works [31, 47] prove the diffusion model can provide robustness with regard to the adaptive attacker. They use the adjoint method [27] to compute the full gradient of the reverse generative process and craft the adaptive attack samples. Their experiment shows that the diffusion-based defense achieves ~70% robustness accuracy. To summarize, the adaptive attacker of WavePurifier requires substantial computational effort for crafting the attack samples, and the diffusion-based defense models are potentially capable of defending against those attacks.

Process		Avg. Running Time (s)
ASR Models	DeepSpeech	9.04 / 0.56
	Lingvo	4.03 / 0.24
Purify Steps	Diffuse	8.77 / 0.44
	Reconstruct	0.11 / 0.12
	Denoise	0.02 / 0.02
Total Time of WavePurifier		8.9 / 0.58

Table 1: Time budget for all processes. Left: running on desktop; Right: running on GPU server

5.4 Purification in noisy environment

To further evaluate the performance of WavePurifier in different working scenarios, we conduct extensive experiments. Specifically, we take 10 Adversarial examples from the C&W attack and apply 4 different noises (Babble noise, Car noise, Gaussian noise, and Factory noise) from NOISEX-92 dataset [45]. For each AE, we add 4 different levels of noise, the SNR ranges from -10dB to 5 dB. Next, we feed the AEs with different noises to WavePurifier, and compute the PSR. The result is reported in Fig. 12. The results show that our method is effective in all noise types and SNR levels. Notably, our purification only fails to purify 1 of 10 AEs with -10dB SNR and limited noise type (Babble and Car noise). If the noise is not in extreme condition, WavePurifier can perform well and achieve similar performance compared with no noise attack. We believe the reason our method is effective in noisy environments is that it inherently includes multiple noise reduction modules, such as hierarchical denoising within the diffusion model and an additional denoising module applied after the diffusion model. This result implies the robustness of WavePurifier in different noise scenarios.

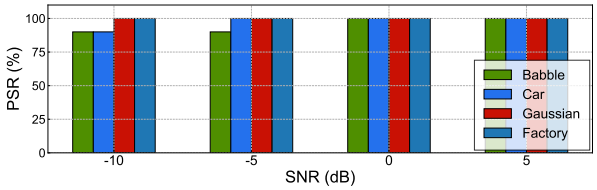


Figure 12: Purification in noisy environment

5.5 Limitation and Future work

Purifying physical adversarial attacks: While our current study demonstrates that WavePurifier is robust against various adversarial attacks, there are emerging attack categories such as ultrasound attacks [49] and backdoor attacks [18, 37, 46] that present unique challenges to audio purification frameworks. In future work, we plan to extend our design to include these more complex and less-studied attack vectors. We aim to validate further the effectiveness and adaptability of WavePurifier across an even wider array of adversarial scenarios. Additionally, we will explore enhancements to our purification methods that could address these new threats, making a robust defense solution.

Purifying multilingual audio adversarial attacks: While our current work evaluates our robustness against various adversarial attacks, we acknowledge the importance of assessing its performance across different languages and accents. Speech recognition systems often face significant challenges when dealing with linguistic diversity, which can impact the effectiveness of purification methods. In future work, we plan to extend our evaluation to include multiple languages and accents, ensuring that WavePurifier is capable of adapting to a wide range of linguistic variations. However, it is important to note that the current availability of adversarial attacks targeting non-English languages is limited, making it challenging to implement and test these scenarios comprehensively. As more diverse datasets and adversarial examples become available, we will integrate them into our framework to provide a more thorough analysis of WavePurifier's performance across different linguistic contexts.

6 CONCLUSION

We present WavePurifier, the first long audio adversarial attack purification framework. We demonstrate that WavePurifier can defend against highly distorted audio AE, achieving high purification success rate on multiple scenarios.

ACKNOWLEDGMENTS

We would like to extend our appreciation to our shepherd and the anonymous reviewers for their invaluable input on our study. This work was supported in part by the U.S. National Science Foundation grants CNS-2226888 and CNS-2310207.

REFERENCES

[1] 2023. RapidFuzz. https://pypi.org/project/rapidfuzz/.

[2] Moustafa Alzantot, Bharathan Balaji, and Mani Srivastava. 2017. Did you hear that? adversarial examples against automatic speech recognition. In *31st Conference on Neural Information Processing Systems (NIPS)*.

[3] Amazon. 2021. Amazon Echo. https://www.amazon.com/All-New-Echo-4th-Gen/dp/B07XKF5RM3.

[4] Françoise Beaufays. 2022. Ask a Techspert: How does Google Assistant understand your questions? https://blog.google/products/assistant/ask-a-techspert-assistant-questions/.

[5] Nicholas Carlini and David Wagner. 2018. Audio adversarial examples: Targeted attacks on speech-to-text. In *2018 IEEE Security and Privacy Workshops (SPW)*. IEEE, 1–7.

[6] Tao Chen, Longfei Shangguan, Zhenjiang Li, and Kyle Jamieson. 2020. Metamorph: Injecting inaudible commands into over-the-air voice controlled systems. In *Network and Distributed Systems Security (NDSS) Symposium*.

[7] Yuxuan Chen, Xuejing Yuan, Jiangshan Zhang, Yue Zhao, Shengzhi Zhang, Kai Chen, and XiaoFeng Wang. 2020. Devil's whisper: A general approach for physical adversarial attacks against commercial black-box speech recognition devices. In *29th USENIX Security Symposium (USENIX Security 20)*. 2667–2684.

[8] Moustapha M Cisse, Yossi Adi, Natalia Neverova, and Joseph Keshet. 2017. Houdini: Fooling deep structured visual and speech recognition models with adversarial examples. *Advances in neural information processing systems* 30 (2017).

[9] cleverhans. 2019. Lingvo. https://github.com/cleverhans-lab/cleverhans.

[10] Prafulla Dhariwal and Alexander Nichol. 2021. Diffusion models beat gans on image synthesis. *Advances in Neural Information Processing Systems* 34 (2021), 8780–8794.

[11] Yue Gao, Ilia Shumailov, Kassem Fawaz, and Nicolas Papernot. 2022. On the Limitations of Stochastic Pre-processing Defenses. *arXiv preprint arXiv:2206.09491* (2022).

[12] Wolf Garbe. 2022. SymSpell: 1 million times faster through Symmetric Delete spelling correction algorithm. https://github.com/wolfgarbe/SymSpell.

[13] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. 2014. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572* (2014).

[14] Google. 2021. Google Assistant. https://assistant.google.com/.

[15] Google. 2021. Google Home/Nest. https://store.google.com/product.

[16] Sven Gowal, Sylvestre-Alvise Rebuffi, Olivia Wiles, Florian Stimberg, Dan Andrei Calian, and Timothy A Mann. 2021. Improving robustness using generated data. *Advances in Neural Information Processing Systems* 34 (2021), 4218–4233.

[17] Chuan Guo, Mayank Rana, Moustapha Cisse, and Laurens Van Der Maaten. 2017. Countering adversarial images using input transformations. *arXiv preprint arXiv:1711.00117* (2017).

[18] Hanqing Guo, Xun Chen, Junfeng Guo, Li Xiao, and Qiben Yan. 2023. Masterkey: Practical backdoor attack against speaker verification systems. In *Proceedings of the 29th Annual International Conference on Mobile Computing and Networking*. 1–15.

[19] Hanqing Guo, Guangjing Wang, Yuanda Wang, Bocheng Chen, Qiben Yan, and Li Xiao. 2023. PhantomSound: Black-Box, Query-Efficient Audio Adversarial Attack via Split-Second Phoneme Injection. In *Proceedings of the 26th International Symposium on Research in Attacks, Intrusions and Defenses*. 366–380.

[20] Hanqing Guo, Yuanda Wang, Nikolay Ivanov, Li Xiao, and Qiben Yan. 2022. SpecPatch: Human-in-the-Loop Adversarial Audio Spectrogram Patch Attack on Speech Recognition. In *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*. 1353–1366.

[21] Awni Hannun, Carl Case, Jared Casper, Bryan Catanzaro, Greg Diamos, Erich Elsen, Ryan Prenger, Sanjeev Satheesh, Shubho Sengupta, Adam Coates, et al. 2014. Deep speech: Scaling up end-to-end speech recognition. *arXiv preprint arXiv:1412.5567* (2014).

[22] Jonathan Ho, Ajay Jain, and Pieter Abbeel. 2020. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems* 33 (2020), 6840–6851.

[23] Shehzeen Hussain, Paarth Neekhara, Shlomo Dubnov, Julian McAuley, and Farinaz Koushanfar. 2021. {WaveGuard}: Understanding and Mitigating Audio Adversarial Examples. In *30th USENIX Security Symposium (USENIX Security 21)*. 2273–2290.

[24] Zhifeng Kong, Wei Ping, Jiaji Huang, Kexin Zhao, and Bryan Catanzaro. 2020. Diffwave: A versatile diffusion model for audio synthesis. *arXiv preprint arXiv:2009.09761* (2020).

[25] Leibniz. 1700. Leibniz integral rule. https://en.wikipedia.org/wiki/Leibniz_integral_rule.

[26] Yichong Leng, Xu Tan, Wenjie Liu, Kaitao Song, Rui Wang, Xiang-Yang Li, Tao Qin, Ed Lin, and Tie-Yan Liu. 2023. Softcorrect: Error correction with soft detection for automatic speech recognition. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 37. 13034–13042.

[27] Xuechen Li, Ting-Kam Leonard Wong, Ricky TQ Chen, and David K Duvenaud. 2020. Scalable gradients and variational inference for stochastic differential equations. In *Symposium on Advances in Approximate Bayesian Inference*. PMLR, 1–28.

[28] Zhuohang Li, Yi Wu, Jian Liu, Yingying Chen, and Bo Yuan. 2020. AdvPulse: Universal, Synchronization-free, and Targeted Audio Adversarial Attacks via Subsecond Perturbations. In *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*. 1121–1134.

[29] Ji Lin, Chuang Gan, and Song Han. 2019. Defensive quantization: When efficiency meets robustness. *arXiv preprint arXiv:1904.08444* (2019).

[30] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. 2017. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083* (2017).

[31] Weili Nie, Brandon Guo, Yujia Huang, Chaowei Xiao, Arash Vahdat, and Anima Anandkumar. 2022. Diffusion Models for Adversarial Purification. *arXiv preprint arXiv:2205.07460* (2022).

[32] Peter Norvig. 2007. How to Write a Spelling Corrector. https://norvig.com/spell-correct.html.

[33] OpenAI. 2022. dall-e-2. https://openai.com/dall-e-2/.

[34] opne ai. 2019. guided-diffusion. https://github.com/openai/guided-diffusion.

[35] Yao Qin, Nicholas Carlini, Garrison Cottrell, Ian Goodfellow, and Colin Raffel. 2019. Imperceptible, robust, and targeted adversarial examples for automatic speech recognition. In *International conference on machine learning*. PMLR, 5231–5240.

[36] Jonathan Shen, Patrick Nguyen, Yonghui Wu, Zhifeng Chen, Mia X Chen, Ye Jia, Anjuli Kannan, Tara Sainath, Yuan Cao, Chung-Cheng Chiu, et al. 2019. Lingvo: a modular and scalable framework for sequence-to-sequence modeling. *arXiv preprint arXiv:1902.08295* (2019).

[37] Cong Shi, Tianfang Zhang, Zhuohang Li, Huy Phan, Tianming Zhao, Yan Wang, Jian Liu, Bo Yuan, and Yingying Chen. 2022. Audio-domain position-independent backdoor attack via unnoticeable triggers. In *Proceedings of the 28th Annual International Conference on Mobile Computing And Networking*. 583–595.

[38] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. 2020. Score-based generative

modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456* (2020).

[39] Speechly. 2023. Analyzing OpenAI's Whisper ASR Model's Word Error Rates Across Languages. https://www.speechly.com/blog/analyzing-open-ais-whisper-asr-models-word-error-rates-across-languages. Accessed: 2024-08-14.

[40] stability.ai. 2022. stable diffusion. https://stability.ai/blog/stable-diffusion-public-release.

[41] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. 2013. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199* (2013).

[42] Rohan Taori, Amog Kamsetty, Brenton Chu, and Nikita Vemuri. 2019. Targeted adversarial examples for black box audio systems. In *2019 IEEE Security and Privacy Workshops (SPW)*. IEEE, 15–20.

[43] Siri Team. 2017. Hey siri: An on-device dnn-powered voice trigger for apple's personal assistant. *Apple Machine Learning Journal* 1, 6 (2017).

[44] Arash Vahdat, Karsten Kreis, and Jan Kautz. 2021. Score-based generative modeling in latent space. *Advances in Neural Information Processing Systems* 34 (2021), 11287–11302.

[45] Andrew Varga and Herman JM Steeneken. 1993. Assessment for automatic speech recognition: II. NOISEX-92: A database and an experiment to study the effect of additive noise on speech recognition systems. *Speech communication* (1993).

[46] Yuanda Wang, Qiben Yan, Nikolay Ivanov, and Xun Chen. 2023. A Practical Survey on Emerging Threats from AI-driven Voice Attacks: How Vulnerable are Commercial Voice Control Systems? *arXiv preprint arXiv:2312.06010* (2023).

[47] Shutong Wu, Jiongxiao Wang, Wei Ping, Weili Nie, and Chaowei Xiao. 2023. Defending against adversarial audio via diffusion model. *arXiv preprint arXiv:2303.01507* (2023).

[48] Hiromu Yakura and Jun Sakuma. 2018. Robust audio adversarial example for a physical attack. *arXiv preprint arXiv:1810.11793* (2018).

[49] Qiben Yan, Kehai Liu, Qin Zhou, Hanqing Guo, and Ning Zhang. 2020. Surfingattack: Interactive hidden attack on voice assistants using ultrasonic guided waves. In *Network and Distributed Systems Security (NDSS) Symposium*.

[50] Xuejing Yuan, Yuxuan Chen, Yue Zhao, Yunhui Long, Xiaokang Liu, Kai Chen, Shengzhi Zhang, Heqing Huang, Xiaofeng Wang, and Carl A Gunter. 2018. Commandersong: A systematic approach for practical adversarial voice recognition. In *27th USENIX Security Symposium (USENIX Security 18)*. 49–64.