Protecting Your Voice from Speech Synthesis Attacks

Zihao Liu
Iowa State University
Ames, Iowa, USA
zihaoliu@iastate.edu

Yan Zhang
Iowa State University
Ames, Iowa, USA
yanzh@iastate.edu

Chenglin Miao
Iowa State University
Ames, Iowa, USA
cmiao@iastate.edu

ABSTRACT

In recent years, much attention has been paid to speech synthesis, which aims to generate synthetic speeches in a voice of a target speaker. Although the speech synthesis technique has facilitated a wide spectrum of applications that positively impact our daily lives, it can also be used by attackers to perform speech synthesis attacks. An attacker can use this technique to mimic the voice of a victim and transform arbitrarily chosen text or voice samples into the same content spoken by the victim. To protect a speaker's voice from speech synthesis attacks, in this paper, we propose two novel defense schemes that can be used by the speaker to process his or her speeches before publishing them on social media platforms or sending them to others. The processed speeches cannot only significantly degrade the performance of speech synthesis systems but also keep the sound of the speaker's voice so that they can still be used for normal purposes. The desirable performance of the proposed defense schemes is verified through extensive experiments conducted on several real-world speaker recognition (SR) systems and a user study on a public crowdsourcing platform.

CCS CONCEPTS

• **Computing methodologies** → **Machine learning**; • **Security and privacy** → **Human and societal aspects of security and privacy**.

KEYWORDS

speech synthesis, malicious attacks, biometric security

ACM Reference Format:
Zihao Liu, Yan Zhang, and Chenglin Miao. 2023. Protecting Your Voice from Speech Synthesis Attacks. In *Annual Computer Security Applications Conference (ACSAC '23), December 04–08, 2023, Austin, TX, USA.* ACM, New York, NY, USA, 15 pages. https://doi.org/10.1145/3627106.3627183

1 INTRODUCTION

Today, we are living in a voice-driven world. The consumption of audio content and voice-based automated services are penetrating every corner of human society and becoming a necessary part of our lives. Many content creators are moving to audio platforms such as SoundCloud [9] and Audible [6]. The tech giants like Google, Amazon, and Apple are investing heavily in their voice-based services. In addition, the pervasive social media platforms (e.g., Facebook, WhatsApp, and TikTok) make it very easy for us to share and enjoy various audio contents.

With the above advancements, more and more users expect to be able to customize the voice of various voice agents as they like. For example, some users may like to hear their familiar voice when using audiobooks or voice-based traffic navigation. To address this need, much attention has been paid to *speech synthesis*, which aims to generate synthetic speech in a voice of a target speaker. The state-of-the-art speech synthesis methods mainly rely on deep neural networks (DNNs) to achieve outstanding performance, and these methods can be divided into two categories: *voice conversion (VC)* [19, 22, 37, 40, 41, 52] and *text-to-speech (TTS)* [16, 29, 30, 36, 51, 57].

The goal of VC is to convert the voice samples of a source speaker into the same content spoken by a target speaker, while TTS aims to transform arbitrary text into the spoken words of the target speaker. Speech synthesis has enabled a wide spectrum of applications with positive effects on our daily life. For example, this technology can help people who have lost their voice communicate with others [10, 43]. It can also benefit spoken language translation [35, 47] and increase human trust in healthcare robots [33].

While we are enjoying the positive uses of speech synthesis, we should not neglect the fact that it can also be used by attackers to perform malicious attacks. According to the Wall Street Journal, in August 2019, criminals used artificial intelligence-based software to impersonate the voice of a CEO who worked in a U.K.-based energy firm and successfully swindled more than $243,000 by speaking on the phone [2]. In October 2021, Forbes reported that speech synthesis had been used in a huge heist, where fraudsters cloned a company director's speech based on this technology and finally stole $35 million from a bank [3]. These reports show that speech synthesis has been misused by malicious parties and brought severe damage in practice. In this paper, we refer to the above attack as *speech synthesis attack*, where an attacker aims to mimic the voice of a target speaker and transform his chosen text or voice samples into the same content spoken by the target. Besides carrying out a heist, speech synthesis attack can also be used for many other malicious purposes. For example, it can fool the voice-based authentication systems built in various devices (e.g., laptops, tablets, or smartphones) and allow the attacker to gain access to these devices [14, 50, 54, 55, 60, 62, 65, 70, 71].

While many defense schemes have been developed to mitigate the abuse of speech synthesis, the majority concentrate on fake speech detection [12, 13, 18, 20, 23, 27, 63, 68, 69]. To the best of our knowledge, only one work, Attack-VC [31], focuses on fake speech prevention. Existing detection algorithms usually achieve the detection goal by discovering artifacts [12, 13, 26, 39, 64] of fake speeches or identifying unique evidence of real speeches, such as liveness evidence [18, 63, 69]. These defenses have been shown to rely heavily on some specific assumptions and recording conditions.

ACSAC '23, December 04–08, 2023, Austin, TX, USA
© 2023 Copyright held by the owner/author(s).
ACM ISBN 979-8-4007-0886-2/23/12.
https://doi.org/10.1145/3627106.3627183

In addition, the detection algorithm is usually implemented after severe consequences have already occurred. The above issues have significantly limited the application of existing detection algorithms. Attack-VC studies the prevention of unauthorized speech synthesis by adding carefully-designed perturbations to the target speaker's speeches before the attacker obtains them. Despite the significant contributions of Attack-VC to this new area, it does have certain limitations. First, Attack-VC faces a challenge in simultaneously reducing the attack's effectiveness while maintaining the normal usability of the target speaker's speeches. Second, this scheme primarily follows a white-box setting, requiring the defender to have full knowledge of a speech synthesis system. However, in reality, users may only have access to the system's API and cannot get its details. Third, Attack-VC lacks efficiency in protecting a speaker's voice, as it necessitates a separate optimization process for each of the target speaker's speech samples to create a protected version.

To address the above limitations, in this paper, we propose a novel defense scheme that aims to prevent the attacker from generating synthetic speeches with high quality. The proposed scheme is based on a black-box setting, and it does not require the details of potential speech synthesis systems. The basic idea of our defense scheme is to modify the target speaker's speeches in the frequency domain before publishing them or sending them to others. Our investigation shows that the speech signal contains some specific frequencies on which the modification can significantly degrade the performance of the speech synthesis models but has little impact on human perception. By identifying these important frequencies for each speech sample and modifying the signal accordingly, the target speaker can effectively protect his or her voice from speech synthesis attacks. Besides, to make the defense efficient, we also propose a speaker-level defense scheme, based on which the target speaker does not need to identify the important frequencies for each of his or her speeches. Instead, the target speaker can use a universal defense strategy to efficiently process any speech. The performance of the proposed defense schemes is evaluated on several real-world speaker recognition (SR) systems. We also conduct a user study to evaluate the impact of these schemes on human perception. Extensive experimental results demonstrate that our defense schemes can significantly degrade the performance of existing speech synthesis systems while guaranteeing that the processed speeches can still be used for their normal purposes.

2 BACKGROUND AND RELATED WORK

2.1 Speech Synthesis

In speech synthesis attacks, the attacker aims to steal a target speaker's voice identity and generate a synthetic version of the target's voice, speaking some specific words chosen by the attacker. The history of generating synthetic speech can be traced back to the 1930s [45], and many speech synthesis methods have been proposed since then [15, 38, 58]. In recent years, with the rapid development of deep learning techniques, deep neural network (DNN)-based speech synthesis has drawn much attention and has significantly improved the quality of the synthesized speech [16, 19, 22, 29, 30, 36, 37, 40, 41, 51, 52, 57]. However, the outstanding performance of DNN-based speech synthesis also strengthens the attackers who want to perform speech synthesis attacks and brings more security concerns. In this paper, we mainly focus on two types of DNN-based speech synthesis systems that can be used to perform the speech synthesis attacks: *voice conversion (VC)* and *text-to-speech (TTS).*

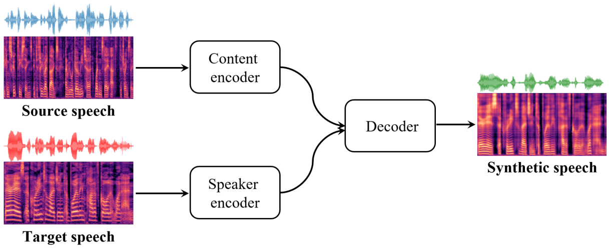


Figure 1: Voice conversion framework.

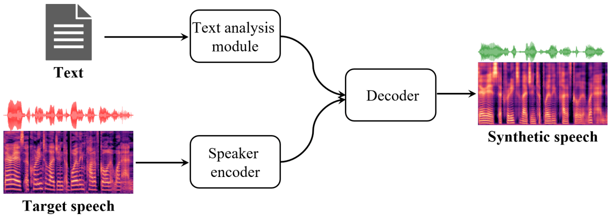


Figure 2: Text-to-speech framework.

2.1.1 *Voice conversion.*

The VC systems aim to convert a speech signal uttered by a source speaker to sound as if it was uttered by a target speaker while keeping the linguistic contents unchanged. Figure 1 shows the general framework of state-of-the-art voice conversion models [32], which mainly contains a content encoder, a speaker encoder, and a decoder. The inputs of the content encoder and the speaker encoder are the speeches provided by the source speaker and the target speaker, respectively. The goal of the content encoder is to extract the content information from the source speech, and the speaker encoder aims to embed the voice characteristics of the target speaker as a latent vector. The outputs of the two encoders are fed into the decoder, which can generate the synthetic speech. The sound of the synthetic speech is similar to that of the target speech, but the content information is the same as that of the source speech. VC has been widely adopted to perform speech synthesis attacks [42, 44, 67], where the source speech is chosen by the attacker, and the target speech is collected by the attacker from the victim speaker. In practice, there are many ways that can be used by the attacker to collect the victim's speech. For example, the attacker may obtain the victim's speech from public or private media. He can also collect the speech samples by recording the victim's speech in a public setting.

2.1.2 *Text-to-speech.*

Similar to VC, TTS takes arbitrary texts and the target utterance that provides voice characteristics as inputs to synthesize a speech [16, 24, 30, 36, 51, 53]. The general framework for the DNN-based TTS systems is shown in Figure 2. The speaker encoder here is similar to that in Figure 1, and it outputs an embedding that captures the voice characteristics of the target speaker. The text analysis module is used to extract the linguistic

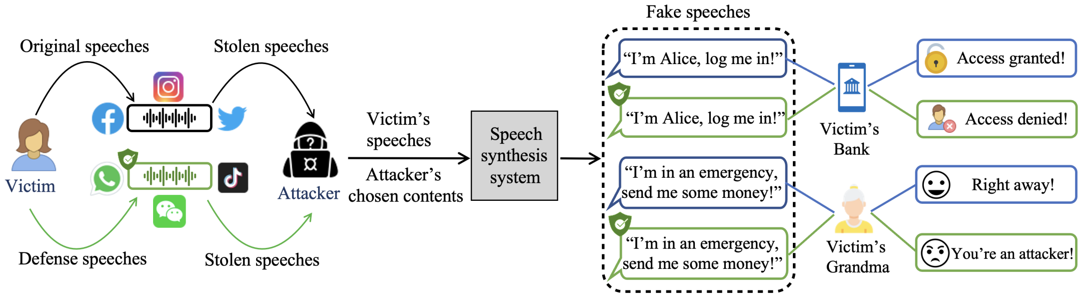


Figure 3: Workflow of the speech synthesis attack and the defense scheme.

features from the input text, which is chosen by the attacker. Both the embedding and linguistic features are fed into the decoder to generate the synthetic speech.

2.2 Defending against Speech Synthesis Attacks

To mitigate speech synthesis attacks, many defense methods have been developed. Most focus on the post-detection of synthetic speeches while only one work, Attack-VC [31], studies how to prevent the attacker from generating synthetic speeches.

Existing detection methods focus either on identifying the speaker or detecting the artifacts of fake speeches. SR systems are engineered to detect speakers. They typically enroll a speaker's voice identity and then verify if a new voice sample matches the enrolled voice identity. However, a recent work [67] found that modern SR systems are vulnerable to speech synthesis attacks [42, 44, 67], highlighting the need for more advanced algorithms to enhance the reliability of SRs. In addition to considering the biometrics embedded in speeches, many works detect fake speeches by searching for supporting evidence. Some of them [18, 63, 69] verify the evidence of whether a speech is spoken by a live person or is uttered by a registered fraudster [23]. Other detection algorithms [12, 13, 56, 68] focus on identifying the traces produced by machines. However, these works usually rely on specific assumptions or recording conditions, such as specific devices and distances between the microphone and the speaker. Additionally, existing detection algorithms are sometimes sensitive to contents and languages [26]. The above issues have significantly limited the application of existing detection algorithms in practice.

Unlike the fake speech detection algorithms implemented after the speech synthesis process, Attack-VC focuses on defense beforehand. This mechanism mitigates the attack by adding carefully-designed perturbations to speech samples. However, to achieve good defense performance, Attack-VC needs to add large perturbations to speeches. As a result, the perturbed speeches do not sound like the original speeches, which may affect the normal usability of these speeches. Besides, Attack-VC is based on white-box setting, where the defender needs to know the details about the speech synthesis system (e.g., model parameters) adopted by the attacker, which are usually difficult to obtain in practice. Additionally, this mechanism requires optimization to generate perturbations for each speech sample, making it inefficient for time-sensitive applications such as sending instant voice messages.

3 PROBLEM SETTING

As shown in Figure 3, we consider a scenario where an attacker aims to imitate a target speaker's voice by launching speech synthesis attacks. We assume that the attacker can obtain some speech samples of the target speaker (i.e., the victim) from public or private media. For example, the attacker can obtain these speech samples from the video/audio published by the target speaker on some social media platforms (e.g., Facebook, Instagram, and TikTok). In addition, the attacker may be a friend of the target speaker and they may send voice messages to each other via messaging apps (e.g., WhatsApp and WeChat). It is easy for the attacker to extract some speech samples from the video/audio or voice messages he obtained. After collecting the speech samples, the attacker uses the speech synthesis system based on either VC or TTS to generate the synthetic speech with arbitrary chosen contents.

Our goal in this paper is to develop a defense scheme that can be used to protect the target speaker's voice from speech synthesis attacks. As shown in Figure 3, the target speaker can use our proposed scheme to process his or her speeches to generate defense speeches before publishing them on social media platforms or sending them to others. Even if the attacker obtains the processed speech samples, he cannot generate his desirable synthetic speeches based on existing speech synthesis systems. In addition, we hope that the proposed defense scheme has little impact on the sound of the target speaker's voice so that the processed speeches can still be used for normal purposes. For example, the processed speeches should be normal for human perception.

We formally define the problem targeted in this paper as follows. Suppose $\boldsymbol{x}$ is the speech sample collected by the attacker from the target speaker. $\mathcal{D}$ denotes the defense strategy that is derived based on our proposed scheme, and it is used to process the target's speeches, and $\boldsymbol{x}_d$ denotes the processed speech after applying $\mathcal{D}$ to $\boldsymbol{x}$ (i.e., $\boldsymbol{x}_d = \mathcal{D}(\boldsymbol{x})$). To measure the impact of $\mathcal{D}$ on $\boldsymbol{x}$, we define the quality change of $\boldsymbol{x}$ after applying the defense strategy as

$$\Delta Q_d(\boldsymbol{x}) = 1 - S(E_s(\boldsymbol{x}_d), E_s(\boldsymbol{x})), \tag{1}$$

where E_s is the speaker encoder. $E_s(\boldsymbol{x}_d)$ and $E_s(\boldsymbol{x})$ are the embeddings of $\boldsymbol{x}_d$ and $\boldsymbol{x}$, respectively. $S(E_s(\boldsymbol{x}_d), E_s(\boldsymbol{x})) \in [0, 1]$ is the similarity score between $E_s(\boldsymbol{x}_d)$ and $E_s(\boldsymbol{x})$. Obviously, the smaller the $\Delta Q_d(\boldsymbol{x})$, the less impact the defense strategy has on $\boldsymbol{x}$. We use $\mathcal{W}$ to denote the speech synthesis model. Please note that in this paper we consider a *black-box setting*, where we do not know the details about the speech synthesis model (e.g., model architecture and parameters), but we can obtain the model output (i.e., the synthetic speech) given an input. Similarly, to measure the impact of $\mathcal{D}$ on the synthetic speech, we define the quality change of the synthetic speech after applying the defense strategy as

$$\Delta Q_I(\boldsymbol{x}) = S(E_s(\mathcal{W}(\boldsymbol{x})), e_s) - S(E_s(\mathcal{W}(\boldsymbol{x}_d)), e_s), \tag{2}$$

where $\mathcal{W}(\boldsymbol{x})$ and $\mathcal{W}(\boldsymbol{x}_d)$ denote the generated synthetic speeches based on $\boldsymbol{x}$ and $\boldsymbol{x}_d$, respectively. e_s is the speaker's average voice embedding, which is derived based on the speaker's real speech samples. $S(E_s(\mathcal{W}(\boldsymbol{x})), e_s)$ represents the similarity score between the embedding of the generated synthetic sample and e_s. Here e_s is used to measure the quality of the synthetic speech. The larger the $S(E_s(\mathcal{W}(\boldsymbol{x})), e_s)$, the better the synthetic speech. $\Delta Q_I(\boldsymbol{x})$ reflects the change of the above similarity score after applying the defense strategy. *Our goal here is to find a defense strategy $\mathcal{D}$ that can maximize $\Delta Q_I(\boldsymbol{x})$ while guaranteeing that $\Delta Q_d(\boldsymbol{x})$ has little impact on the usability of $\boldsymbol{x}$ in benign environments.*

4 METHODOLOGY

4.1 Defense via Frequency Modification

To protect the target's voice from speech synthesis attacks, we propose to modify the target's speeches in the frequency domain before publishing them or sending them to others. Our investigation shows that the speech signal contains some specific frequencies on which the modification can significantly degrade the performance of speech synthesis models but has little impact on human perception. The basic idea of our proposed defense scheme is to identify these specific frequencies and modify the signal on these frequencies. In this paper, we consider three types of modification methods: Zero Mask, Adaptive Noise Mask (AN-Mask), and Gaussian Blur Mask (GB-Mask).

Zero Mask. This method is intuitive and it aims to mask some frequencies of a target speech signal by setting their amplitudes to 0. Suppose $\boldsymbol{x} \in \mathbb{R}^{M \times T}$ is the mel spectrogram of a speech sample produced by the target speaker, where M refers to the dimension of the mel spectrogram in the frequency domain, and the T refers to the dimension in the time domain. We denote such modification method as

$$\mathcal{M}_Z(\boldsymbol{x}, \mathbb{F}) = \{\boldsymbol{x} | a_f^t = 0, \forall f \in \mathbb{F} \text{ and } \forall t \in [0, T]\}, \tag{3}$$

where $\mathbb{F}$ is a set of frequencies that are chosen to modify for $\boldsymbol{x}$. a_f^t is the amplitude of the frequency f at time t.

Adaptive Noise Mask (AN-Mask). In this method, we perturb the speech sample by adding some noises to the amplitudes of the chosen frequencies. Specifically, we denote this method as

$$\mathcal{M}_{AN}(\boldsymbol{x}, \mathbb{F}) = \{\boldsymbol{x} | a_f^t = a_f^t + C(\eta(\cdot)), \forall f \in \mathbb{F} \text{ and } \forall t \in [0, T]\}, \tag{4}$$

where $\eta(\cdot)$ is a noise generation function (e.g., Gaussian noise and Laplace noise), and $C(\cdot)$ refers to the clipping function that constrains the perturbation. To ensure the perturbation remains subtle, we constrain the noise using a constant ϵ and limit the perturbation to a valid range.

Gaussian Blur Mask (GB-Mask). The third type of modification method is based on Gaussian blur, which is a noise reduction low-pass filter that is widely used in image processing. The intuition behind this method is to filter some details of the target speech signal that may help speech synthesis models capture the speaker's voice characters. Specifically, the Gaussian blur smooths speech signals by convolving them with a Gaussian kernel. The Gaussian function for constructing the kernel can be expressed as

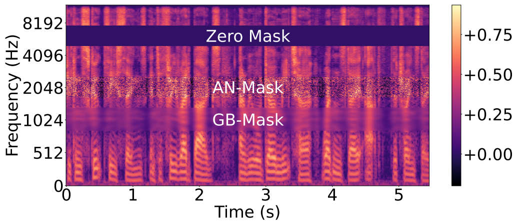


Figure 4: The mel spectrogram with different modification methods.

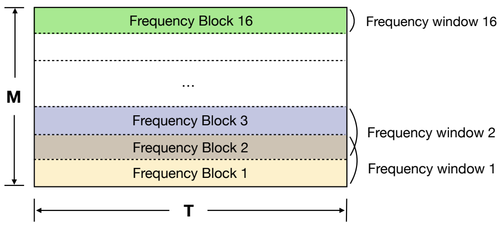


Figure 5: Frequency partition.

$$G(p, q) = \frac{1}{2\pi\sigma^2} e^{-\frac{p^2+q^2}{2\sigma^2}}, \tag{5}$$

where p and q are the distances to the center of the kernel in the horizontal and vertical axis, respectively. Similar to the aforementioned methods, we simplify the process of applying GB-Mask to $\boldsymbol{x}$ as:

$$\mathcal{M}_{GB}(\boldsymbol{x}, \mathbb{F}) = \{\boldsymbol{x} | a_f^t = \phi(a_f^t, G), \forall f \in \mathbb{F} \text{ and } \forall t \in [0, T]\}, \tag{6}$$

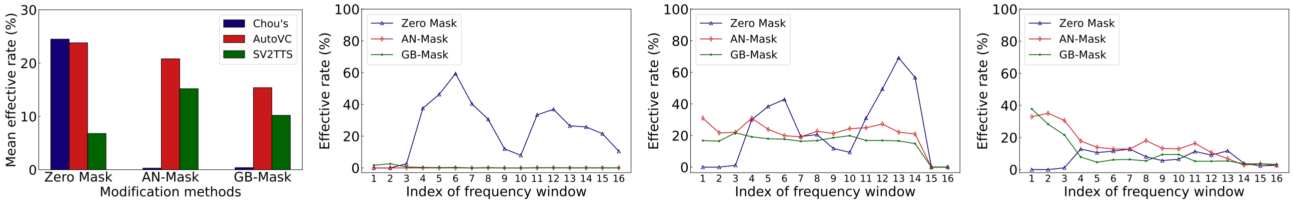
where the $\phi(\cdot, \cdot)$ refers to the mapping function that converts the original amplitude a_f^t to a new value based on the given Gaussian function G.

Figure 4 shows the spectrogram of a speech sample after applying the above three types of modifications to different frequency bands. We can see that different modifications have different impacts on the speech signal, and they can obviously change the pattern of the signal.

To demonstrate the effectiveness of the above modification methods on defending against the speech synthesis attack, we next conduct a case study that aims to answer the following three key questions.

(1) Are the above modification methods effective for different speech synthesis models?
(2) Do the three types of modification methods have different defense effectiveness for a given speech synthesis model?
(3) Does the selection of frequency bands affect the defense effectiveness?

In this case study, we consider three widely used speech synthesis models: Chou's model [22], AutoVC [52], and SV2TTS [36]. The



(a) (b) (c) (d)

Figure 6: Effective rates of different modification methods. (a) Overall effective rates of different modification methods. (b) Effective rates on different frequency windows when Chou's model is used for speech synthesis. (c) Effective rates on different frequency windows when AutoVC is used for speech synthesis. (d) Effective rates on different frequency windows when SV2TTS is used for speech synthesis.

first two models are used for VC, and the third one is used for TTS. In addition, we use the VCTK dataset [59] to study the effectiveness of different modification methods. This dataset contains the speeches of 109 English speakers. For each speech sample $\boldsymbol{x}$, we uniformly partition it into 16 blocks in the frequency domain. As shown in Figure 5, the frequency band that combines two continuous blocks is called a *frequency window*. Please note that the last frequency window contains only one block. For each frequency window, we use the aforementioned three modification methods to modify it and feed the modified sample into each of the three adopted speech synthesis models. Here we take each of the aforementioned modification methods as a defense strategy (i.e., $\mathcal{D}$) and modify one frequency window at a time. Then, we evaluate the quality changes of sample $\boldsymbol{x}$ and the generated synthetic speech (i.e., $\Delta Q_d(\boldsymbol{x})$ and $\Delta Q_I(\boldsymbol{x})$), respectively. The similarity score between different embeddings in this case study is calculated based on cosine similarity function with a well-known speaker encoder, Resemblyzer [1]. For example, if applying Zero-Mask method to a specific frequency window significantly degrades the synthetic speech quality while the speech sample remains nearly unaltered for human perception, we can infer that Zero Mask is effective on defending against the speech synthesis attack. For a given speech synthesis model, we say a modification method $\mathcal{M} \in \{\mathcal{M}_Z, \mathcal{M}_{AN}, \mathcal{M}_{GB}\}$ is effective on frequency window w_i if and only if the following two inequalities are satisfied.

$$\Delta Q_I(\boldsymbol{x}) > \tau_1, \tag{7}$$

$$\Delta Q_d(\boldsymbol{x}) < \tau_2, \tag{8}$$

where τ_1 and τ_2 are two thresholds that imply a noticeable quality decrease on the synthetic speech with an acceptable distortion on the original sample. The parameters in the case study can be found in Section 8.1 of the Appendix.

To quantitatively analyze the effectiveness of the aforementioned modification methods, we further define the *effective rate* of a given modification method $\mathcal{M}$ on a specific frequency window w_i as

$$r_i^{\mathcal{M}} = \frac{\sum_{j=1}^{N} n_j^i}{N}, \tag{9}$$

where N is the total number of considered speech samples (1000 in this case study). n_j^i equals to 1 if Eq. (7) and Eq. (8) are satisfied after applying $\mathcal{M}$ to frequency window w_i of the j-th sample, otherwise n_j^i equals to 0.

Figure 6a shows the average effective rates of the three modification methods over all frequency windows. We can observe that the modification methods can be effective in defending against different speech synthesis models, but they have different effective rates for every speech synthesis model. For example, Zero Mask can be very effective on Chou's model while the effective rates of AN-Mask and GB-Mask are very low on this model. Figure 6b, Figure 6c, and Figure 6d report the effective rates of the considered modification methods on different frequency windows when Chou's model, AutoVC, and SV2TTS are used for speech synthesis, respectively. These figures show that the selection of the frequency bands plays an important role in defending against the speech synthesis attack. The aforementioned modification methods can be very effective on some specific frequency bands.

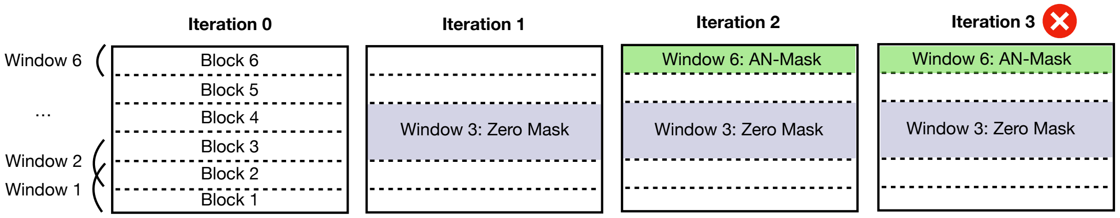
In summary, different modification methods have different defense effectiveness not only on a specific speech synthesis model but also on a specific frequency band. In addition, a specific modification method can behave differently on different speech synthesis models or on different frequency bands. These findings indicate that it is necessary to identify the most important frequency bands and derive the optimal combination of different modification methods to maximize the possibility of achieving the defense goal.

4.2 The Optimal Defense Strategy

Next we discuss how to derive an optimal defense strategy to protect the target speaker's voice from speech synthesis attacks. Recall that our goal in this paper is to maximize the quality change of the synthetic speech after applying the defense strategy while guaranteeing that the processed speech sample can still be used for its normal purposes. We formulate the problem of finding the optimal defense strategy as the following optimization problem.

$$\begin{aligned} \max_{\mathcal{D}} \ & \Delta Q_I(\boldsymbol{x}) \\ \text{s.t.} \ & \Delta Q_d(\boldsymbol{x}) < \tau_d, \end{aligned} \tag{10}$$

where $\boldsymbol{x}$ is a speech sample of the target speaker, and τ_d is a customized threshold that reflects how much the quality change (distortion) on $\boldsymbol{x}$ the target speaker can accept. In practice, τ_d can be selected based on the variance observed in the target speaker's voice across different sentences. The method used to select τ_d in our experiments is discussed in Section 5.3.



	Frequency Sensitivity			Frequency Sensitivity			Stop conditions activated		
Window index	Zero Mask	AN-Mask	GB-Mask	Zero Mask	AN-Mask	GB-Mask	Zero Mask	AN-Mask	GB-Mask
6	2	6	3	1	5	2	-	-	-
5	2	1	4	-	0	1	-	-	-
4	5	4	3	3	1	2	-	-	-
3	10	2	5	-	-	-	-	-	-
2	-	3	1	-	1	-	-	-	-
1	-	1	-	-	-	-	-	-	-

Figure 7: An example for iteration search.

Suppose sample $\boldsymbol{x}$ is uniformly partitioned into B blocks in the frequency domain, and two consecutive blocks form a frequency window (as shown in Figure 5). We represent the defense strategy $\mathcal{D}$ as a sequence that contains a series of pairs, i.e., $\mathcal{D} = \{(b_i, \mathcal{M}_i)\}_{i=1}^{P}$, where $b_i \in [1, B]$ is the block number, and $\mathcal{M}_i \in \{\mathcal{M}_Z, \mathcal{M}_{AN}, \mathcal{M}_{GB}\}$ is one of the modification methods introduced in Section 4.1. P is the number of pairs in the sequence. Given a specific speech synthesis model, the defense strategy $\mathcal{D}$ provides a guidance on how to process the target speaker's speeches so that the attacker cannot use them to generate synthetic speeches with high quality. Specifically, before uploading $\boldsymbol{x}$ to public social media platforms or sending it to others, the target speaker can process $\boldsymbol{x}$ by modifying the frequency blocks in sequence $\mathcal{D}$ using corresponding modification methods.

Since the selection of the modification method for each frequency block is not a continuous process, it is difficult to directly use the gradient-based method to solve the optimization problem in Eq. (10). To address this challenge, we develop an effective solution based on iterative search. The basic idea of this solution is to iteratively search a pair $(w_j, \mathcal{M}_j)$ that can maximize $\Delta Q_I(\boldsymbol{x})$ while guaranteeing that $\Delta Q_d(\boldsymbol{x})$ is less than threshold τ_d. w_j is a frequency window as shown in Figure 5. $\mathcal{M}_j$ is one of the aforementioned modification methods, which is used to modify sample $\boldsymbol{x}$ within frequency window w_j. The reason why we use the frequency window instead of the frequency block during the iterative search is that we want to search some overlapped areas to enhance the search robustness while two consecutive frequency blocks are independent.

To measure the effect of a pair $(w_j, \mathcal{M}_j)$ in the t-th iteration, we introduce a metric called *frequency sensitivity* that is defined as

$$s_j^t = \frac{\Delta Q_I^t(\boldsymbol{x}) - \Delta Q_I^{t-1}(\boldsymbol{x})}{\Delta Q_d^t(\boldsymbol{x}) - \Delta Q_d^{t-1}(\boldsymbol{x})}, \tag{11}$$

where $\Delta Q_I^t(\boldsymbol{x})$ and $\Delta Q_d^t(\boldsymbol{x})$ are calculated after taking into account $(w_j, \mathcal{M}_j)$ in the t-th iteration. s_j^t reflects how much $(w_j, \mathcal{M}_j)$ can change the values of $\Delta Q_I^t(\boldsymbol{x})$ and $\Delta Q_d^t(\boldsymbol{x})$ compared with that in the $(t-1)$-th iteration. The larger the value of s_j^t, the better the pair $(w_j, \mathcal{M}_j)$. In each iteration, we apply different modification methods to each of the frequency windows that are not selected in previous iterations and calculate the corresponding frequency sensitivities. Then, we select the pair with the largest frequency sensitivity and add the corresponding frequency block and modification method pairs to sequence $\mathcal{D}$. For example, if $(w_j, \mathcal{M}_j)$ is selected in iteration t, the pairs $(b_j, \mathcal{M}_j)$ and $(b_{j+1}, \mathcal{M}_j)$ will be added to sequence $\mathcal{D}$ if w_j is not the last frequency window. If w_j is the last frequency window, $(b_j, \mathcal{M}_j)$ will be added to sequence $\mathcal{D}$. The iteration search will stop if $\Delta Q_I^t(\boldsymbol{x}) - \Delta Q_I^{t-1}(\boldsymbol{x}) \leq 0$ or $\Delta Q_d^t(\boldsymbol{x}) \geq \tau_d$.

Figure 7 presents an example of our proposed solution. Here we divide the sample into six frequency blocks, which form 6 frequency windows. Both $\Delta Q_I^0(x)$ and $\Delta Q_d^0(x)$ are set to zero in the initial state (i.e., iteration 0). In the first iteration, we scan the frequency window in a specific order, for instance, from the bottom to the top, calculating the frequency sensitivities for all possible $(w_j, \mathcal{M}_j)$. There are a total of 6 × 3 sensitivities generated in this iteration. We then compare these values and identify the $(w_j, \mathcal{M}_j)$ that results in the largest frequency sensitivity. For instance, $(w_3, \mathcal{M}_Z)$ has the largest frequency sensitivity (s_3^1=10), so we add $(b_3, \mathcal{M}_Z)$ and $(b_4, \mathcal{M}_Z)$ to $\mathcal{D}$. The symbol "-" in the example denotes a frequency window that has already been masked or where the sample distortion exceeds τ_d with that pair, and it will not be selected. Using the sample generated in the first iteration, we repeat the process in the second iteration and find s_6^2=5 with AN-Mask. Then, we add $(b_6, \mathcal{M}_{AN})$ to $\mathcal{D}$. In the third iteration, we find that the sample distortion with any pair exceeds τ_d, triggering the stop condition. In this case, the final defense strategy $\mathcal{D}$ would be $\{(b_3, \mathcal{M}_Z), (b_4, \mathcal{M}_Z), (b_6, \mathcal{M}_{AN})\}$.

Since using a fixed threshold may cause a considerable variance of sample distortion, in our implementation, we increase the stability of the defense sample generation by flexibly controlling the sample distortion into the range of $[\tau_d - \rho, \tau_d + \rho]$, where parameter ρ has a small value. If the sample distortion does not fall into that range when the search stops based on the above stopping conditions, we can further mask half area of the block in the last iteration based on binary search until the requirement is satisfied or reaches the last split.

Although the above solution can effectively derive an optimal defense strategy, it has high complexity because it needs to iteratively search all frequency windows, even some windows that cannot benefit the defense according to the results in previous iterations. To make the solution more efficient, we can utilize the Tree-structured Parzen Estimator (TPE) [17], a widely-used algorithm for hyperparameter optimization. The TPE algorithm is an iterative process that uses the historical information of the evaluated hyperparameters to construct a probabilistic model, guiding the selection of the next hyperparameter set to optimize a given objective function. By taking historical information into account, TPE usually requires fewer function evaluations (trails) than traditional grid search methods, yet delivers comparable results. In this paper, we consider the pair consisting of the frequency window index and the modification method as the hyperparameters to optimize. The frequency sensitivity (i.e., s_j^t) serves as the predefined objective function. Based on the TPE algorithm, we first sample a few random configurations of the hyperparameters and evaluate the objective function. Then, we split the results into more successful and less successful based on a threshold (e.g., the median of the objective values). Next, we fit a Gaussian Mixture Model (GMM) to the configurations of each group and propose a new configuration of the hyperparameters based on the ratio of the two GMMs. After proposing the new configuration, we evaluate the objective function and update the above GMMs. The aforementioned steps will continue until a stopping criterion is met. This procedure will guide the search towards regions of the configuration space that are more likely to yield good results (i.e., larger values of s_j^t) based on the GMMs' understanding of the observed data.

4.3 Speaker-level Defense

In the above section, we mainly discuss how to derive the optimal defense strategy for a single speech sample. To defend against the speech synthesis attack, the speaker can use the above algorithm to derive an optimal defense strategy for each of his or her speech samples, and then modify each sample based on the corresponding defense strategy before uploading it to social media platforms or sending it to others. However, in some cases, a speaker needs to send instant audio messages to others, and the above algorithm is still not efficient enough. To address this challenge, we further propose to derive a speaker-level defense strategy that is general enough to be directly applied to any speech of the speaker.

To derive the speaker-level defense strategy, we first collect K speech samples of the speaker. Then, we apply the above algorithm to each sample k and derive the corresponding optimal defense strategy (denoted as sequence $\mathcal{D}_k$). Next we combine the K sequences and derive a new sequence $\mathcal{D}' = \{(b_i, \mathcal{M}_i, m_i)\}_{i=1}^{Q}$, where m_i is the total times that $(b_i, \mathcal{M}_i)$ appears in the derived sequences for all K samples. We then rank the pairs in $\mathcal{D}'$ based on descending order of m_i. Please note that the derivation of $\mathcal{D}'$ can be conducted offline. After a speaker obtains his or her defense strategy $\mathcal{D}'$, the speaker can directly apply the ranked $\mathcal{D}'$ to process an arbitrary speech sample. Specifically, the speaker first modifies the sample based on the first pair in the ranked $\mathcal{D}'$ and then modifies it using the following pairs in order until the stopping condition is satisfied. Here the stopping condition is the same as that in Section 4.2.

5 PERFORMANCE EVALUATION

5.1 Speech Synthesis Models

We evaluate our defense schemes under a realistic and high-risk scenario where the attacker uses zero-shot style speech synthesis models to clone unseen target speakers' voices. Specifically, we chose two VC models, Chou's model [22] and AutoVC [52], and one TTS model, SV2TTS [36], as they have been widely used and demonstrated strong generalization to unseen speakers. They all share the general VC or TTS frameworks introduced in Section 2.

Chou's model. This model employs adaptive instance normalization to achieve the goal of generating the voice of unseen speakers. It utilizes an encoder-decoder structure. The speaker encoder takes a 512-dim mel spectrogram as input and generates a 128-dim speaker embedding as the representation of the target speaker's speech. The decoder takes the content embedding extracted from the content module and the speaker embedding as inputs to output a synthesized 512-dim mel spectrogram. To recover the waveform from the synthesized spectrogram, Chou's model leverages Griffin-Lim [28] as the vocoder. In this paper, we adopted the same pretrained model used in the official implementation of Attack-VC [4].

AutoVC. AutoVC also utilizes an encoder-decoder structure. The author design an encoder bottleneck to match unseen speakers' distribution. The speaker encoder adapts a pre-trained d-vector using the generalized end-to-end (GE2E) loss [61] to generate a 256-dim speaker embedding, with an 80-dim mel spectrogram as input. The AutoVC model used in this paper is trained on the VoxCeleb1 [46] and LibriSpeech datasets [49].

SV2TTS. SV2TTS is a text-dependent TTS model that carries out speech synthesis in three stages. First, it uses an LSTM speaker encoder to capture the speaker's voice characteristics. Then, it applies Tatocron 2 [66] for spectrogram synthesis, and finally, it employs the WaveNet Vocoder [48] for waveform generation. The speaker encoder in this model takes a 40-dim mel spectrogram as the input and generates a 256-dim speaker embedding. Here we adopt the public implementation of this model [34], where the speaker encoder is trained on VoxCeleb1/2 [46] and LibriSpeech [49], and the synthesis network Tatocron 2 is trained on LibriSpeech. In our implementation of SV2TTS, we adopt one of ten phrases of normal conversation as text input for synthesizing each speech, with detail listed in the Section 8.2 of the Appendix.

5.2 Dataset and Baselines

We conduct our experiments using CSTR VCTK corpus [59], which contains 109 speakers with different genders and accents. The samples are collected by reading short newspaper phrases, Rainbow passages [25], and elicitation paragraphs.

In our experiments, we consider the following two baselines.

- **Raw**. In this baseline, we do not consider any defense scheme. The attacker uses the raw speech samples of the target speaker to generate synthetic speeches. We follow the same baseline implementation as described in [31, 67].
- **Attack-VC**. Attack-VC [31] is the only method in the literature that studies the same problem as ours. Attack-VC achieves the defense goal by adding carefully designed noise to the speech samples before uploading them to social media platforms and sending them to others.

Table 1: Attack success rate (ASR) on Resemblyzer (%).

	Chou's			AutoVC			SV2TTS	
	Attack-VC	**SampleMask**	**SpeakerMask**	Attack-VC	**SampleMask**	**SpeakerMask**	**SampleMask**	**SpeakerMask**
$\tau_d = 0.06$	69.7	**18.2**	**38.8**	34.3	**19.1**	**24.8**	**19.4**	**49.0**
$\tau_d = 0.12$	46.3	**9.2**	**17.1**	29.3	**13.0**	**15.1**	**8.3**	**29.9**
$\tau_d = 0.18$	30.3	**0.9**	**9.4**	17.2	**6.5**	**10.9**	**3.5**	**13.5**

Please note that our proposed defense scheme for single speech samples in Section 4.2 is denoted as **SampleMask**, and the proposed speaker-level defense scheme in Section 4.3 is denoted as **SpeakerMask**.

5.3 Experimental Setup

Speaker encoder E_s and embedding e_s. To calculate the quality changes described in Eq. (1) and Eq. (2), we use the LSTM speaker encoder trained by Resembylzer [1] as E_s. This encoder is widely adopted and it shows a remarkable ability to distinguish different speakers. For a speaker's average voice embedding e_s, we calculate it with the above speaker encoder using the speaker's 10 speech samples that are randomly selected from the VCTK dataset. **The threshold τ_d.** This threshold reflects how much the quality change on the speech sample the target speaker can accept after applying the defense strategy. In our experiments, we determine τ_d based on the following method. For each speaker in the VCTK dataset, we randomly select 10 speech samples and calculate the similarity score between the embedding of each sample and the average voice embedding e_s. Here cosine similarity is used to calculate the similarity score. Then, we can derive an average similarity score over the randomly selected 10 speech samples for each speaker. Finally, we calculate the average similarity score over the 109 speakers in the dataset and its value is 0.88, based on which we can derive that the average difference between a speaker's speech embeddings and his or her e_s is 0.12. We set τ_d to 0.12 in our experiments. Since all the speech samples in the VCTK dataset are collected without any modification, the above difference reflects the variance of a speaker's voice when he or she say different sentences.

Other parameters. In the paper, we partition the spectrogram into 16 frequency blocks. We generate standard Gaussian noise for the AN-Mask and clip the noise with a magnitude constraint of 0.1. We use (11,11) as kernel size and set the standard deviation to 1.5 for generating GB-Mask. We apply the modification on a 512-dim mel spectrogram in all experiments, using the same parameter configuration as in [31]. The relax bound ρ is set to 0.02.

Ethics. To study the performance of our proposed defense schemes on real-world SR systems, we recruit some English speakers and collect some of their voice recordings. We also conduct a user study with human participants to assess the impact of our defense on human perception. These studies have received approval from the IRB. The details of them can be found in Section 5.4 and Section 5.5.

5.4 Experimental Results

We first use real-world SR systems to evaluate the performance of the proposed defense schemes. Specifically, we study whether the synthetic speeches generated by the attacker can fool SR systems or let SR systems believe that the synthetic speech is from the target speaker. We use the *attack success rate (ASR)* as the evaluation metric, which is defined as the percentage of the synthetic speeches that successfully fool a specific SR system. The lower the ASR, the better the defense scheme. In our paper, we consider four state-of-the-art SR systems: Resemblyzer [1], Microsoft Azure [7], Amazon Alexa [5], and WeChat [11].

Resemblyzer. Resemblyzer is an open-source speaker encoder that is widely adopted for SR. It enrolls each speaker with his or her real speeches and generates an embedding to represent the speaker's voice identity. To recognize a speaker, Resemblyzer calculates the embedding of the input speech and compare it with the enrolled embeddings, using cosine similarity as a metric. Then a threshold is used to determine whether two embeddings belong to the same speaker.

For each speaker from VCTK dataset, we randomly select their 100 speech samples and derive 100 synthetic speeches based on the selected samples. Then, we calculate the ASR of all synthetic speeches. Table 1 reports the ASRs of the different speech synthesis models when different defense schemes are applied. Here we consider three cases where the values of τ_d are set to 0.06, 0.12, and 0.18, respectively. Since parameter τ_d does not have impact on Raw, we do not show the results of Raw in the table. With Raw, the ASRs of Chou's model, AutoVC, and SV2TTS are 84.1%, 52.4%, and 57.1%, respectively. For Attack-VC, the added noise to a speech sample is determined by parameter ϵ, which is a constraint making the perturbation subtle. Here we assign appropriate values to ϵ in the above three scenarios so that Attack-VC and our proposed schemes can generate similar distortion on the speech sample when they are applied to the same case. The results in Table 1 show that our proposed defense schemes perform much better than Attack-VC in all cases. For example, when the value of τ_d is 0.12, SampleMask can reduce the ASR of Chou's model from 84.1% to 9.2% while Attack-VC can only reduce that to 46.3%. When being applied to unseen speech samples, our scheme SpeakerMask can reduce the ASR of Chou's model from 84.1% to 17.1%. The results demonstrate that our scheme can still achieve good performance even though the test sample is not involved in the optimization, indicating the good effectiveness and generalizability of our defense. Although the authors in [31] do not evaluate the performance of Attack-VC on SV2TTS, we explore the possibility of using Attack-VC to defend against SV2TTS. However, we find that all defense samples are beyond human perception. So we only show the results of our schemes on SV2TTS in Table 1.

Microsoft Azure. Azure is a real-world, open-API SR system that has been officially adopted by many international entrepreneurs. Users can register their voice profiles using approximately 20-50 seconds of authentic speech. For an input speech sample, Azure's backend can clearly display a message indicating whether the sample is accepted or rejected. An attack is considered successful against Azure if the backend indicates that the test speech has been accepted. In this experiment, we still consider the 109 speakers in the VCTK dataset and enroll them with their speeches. We feed the same 10900 synthetic speeches generated in the experiment for Resemblyzer into Azure. Here we only evaluate our speaker-level defense scheme SpeakerMask and set the value of τ_d to 0.12. Please note that the derived defense strategy for each speaker is a combination of the three modification methods discussed in Section 4.1. The results are shown in Figure 8. We can observe that the speech synthesis models still can easily fool Azure when there is no defense, but they have lower ASRs compared with Resemblyzer. However, our proposed defense scheme SpeakerMask can significantly reduce the ASR of different speech synthesis models and outperform Attack-VC on both Chou's model and AutoVC.

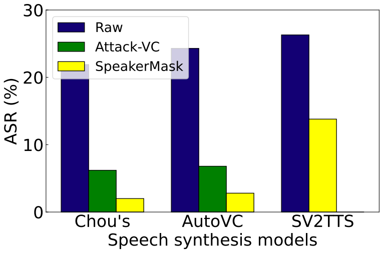


Figure 8: Attack success rate (ASR) on Microsoft Azure.

In addition, we study the effect of our proposed defense on the usability of speakers' speeches. Specifically, we feed the modified speeches of the 109 speakers into both Resemblyzer and Azure, and evaluate the acceptance rate (ACR), which is defined as the percentage of the modified speeches that are successfully recognized by the SR system. The results are shown in Table 2. Here we modify speeches using the defense strategies derived based on Chou's model, AutoVC, and SV2TTS, respectively. Please note that the ACRs of Resemblyzer and Azure on unmodified speeches are 100% and 94.1%, respectively. We can observe that the two SR systems exhibit high ACRs when using the modified speeches. This indicates that speeches processed by our proposed defense retain high usability.

Table 2: Acceptance Rate (ACR) of modified speeches (%).

	Chou's	AutoVC	SV2TTS
Resemblyzer	100	100	100
Azure	89.9	84.7	90.1

Amazon Alexa. As a popular virtual assistant embedded in Amazon's smart speaker, Alexa has been widely used for customizing user interactions and control access to sensitive apps like email and calendar [5]. Alexa does not provide an API for our evaluation, and its speaker verification mechanism is black-box for users. Alexa allows users to enroll their voice by uttering simple phrases, utilizing a text-dependent speaker verification system accessible on mobile apps or various IoT devices [5]. It differs from the above SR systems, as Alexa does not explicitly indicate if an attack is successful. In this experiment, we follow the design in [67] and deem an attack successful if Alexa responds to the synthetic speech the same way it responds to a non-synthesized version of the speech. We recruit 12 English speaks (7 males/5 females) and gather a small collection of their voice recordings as the target speeches to synthesize test commands. Each participant is asked to read 20 phrases from the Rainbow Passage, a resource widely utilized in linguistic studies. The detailed phrases are listed in Section 8.2 of the Appendix. 8 participants record their speeches using a voice memo app on an iPhone 11+, and 4 participants use MacBook Pros. The distance between each participant and the microphone is 6 inches. Here we still evaluate our speaker-level defense scheme SpeakerMask. For each speaker, we use the first ten phrases to generate the defense strategy and then apply it to the remaining phrases. The modified speeches are then fed into different speech synthesis models to generate synthetic commands.

As shown in Table 3, for each participant, we test 7 commands that may disclose users' private information if the attack succeeds. The results in Table 3 show that the Chou's model achieves an overall ASR of 48.8% without any defense scheme. After we apply Attack-VC and SpeakerMask, the ASR of the Chou's model is decreased to 36.9% and 8.3%, respectively. In this experiment, we only consider the scenario where $\tau_d = 0.12$. For AutoVC, its overall ASR without any defense is 21.4%. However, after applying the defense schemes, the ASR is increased to 23.8% with Attack-VC while SpeakerMask can reduce it to 1.2%. These results demonstrate the good performance of our proposed defense scheme in defending against speech synthesis attacks based on VC models. When the attacker uses SV2TTS to perform the attack, our defense scheme SpeakerMask can reduce the overall ASR from 69.0% to 52.4%. We can observe that the performance of SpeakerMask on SV2TTS is not as good as that on the above two VC models. The reason may be that Alexa is more sensitive to the content of the command than voice identity, and SV2TTS can generate more understandable content compared with the above VC models.

WeChat. WeChat is a popular chatting and payment App. It employs a "voice lock" to be an alternative option for entering the password. Users can log in to their accounts by speaking a system-assigned 8-digit number, with a maximum of six daily attempts. If a speech matches the enrolled voice of a speaker and the assigned number, the system will allow the speaker to log in. In this experiment, we also recruit 12 English speakers (7 males/5 females) as the users and collect 20 voice recordings for each user using the same method discussed in the experiment for Alexa. We still use the first 10 recordings of each user to generate the speaker-lever defense scheme and then test the scheme on the remaining 10 samples of each user. Our experiment synthesizes six login numbers using different source samples spoken by a same-gender speaker in two VC models and the number text in SV2TTS. An attack is considered successful if at least one attempt enables the user to log in with the

Table 3: Attack success rate (ASR) on Amazon Alexa (%).

Commands	Chou's			AutoVC			SV2TTS	
	Raw	Attack-VC	**SpeakerMask**	Raw	Attack-VC	**SpeakerMask**	Raw	**SpeakerMask**
Hey Alexa add an event to my calendar for tomorrow at 5.	50.0	16.7	**8.3**	16.7	0.0	**0.0**	83.3	**75.0**
Hey Alexa check my email	41.7	25.0	**0.0**	25.0	33.3	**0.0**	41.7	**41.7**
Alexa say who is talking with you now	50.0	33.3	**16.7**	16.7	16.7	**0.0**	50.0	**33.3**
Alexa tell me what is on my calendar	66.7	75.0	**16.7**	33.3	41.7	**8.3**	91.7	**66.7**
Tell me what is on my calendar for this week	58.3	66.7	**8.3**	25.0	41.7	**0.0**	75.0	**58.3**
Alexa make an appointment with my doctor	50.0	41.7	**8.3**	33.3	25.0	**0.0**	83.3	**50.0**
Hey Alexa make a donation to the American Cancer Institute	25.0	0.0	**0.0**	0.0	8.3	**0.0**	58.3	**41.7**
< Average across the above 7 commands >	48.8	36.9	**8.3**	21.4	23.8	**1.2**	69.0	**52.4**

Table 4: User study for real samples.

Answers	Yes (%)	Unsure (%)	No (%)
Real A/Real A	80.9	14.2	4.9
Real A/Real B	6.2	11.5	82.3

Table 5: User study for defense samples.

	Chou's		AutoVC		SV2TTS
	Attack-VC	SpeakerMask	Attack-VC	SpeakerMask	SpeakerMask
Yes (%)	70.9	71.5	70.4	69.9	73.7
Unsure (%)	13.1	12.8	16.6	13.5	15.4
No (%)	16.0	15.7	13.0	16.6	10.9

synthetic commands. Our results show that none of the users can log in with the synthetic commands generated by Chou's model and AutoVC. So there is no need to perform the defense on the two VC models. However, 5 of 12 (ASR = 41.6%) users can successfully log in to their accounts with the synthetic speeches generated by SV2TTS where there is no defense. After applying our speaker-level defense SpeakerMask (τ_d = 0.12), only one user can log in to the account, i.e., the ASR is decreased from 41.6% to 8.3%, which further demonstrates the effectiveness of our proposed scheme.

5.5 User Study

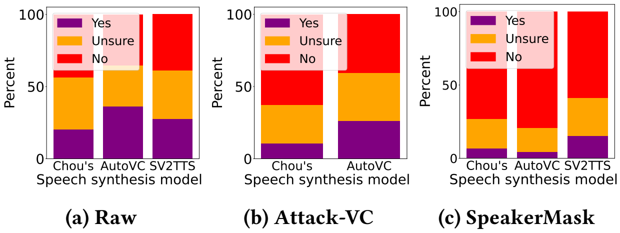
Next, we conduct a user study to evaluate the impact of our defense on human perception. Specifically, we aim to answer two questions: *(1) Can the proposed defense affect the normal usability of a speaker's speeches? (2) Can the synthetic speeches generated by speech synthesis attacks still fool many humans after performing our proposed defense?* In this study, we recruit 80 self-identified English-speaking participants from a public crowdsourcing platform: Amazon Mechanical Turk (Mturk) [8], which has been widely used in research and business [21]. Mturk accommodates participants from various age groups and genders, and all participants were 18 years old or older. We ask the participants to complete an online survey, which is designed to take an average of 5 minutes to complete, and each participant receives a compensation of 1 dollar. We also requested participants to input a predefined task code to eliminate potential bots.

Survey details. In the online survey, we provide 16 audio pairs to each participant. The participants are asked to listen to each audio pair first and then answer the question: Are the two audio samples from the same speaker? The candidate answers are "Yes", "Unsure", and "No". Each audio pair is one of the following combinations: (1) Real A/Real A (two real speech samples from the same speaker); (2) Real A/Real B (two real speech samples from different speakers); (3) Real A/Defense A (one real speech sample from a speaker and its corresponding defense sample generated by a defense scheme); (4) Real A/Fake A (one real speech sample from a speaker and its corresponding synthetic speech sample generated by a speech synthesis model).

Results. The benchmark in this study is users' response to "Real A/Real A" and "Real A/Real B", which can reflect users' ability to distinguish different speakers. As shown in Table 4, 80.9% users choose "Yes" for two samples from the same speaker (Real A/Real A), and 82.3% users choose "No" for two samples from different speakers (Real A/Real B).

Can the proposed defense affect the normal usability of a speaker's speeches? Table 5 shows the answers for "Real A/Defense A". The speech samples here are generated by Attack-VC and SpeakerMask when the speech synthesis models are Chou's model, AutoVC, and SV2TTS, respectively. We can observe that both Attack-VC and our proposed defense can slightly affect the usability of speeches, but the effects are acceptable. For instance, when the attack model is Chou's model and τ_d = 0.12, 70.9% and 71.5% of the users still choose "Yes" for the defense samples generated by Attack-VC and SpeakerMask, respectively. These results are 10.0% and 9.4% lower than that for unprocessed speeches (80.9%), which demonstrates that our proposed defense has little impact on the normal usability of a speaker's speeches.

Can the synthetic speeches generated by speech synthesis attacks still fool many humans after performing our proposed defense? The answers for "Real A/Fake A" are shown in Figure 9, in which 20.4%, 36.2%, and 27.8% of the users choose "Yes" for the synthetic speeches generated by Chou's model, AutoVC, and SV2TTS, respectively, when there is not any defense (Raw). Although all participants are told that fake speeches may exist before filling out the survey, many of them are still fooled by the synthetic speeches. These results again demonstrate the threats of speech synthesis



(a) Raw **(b) Attack-VC** **(c) SpeakerMask**

Figure 9: User study for synthetic samples.

attacks. After performing the defense, both Attack-VC and SpeakerMask can reduce the chance that the participants are fooled, and many participants choose "Unsure" and "No" when being asked whether the two audio sample are from the same speaker. The results also show that our scheme can outperform Attack-VC in all cases. There are fewer participants believing that the real speech sample and the corresponding synthetic speech are from the same speaker after applying our proposed defense. For example, when AutoVC is adopted as the attack model, the participants who believe the target speech and the corresponding synthetic speech are from the same speaker is decreased from 36.2% to 4.5%.

5.6 Transferability

In this experiment, we study the transferability of our proposed defense. Specifically, we study whether the derived defense strategy based on a specific speech synthesis model can be applied to defend against other synthesis models. Here we still take Chou's model, AutoVC, and SV2TTS as the speech synthesis models, and use the VCTK dataset for our study. We first generate defense samples using SpeakerMask based on a specific speech synthesis model and feed these defense samples into different speech synthesis models to generate corresponding synthetic speeches. Then we use Resemblyzer as a SR system to evaluate the attack performance. We consider the case where $\tau_d = 0.12$, and the setting in this experiment is similar to that in the experiment for Resemblyzer. The results are shown in Table 6. The models in the first column are used for deriving the defense samples, and the models in the first row are used for generating synthetic speeches. We can observe that the defense performance is slightly degraded when the strategy derived based on a specific synthesis model is used to defend against other synthesis models. However, the defense is still effective and the ASRs are much lower than that without any defense scheme (the ASRs of Chou's model, AutoVC, and SV2TTS without any defense are 84.1%, 52.4%, and 57.1%, respectively). These results demonstrate that our proposed defense has good transferability. Even though a speaker cannot know which speech synthesis model will be used by the attacker, the speaker can still use the proposed defense scheme to protect his or her voice.

5.7 Efficiency

The efficiency of generating defense samples is also an important factor. In this experiment, we evaluate the average time it takes to generate a defense sample using our speaker-level defense SpeakerMask and Attack-VC (sample lengths are typically between 3-10 seconds) based on 1000 samples. The results are shown in Table 7. Here we consider three cases where the values of τ_d are set to 0.06, 0.12, and 0.18, respectively. We can observe that SpeakerMask takes around only one second to generate a defense sample while Attack-VC takes more than 30 seconds. The results show that our speaker-level defense is efficient in processing speeches, and it enables a speaker to send instant messages to others.

Table 6: ASR for defense transferability (%).

	Chou's	AutoVC	SV2TTS
Chou's	**17.1**	21.1	33.3
AutoVC	40.0	**15.1**	32.1
SV2TTS	42.4	25.6	**29.9**

Table 7: Average time of generating a defense sample (s).

	Chou's	AutoVC	SV2TTS
$\tau_d = 0.06$	0.9	1.0	1.2
$\tau_d = 0.12$	1.1	1.3	1.4
$\tau_d = 0.18$	1.3	1.5	2.0
Attack-VC	31.2	49.2	-

6 LIMITATIONS AND FUTURE WORK

In this paper, we consider speech synthesis models based on English corpora, and our evaluation involves only English speakers. We have not assessed the defense performance on other languages. Further study on other languages will be our future work. In addition, the voice samples mentioned in Section 5.4 were collected in indoor settings. The experimental results demonstrate that our defense is effective with natural indoor noise levels. In our future work, we will further study the performance of the proposed schemes in diverse settings with significant noise levels.

7 CONCLUSION

In this paper, we study how to protect a speaker's voice from speech synthesis attacks. We propose a novel defense scheme that can significantly degrade the performance of existing speech synthesis models by modifying the speaker's speeches in the frequency domain. The modification in the proposed defense scheme has little impact on the quality of speeches, and the modified speeches can still be used for their normal purposes. To improve the efficiency of the defense, we also propose a speaker-level scheme that can produce a universal defense strategy for each speaker. Based on this universal defense strategy, the speaker can efficiently process any of his or her speeches. Extensive experiments are conducted on real-world SR systems to evaluate the performance of the proposed schemes. We also conduct a user study using a public crowdsourcing platform to evaluate the impact of the proposed schemes on human perception. The experimental results show that the speech synthesis attack can be easily recognized by SR systems and humans after applying our defense.

REFERENCES

[1] 2019. Resemblyzer. https://github.com/resemble-ai/Resemblyzer.

[2] 2019. The Wall Street Journal. https://www.wsj.com/articles/fraudsters-use-ai-to-mimic-ceos-voice-in-unusual-cybercrime-case-11567157402.

[3] 2021. Forbes. https://www.forbes.com/sites/thomasbrewster/2021/10/14/huge-bank-fraud-uses-deep-fake-voice-tech-to-steal-millions/?sh=2fdf0c417559.

[4] 2021. vc-github. https://github.com/cyhuang-tw/attack-vc.

[5] 2023. Alexa. https://www.amazon.com/gp/help/customer/display.html?nodeId=GYCXKY2AB2QWZT2X.

[6] 2023. audible. https://www.audible.com/.

[7] 2023. Azure. https://azure.microsoft.com/en-us/services/cognitive-services/speaker-recognition/.

[8] 2023. Mturk. https://www.mturk.com/.

[9] 2023. soundcloud. https://soundcloud.com.

[10] 2023. speechify. https://www.speechify.com/.

[11] 2023. WeChat. https://help.wechat.com/cgi-bin/micromsg-bin/oshelpcenter?opcode=2&plat=ios&lang=en&id=150819uqYnUR150819YzINVb.

[12] Muhammad Ejaz Ahmed, Il-Youp Kwak, Jun Ho Huh, Iljoo Kim, Taekkyung Oh, and Hyoungshick Kim. 2020. Void: A fast and light voice liveness detection system. In *29th USENIX Security Symposium (USENIX Security 20)*. 2685–2702.

[13] Ehab A AlBadawy, Siwei Lyu, and Hany Farid. 2019. Detecting AI-Synthesized Speech Using Bispectral Analysis.. In *CVPR Workshops*. 104–109.

[14] S Abhishek Anand, Jian Liu, Chen Wang, Maliheh Shirvanian, Nitesh Saxena, and Yingying Chen. 2021. Echovib: Exploring voice authentication via unique non-linear vibrations of short replayed speech. In *Proceedings of the 2021 ACM Asia Conference on Computer and Communications Security*. 67–81.

[15] Gopala Krishna Anumanchipalli, Kishore Prahallad, and Alan W Black. 2011. Festvox: Tools for creation and analyses of large speech corpora. In *Workshop on Very Large Scale Phonetics Research, UPenn, Philadelphia*, Vol. 70.

[16] Sercan Arik, Jitong Chen, Kainan Peng, Wei Ping, and Yanqi Zhou. 2018. Neural voice cloning with a few samples. *Advances in Neural Information Processing Systems* 31 (2018).

[17] James Bergstra, Rémi Bardenet, Yoshua Bengio, and Balázs Kégl. 2011. Algorithms for hyper-parameter optimization. *Advances in neural information processing systems* 24 (2011).

[18] Logan Blue, Kevin Warren, Hadi Abdullah, Cassidy Gibson, Luis Vargas, Jessica O'Dell, Kevin Butler, and Patrick Traynor. 2022. Who Are You (I Really Wanna Know)? Detecting Audio {DeepFakes} Through Vocal Tract Reconstruction. In *31st USENIX Security Symposium (USENIX Security 22)*. 2691–2708.

[19] Edresson Casanova, Julian Weber, Christopher D Shulby, Arnaldo Candido Junior, Eren Gölge, and Moacir A Ponti. 2022. Yourtts: Towards zero-shot multi-speaker tts and zero-shot voice conversion for everyone. In *International Conference on Machine Learning*. PMLR, 2709–2720.

[20] Si Chen, Kui Ren, Sixu Piao, Cong Wang, Qian Wang, Jian Weng, Lu Su, and Aziz Mohaisen. 2017. You can hear but you cannot steal: Defending against voice impersonation attacks on smartphones. In *2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS)*. IEEE, 183–195.

[21] Janelle H Cheung, Deanna K Burns, Robert R Sinclair, and Michael Sliter. 2017. Amazon Mechanical Turk in organizational psychology: An evaluation and practical recommendations. *Journal of Business and Psychology* 32 (2017), 347–361.

[22] Ju-chieh Chou, Cheng-chieh Yeh, and Hung-yi Lee. 2019. One-shot voice conversion by separating speaker and content representations with instance normalization. *arXiv preprint arXiv:1904.05742* (2019).

[23] Jiangyi Deng, Yanjiao Chen, Yinan Zhong, Qianhao Miao, Xueluan Gong, and Wenyuan Xu. 2023. Catch You and I Can: Revealing Source Voiceprint Against Voice Conversion. *arXiv preprint arXiv:2302.12434* (2023).

[24] Thierry Dutoit. 1997. *An introduction to text-to-speech synthesis*. Vol. 3. Springer Science & Business Media.

[25] Grant Fairbanks. 1960. The rainbow passage. *Voice and articulation drillbook* 2 (1960), 127–127.

[26] Joel Frank and Lea Schönherr. 2021. Wavefake: A data set to facilitate audio deepfake detection. *arXiv preprint arXiv:2111.02813* (2021).

[27] Haichang Gao, Honggang Liu, Dan Yao, Xiyang Liu, and Uwe Aickelin. 2010. An audio CAPTCHA to distinguish humans from computers. In *2010 Third International Symposium on Electronic Commerce and Security*. IEEE, 265–269.

[28] Daniel Griffin and Jae Lim. 1984. Signal estimation from modified short-time Fourier transform. *IEEE Transactions on acoustics, speech, and signal processing* 32, 2 (1984), 236–243.

[29] Wei-Ning Hsu, Yu Zhang, Ron J Weiss, Heiga Zen, Yonghui Wu, Yuxuan Wang, Yuan Cao, Ye Jia, Zhifeng Chen, Jonathan Shen, et al. 2018. Hierarchical generative modeling for controllable speech synthesis. *arXiv preprint arXiv:1810.07217* (2018).

[30] Qiong Hu, Erik Marchi, David Winarsky, Yannis Stylianou, Devang Naik, and Sachin Kajarekar. 2019. Neural text-to-speech adaptation from low quality public recordings. In *Speech Synthesis Workshop*, Vol. 10.

[31] Chien-yu Huang, Yist Y Lin, Hung-yi Lee, and Lin-shan Lee. 2021. Defending your voice: Adversarial attack on voice conversion. In *2021 IEEE Spoken Language Technology Workshop (SLT)*. IEEE, 552–559.

[32] Tzu-hsien Huang, Jheng-hao Lin, and Hung-yi Lee. 2021. How far are we from robust voice conversion: A survey. In *2021 IEEE Spoken Language Technology Workshop (SLT)*. IEEE, 514–521.

[33] Jesin James, BT Balamurali, Catherine I Watson, and Bruce MacDonald. 2021. Empathetic speech synthesis and testing for healthcare robots. *International Journal of Social Robotics* 13 (2021), 2119–2137.

[34] Corentin Jemine et al. 2019. Master thesis: Real-time voice cloning. (2019).

[35] Ye Jia, Ron J Weiss, Fadi Biadsy, Wolfgang Macherey, Melvin Johnson, Zhifeng Chen, and Yonghui Wu. 2019. Direct speech-to-speech translation with a sequence-to-sequence model. *arXiv preprint arXiv:1904.06037* (2019).

[36] Ye Jia, Yu Zhang, Ron Weiss, Quan Wang, Jonathan Shen, Fei Ren, Patrick Nguyen, Ruoming Pang, Ignacio Lopez Moreno, Yonghui Wu, et al. 2018. Transfer learning from speaker verification to multispeaker text-to-speech synthesis. *Advances in neural information processing systems* 31 (2018).

[37] Hirokazu Kameoka, Takuhiro Kaneko, Kou Tanaka, and Nobukatsu Hojo. 2018. Stargan-vc: Non-parallel many-to-many voice conversion using star generative adversarial networks. In *2018 IEEE Spoken Language Technology Workshop (SLT)*. IEEE, 266–273.

[38] Pierre Lanchantin, Gilles Degottex, and Xavier Rodet. 2010. A HMM-based speech synthesis system using a new glottal source and vocal-tract separation method. In *2010 IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE, 4630–4633.

[39] Alessandro Lieto, Daniele Moro, Francesco Devoti, Claudia Parera, Vincenzo Lipari, Paolo Bestagini, and Stefano Tubaro. 2019. " Hello? Who Am I Talking to?" A Shallow CNN Approach for Human vs. Bot Speech Classification. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2577–2581.

[40] Jheng-hao Lin, Yist Y Lin, Chung-Ming Chien, and Hung-yi Lee. 2021. S2VC: A Framework for Any-to-Any Voice Conversion with Self-Supervised Pretrained Representations. *arXiv preprint arXiv:2104.02901* (2021).

[41] Yist Y Lin, Chung-Ming Chien, Jheng-Hao Lin, Hung-yi Lee, and Lin-shan Lee. 2021. Fragmentvc: Any-to-any voice conversion by end-to-end extracting and fusing fine-grained voice fragments with attention. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 5939–5943.

[42] Takashi Masuko, Takafumi Hitotsumatsu, Keiichi Tokuda, and Takao Kobayashi. 1999. On the security of HMM-based speaker verification systems against imposture using synthetic speech. In *Sixth European conference on speech communication and technology*.

[43] Jiří Mertl, Eva Žáčková, and Barbora Řepová. 2018. Quality of life of patients after total laryngectomy: the struggle against stigmatization and social exclusion using speech synthesis. *Disability and Rehabilitation: Assistive Technology* 13, 4 (2018), 342–352.

[44] Dibya Mukhopadhyay, Maliheh Shirvanian, and Nitesh Saxena. 2015. All your voices are belong to us: Stealing voices to fool humans and machines. In *Computer Security–ESORICS 2015: 20th European Symposium on Research in Computer Security, Vienna, Austria, September 21-25, 2015, Proceedings, Part II 20*. Springer, 599–621.

[45] John Mullennix and Steven Stern. 2010. *Computer Synthesized Speech Technologies: Tools for Aiding Impairment: Tools for Aiding Impairment*. IGI Global.

[46] Arsha Nagrani, Joon Son Chung, and Andrew Zisserman. 2017. Voxceleb: a large-scale speaker identification dataset. *arXiv preprint arXiv:1706.08612* (2017).

[47] Satoshi Nakamura. 2009. *Overcoming the language barrier with speech translation technology*. Technical Report. Citeseer.

[48] 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. *arXiv preprint arXiv:1609.03499* (2016).

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

[50] Sharbani Pandit, Jienan Liu, Roberto Perdisci, and Mustaque Ahamad. 2020. Fighting Voice Spam with a Virtual Assistant Prototype. *arXiv preprint arXiv:2008.03554* (2020).

[51] Wei Ping, Kainan Peng, Andrew Gibiansky, Sercan Ömer Arik, Ajay Kannan, Sharan Narang, Jonathan Raiman, and John Miller. 2017. Deep Voice 3: 2000-Speaker Neural Text-to-Speech. (2017).

[52] Kaizhi Qian, Yang Zhang, Shiyu Chang, Xuesong Yang, and Mark Hasegawa-Johnson. 2019. Autovc: Zero-shot voice style transfer with only autoencoder loss. In *International Conference on Machine Learning*. PMLR, 5210–5219.

[53] Yi Ren, Yangjun Ruan, Xu Tan, Tao Qin, Sheng Zhao, Zhou Zhao, and Tie-Yan Liu. 2019. Fastspeech: Fast, robust and controllable text to speech. *Advances in Neural Information Processing Systems* 32 (2019).

[54] Faysal Hossain Shezan, Hang Hu, Jiamin Wang, Gang Wang, and Yuan Tian. 2020. Read between the lines: An empirical measurement of sensitive applications

of voice personal assistant systems. In *Proceedings of the Web Conference 2020*. 1006–1017.

[55] Cong Shi, Xiangyu Xu, Tianfang Zhang, Payton Walker, Yi Wu, Jian Liu, Nitesh Saxena, Yingying Chen, and Jiadi Yu. 2021. Face-Mic: inferring live speech and speaker identity via subtle facial dynamics captured by AR/VR motion sensors. In *Proceedings of the 27th Annual International Conference on Mobile Computing and Networking*. 478–490.

[56] Maliheh Shirvanian, Manar Mohammed, Nitesh Saxena, and S Abhishek Anand. 2020. Voicefox: Leveraging inbuilt transcription to enhance the security of machine-human speaker verification against voice synthesis attacks. In *Annual Computer Security Applications Conference*. 870–883.

[57] Xu Tan, Jiawei Chen, Haohe Liu, Jian Cong, Chen Zhang, Yanqing Liu, Xi Wang, Yichong Leng, Yuanhao Yi, Lei He, et al. 2022. Naturalspeech: End-to-end text to speech synthesis with human-level quality. *arXiv preprint arXiv:2205.04421* (2022).

[58] Keiichi Tokuda, Yoshihiko Nankaku, Tomoki Toda, Heiga Zen, Junichi Yamagishi, and Keiichiro Oura. 2013. Speech synthesis based on hidden Markov models. *Proc. IEEE* 101, 5 (2013), 1234–1252.

[59] Christophe Veaux, Junichi Yamagishi, Kirsten MacDonald, et al. 2017. CSTR VCTK corpus: English multi-speaker corpus for CSTR voice cloning toolkit. *University of Edinburgh. The Centre for Speech Technology Research (CSTR)* (2017).

[60] Payton Walker and Nitesh Saxena. 2021. SoK: assessing the threat potential of vibration-based attacks against live speech using mobile sensors. In *Proceedings of the 14th ACM Conference on Security and Privacy in Wireless and Mobile Networks*. 273–287.

[61] Li Wan, Quan Wang, Alan Papir, and Ignacio Lopez Moreno. 2018. Generalized end-to-end loss for speaker verification. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 4879–4883.

[62] Chen Wang, Cong Shi, Yingying Chen, Yan Wang, and Nitesh Saxena. 2020. WearID: Wearable-assisted low-effort authentication to voice assistants using cross-domain speech similarity. *arXiv preprint arXiv:2003.09083* (2020).

[63] Qian Wang, Xiu Lin, Man Zhou, Yanjiao Chen, Cong Wang, Qi Li, and Xiangyang Luo. 2019. Voicepop: A pop noise based anti-spoofing system for voice authentication on smartphones. In *IEEE INFOCOM 2019-IEEE Conference on Computer Communications*. IEEE, 2062–2070.

[64] Run Wang, Felix Juefei-Xu, Yihao Huang, Qing Guo, Xiaofei Xie, Lei Ma, and Yang Liu. 2020. Deepsonar: Towards effective and robust detection of ai-synthesized fake voices. In *Proceedings of the 28th ACM international conference on multimedia*. 1207–1216.

[65] Shu Wang, Jiahao Cao, Xu He, Kun Sun, and Qi Li. 2020. When the differences in frequency domain are compensated: Understanding and defeating modulated replay attacks on automatic speech recognition. In *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*. 1103–1119.

[66] Yuxuan Wang, RJ Skerry-Ryan, Daisy Stanton, Yonghui Wu, Ron J Weiss, Navdeep Jaitly, Zongheng Yang, Ying Xiao, Zhifeng Chen, Samy Bengio, et al. 2017. Tacotron: Towards end-to-end speech synthesis. *arXiv preprint arXiv:1703.10135* (2017).

[67] Emily Wenger, Max Bronckers, Christian Cianfarani, Jenna Cryan, Angela Sha, Haitao Zheng, and Ben Y Zhao. 2021. " Hello, It's Me": Deep Learning-based Speech Synthesis Attacks in the Real World. In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*. 235–251.

[68] Chen Yan, Yan Long, Xiaoyu Ji, and Wenyuan Xu. 2019. The catcher in the field: A fieldprint based spoofing detection for text-independent speaker verification. In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*. 1215–1229.

[69] Linghan Zhang, Sheng Tan, and Jie Yang. 2017. Hearing your voice is not enough: An articulatory gesture based liveness detection for voice authentication. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*. 57–71.

[70] Linghan Zhang, Sheng Tan, Jie Yang, and Yingying Chen. 2016. Voicelive: A phoneme localization based liveness detection for voice authentication on smartphones. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*. 1080–1091.

[71] Yangyong Zhang, Lei Xu, Abner Mendoza, Guangliang Yang, Phakpoom Chinprutthiwong, and Guofei Gu. 2019. Life after speech recognition: Fuzzing semantic misinterpretation for voice assistant applications. In *Proc. of the Network and Distributed System Security Symposium (NDSS'19)*.

8 APPENDIX

8.1 Other Experimental Settings

In the case study in Section 4.1 , we evaluate both Gaussian and Laplace distribution noise, and they show similar effective rates. We thus use Gaussian noise to generate AN-Mask for the remaining experiments. To evaluate whether the proposed modification methods are effective on a frequency window, we set the value of τ_1 to 0.05. The values of τ_2 for Zero Mask, AN-Mask, and GB-Mask are set to 0.12, 0.20, and 0.20, respectively.

In Section 5.4, we match the perturbation constraint ϵ used in Attack-VC with τ_d by generating enough (1000) defense samples with different values of ϵ and then calculating the average sample distortion. Figure 10 shows the values of ϵ and τ_d that can generate similar distortions on speech samples.

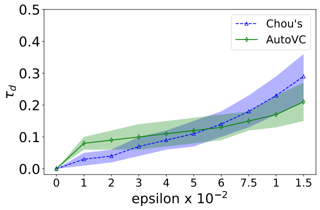


Figure 10: The relationship between ϵ and τ_d.

According to Figure 10, we match $\epsilon = 0.03$ with $\tau_d = 0.06$, ϵ=0.055 with $\tau_d = 0.12$, and ϵ=0.075 with $\tau_d = 0.18$ for Chou's Model. For AutoVC, we match $\epsilon = 0.01$ with $\tau_d = 0.06$, ϵ=0.05 with $\tau_d = 0.12$, and ϵ=0.13 with $\tau_d = 0.18$. We also study whether the defense samples derived in Section 5.4 meet the matching requirement. Table 8 provides the average sample distortion in Section 5.4. We can see that Attack-VC and SpeakerMask can generate similar distortions based on the assigned values of ϵ. Thus, the comparison between Attack-VC and SpeakerMask is under a fair condition.

Table 8: Sample distortion.

	Chou's		AutoVC		SV2TTS
	Attack-VC	SpecMask	Attack-VC	SpecMask	SpecMask
$\tau = 0.06$	0.06 ± 0.02	0.06 ± 0.01	0.08 ± 0.03	0.06 ± 0.01	0.06 ± 0.01
$\tau = 0.12$	0.13 ± 0.04	0.12 ± 0.01	0.11 ± 0.04	0.12 ± 0.01	0.12 ± 0.01
$\tau = 0.18$	0.18 ± 0.05	0.18 ± 0.01	0.19 ± 0.06	0.18 ± 0.01	0.18 ± 0.01

8.2 Phrases for Speech Synthesis

All participants involved in Section 5.4 are asked to read the below rainbow passage containing 20 phrases. The first 10 phrases are used for determining their personal defense strategies. To demonstrate generalizability of our scheme, we apply the defense strategy to the remaining 10 phrases. The following is the passage read by them.

When the sunlight strikes raindrops in the air, they act as a prism and form a rainbow. The rainbow is a division of white light into many beautiful colors. These take the shape of a long round arch, with its path high above, and its two ends apparently beyond the horizon. There is , according to legend, a boiling pot of gold at one end. People look, but no one ever finds it. When a man looks for something beyond his reach, his friends say he is looking for the pot of gold at the end of the rainbow. Throughout the centuries people have explained the rainbow in various ways. Some have accepted it as a miracle without physical explanation. To the Hebrews it was a token that there would be no more universal floods. The Greeks used to imagine that it was a sign from the gods to foretell war or heavy rain. The Norsemen considered the rainbow as a bridge over which the gods passed from earth to their home in the sky.

Table 9: Phrases used for SV2TTS speech synthesis on Resemblyzer and Azure.

We control complexity by establishing new languages for describing a design, each of which emphasizes particular aspects of the design and deemphasizes others.
An interpreter raises the machine to the level of the user program.
Everything should be made as simple as possible, and no simpler.
The great dividing line between success and failurecan be expressed in five words: "I did not have time."
When your enemy is making a very serious mistake,don't be impolite and disturb him.
A charlatan makes obscure what is clear; a thinker makes clear what is obscure.
There are two ways of constructing a software design; one way is to make it so simple that there are obviously no deficiencies, and the other way is to make it so complicated that there are noobvious deficiencies.
The three chief virtues of a programmer are: Laziness, Impatience and Hubris.
All non-trivial abstractions, to some degree, are leaky.
XML wasn't designed to be edited by humans on a regular basis.

Others have tried to explain the phenomenon physically. Aristotle thought that the rainbow was caused by reflection of the sun's rays by the rain. Since then physicists have found that it is not reflection, but refraction by the raindrops which causes the rainbows. Many complicated ideas about the rainbow have been formed. The difference in the rainbow depends considerably upon the size of the drops, and the width of the colored band increases as the size of the drops increases. The actual primary rainbow observed is said to be the effect of super-imposition of a number of bows. If the red of the second bow falls upon the green of the first, the result is to give a bow with an abnormally wide yellow band, since red and green light when mixed form yellow. This is a very common type of bow, one showing mainly red and yellow, with little or no green or blue.

For all speech synthesis in SV2TTS, we adopt the same 10 phrases used in [67] for providing linguistic information, as is shown in Table 9.

8.3 Analysis of the Defense Strategy

We also analyze which frequency-strategy pairs are the most effective (appear most often) in the derived speaker-level defense strategies. Specifically, after deriving the defense strategies for the speakers in the VCTK dataset, we count the number of each frequency block and modification method pair's occurrences. Then, we calculate the appearance rate (percentage) of each pair and show the top 6 pairs in Table 10.

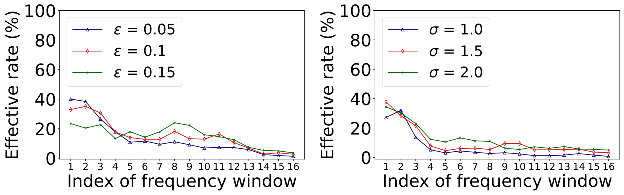
We can see that Zero Mask is highly effective on Chou's model. AN-Mask and GB-Mask are both effective on AutoVC and SV2TTS. In addition, most of the top pairs contain large block numbers, which means modifying high frequencies (above 4000 Hz) can result in better defense effects in most cases. Those high frequencies are usually less perceptible to human beings.

8.4 Impact of Parameters in AN-Mask and GB-Mask

We next study the impact of noise magnitude constraint (i.e., ϵ) in AN-Mask and the standard deviation (i.e., σ) in GB-Mask on the defense effect. Specifically, we conduct a case study with SV2TTS and test the effective rate (defined in Section 4.1). The results in Figure 11 indicate that a smaller noise magnitude constraint in AN-Mask and a smaller standard deviation in GB-Mask typically result in reduced effective rates. Conversely, larger values of these parameters tend to increase the effective rates. However, higher values of these parameters may introduce noticeable background noise, potentially compromising the usability of the speech.

Table 10: Top 6 pairs and their appearance rates (%).

	Chou's		AutoVC		SV2TTS	
Rank	Pair	Rate	Pair	Rate	Pair	Rate
1	$(b_{15},\mathcal{M}_Z)$	14.4	$(b_{13},\mathcal{M}_Z)$	13.7	$(b_{16},\mathcal{M}_Z)$	6.7
2	$(b_{16},\mathcal{M}_Z)$	13.8	$(b_{14},\mathcal{M}_Z)$	10.1	$(b_{14},\mathcal{M}_Z)$	6.5
3	$(b_{14},\mathcal{M}_Z)$	11.6	$(b_{13},\mathcal{M}_{AN})$	9.9	$(b_{16},\mathcal{M}_{AN})$	6.5
4	$(b_6,\mathcal{M}_Z)$	10.2	$(b_{12},\mathcal{M}_Z)$	6.3	$(b_{15},\mathcal{M}_Z)$	5.9
5	$(b_7,\mathcal{M}_Z)$	9.6	$(b_{12},\mathcal{M}_{AN})$	6.2	$(b_{15},\mathcal{M}_{AN})$	5.9
6	$(b_{13},\mathcal{M}_Z)$	8.4	$(b_{14},\mathcal{M}_{GB})$	6.1	$(b_{13},\mathcal{M}_{GB})$	5.8

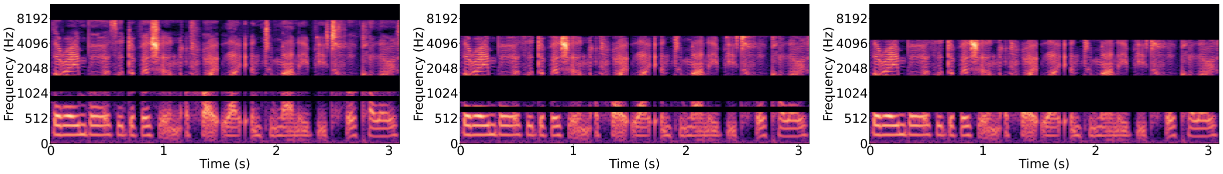


(a) Impact of ϵ in AN-Mask **(b) Impact of σ in GB-Mask**

Figure 11: Impact of parameters in AN-Mask and GB-Mask.

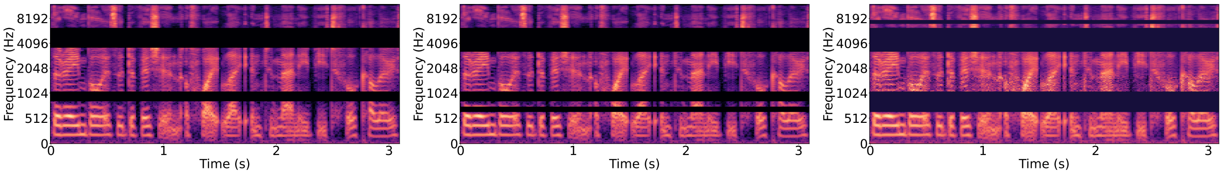
8.5 Visualization

We further provide some examples of the spectrogram generated by SpeakerMask based on the three speech synthesis models. Here We randomly select two speakers' samples for demonstration. The speaker-sample ID are shown in the following figures.



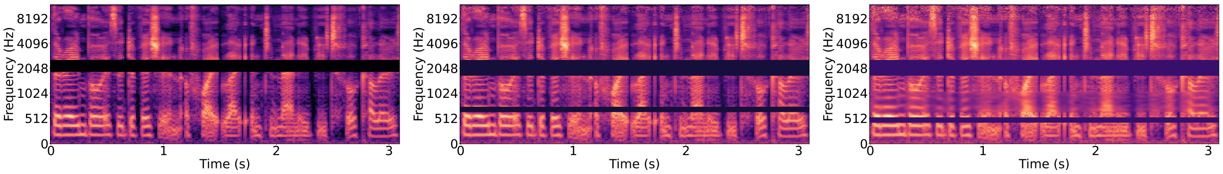
(a) $\tau_d = 0.06$ (b) $\tau_d = 0.12$ (c) $\tau_d = 0.18$

Figure 12: Speaker-sample ID: p249-007 - Chou' defense samples.



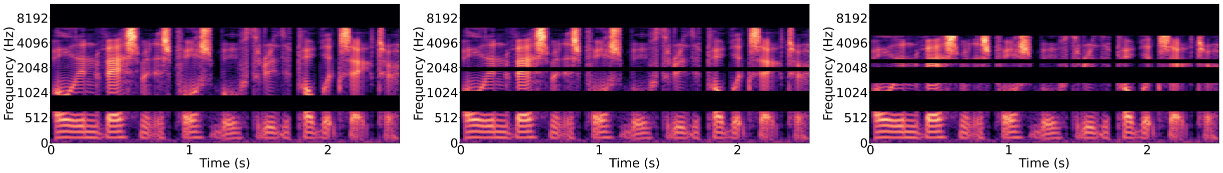
(a) $\tau_d = 0.06$ (b) $\tau_d = 0.12$ (c) $\tau_d = 0.18$

Figure 13: Speaker-sample ID: p249-007 - AutoVC defense samples.



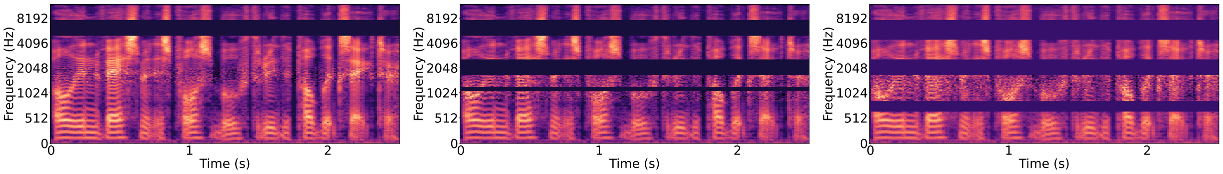
(a) $\tau_d = 0.06$ (b) $\tau_d = 0.12$ (c) $\tau_d = 0.18$

Figure 14: Speaker-sample ID: p249-007 - SV2TTS defense samples.



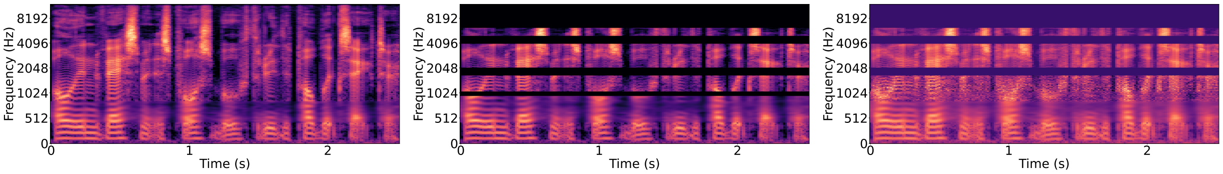
(a) $\tau_d = 0.06$ (b) $\tau_d = 0.12$ (c) $\tau_d = 0.18$

Figure 15: Speaker-sample ID: p275-360 - Chou's defense samples.



(a) $\tau_d = 0.06$ (b) $\tau_d = 0.12$ (c) $\tau_d = 0.18$

Figure 16: Speaker-sample ID: p275-360 - AutoVC defense samples.



(a) $\tau_d = 0.06$ (b) $\tau_d = 0.12$ (c) $\tau_d = 0.18$

Figure 17: Speaker-sample ID: p275-360 - SV2TTS defense samples.