Catch Me If You Can: Blackbox Adversarial Attacks on Automatic Speech Recognition using Frequency Masking

Catch Me If You Can: Blackbox Adversarial Attacks on Automatic Speech Recognition using Frequency Masking

***Abstract*—Automatic speech recognition (ASR) models are used widely in applications for voice navigation and voice control of domestic appliances. ASRs have been misused by attackers to generate malicious outputs by attacking the deep learning component within ASRs. To assess the security and robustnesss of ASRs, we propose techniques within our framework SPAT that generate blackbox (agnostic to the DNN) adversarial attacks that are portable across ASRs. This is in contrast to existing work that focuses on whitebox attacks that are time consuming and lack portability.**

Our techniques generate adversarial attacks that have no human audible difference by manipulating the input speech signal using a psychoacoustic model that maintains the audio perturbations below the thresholds of human perception. We propose a framework SPAT with three attack generation techniques based on the psychoacoustic concept and frame selection techniques to selectively target the attack. We evaluate portability and effectiveness of our techniques using three popular ASRs and two input audio datasets using the metrics - `Word Error Rate (WER)` of output transcription, `Similarity` to original audio, attack `Success Rate` on different ASRs and `Detection score` by a defense system. We found our adversarial attacks were portable across ASRs, not easily detected by a state-of-the-art defense system, and had significant difference in output transcriptions while sounding similar to original audio.

***Index Terms*—Automatic Speech Recognition, Adversarial Attack, Blackbox, Frequency Masking**

I. Introduction

Automatic speech recognition models (ASRs) are widely used in a variety of applications, such as mobile virtual assistants (Siri, Google Assistant), in-vehicle voice navigation and voice smart home appliances like Alexa and Google Home with built-in voice assistants. Owing to the prevalence of ASRs in our daily lives, their security and integrity are of paramount concern. The computational core of ASRs are deep neural networks (DNNs) that have been shown to be susceptible to adversarial perturbations; easily misused by attackers to generate malicious outputs [18], [21], [34].

a) Existing work on ASR adversarial attacks.: Adversarial perturbations[1] were first presented by Szegedy et al. to demonstrate the lack of robustness in DNN models – a small perturbation of an input may lead to a significant perturbation of the output of a DNN model [27]. This vulnerability can be exploited by adversaries to augment the original input with a crafted perturbation, invisible to a human but sufficient for the DNN model to misclassify this input. This influential work triggered several research contributions in the computer vision domain that generate adversarial attacks for testing security and robustness of vision tasks [10], [15], [19]. Research on the use of adversarial attacks on ASRs is, however, only just emerging, and can be classified along two dimensions,

1. Un-targeted or Targeted The aim of un-targeted adversarial audio is to make an ASR model incorrectly transcribe speech while sounding similar to original input, while the aim of targeted adversarial attack is to cause an ASR model to output a specific transcription (target) injected by an adversary. This paper focuses on un-targeted adversarial attack.

2. Whitebox or Blackbox Threat Model In a whitebox threat model, the adversary assumes knowledge of the internal structure of the ASR model, while in a blackbox threat model, the adversary can only probe the ASR with input audio and analyze the resulting transcription. We use a blackbox threat model.

Most existing methods [7], [8], [23], [32] for ASR adversarial attack generation are *targeted and whitebox*. These methods suffer from one or more of the following drawbacks (1) Whitebox assumption is not practical and lacks portability since commercial ASR application developers do not typically reveal the internal workings of their systems, (2) time taken to generate attacks is considerable and cannot be used in real-time. , and (3) poor quality audio in attacks makes them easily detectable by defense techniques like [8], [20]. Existing few methods [4], [29] for *blackbox, targeted* attacks suffer from the drawback of intractable number of queries to the ASR, that are time-consuming and impractical. *Blackbox untargeted* attacks that do not rely on the knowledge of the internal NN structure or queries to the ASR would address the above limitations and the only known technique was proposed by Abdullah et al. in 2020 [1]. To create adversarial audio, they decompose the original audio and remove components with low-amplitude that they believe will not affect audio comprehension. Although interesting, their approach does not strive to ensure the adversarial and original audio sound similar. Additionally, difference achieved in transcribed texts is not measured or reported. We found the ability of their attacks in bypassing a state of the art defense system was not effective.

b) Proposed Attack Generation: We propose a blackbox un-targeted attack generation approach that is faster, more portable across ASRs, and robust to a state-of-the-art defense than Abdullah et al. Our framework, SPAT, for attacking ASRs uses a psychoacoustics concept called frequency masking that determines how sounds interfere and mask each other. We manipulate masked (or inaudible) components of the original audio in such a way that their spectral density is different but they remain masked. Such a manipulation ensures the adversarial attack is indistinguishable from the original but has the potential to change the resulting transcription. We propose

[1]Also referred to as Adversarial examples or Adversarial attacks.

three attack generation approaches centered around this idea – `Griffin Lim Reconstruction (GL)`,`Original Phase (OP)` and `Deletion (DE)`. Additionally, to help increase similarity to the original audio, we provide the option of selectively introducing perturbations to a small fraction of audio frames rather than all of them. The SPAT framework provides three frame selection options – `Random`, `Important` and `All`. Among them, the `Important` option identifies the frames that cause the most change to output text when set to zero and we then introduce perturbations to just these important frames.

We evaluate SPAT on three different ASRs – Deepspeech [12], Sphinx [16] and Google cloud speech-to-text API, using two different input audio datasets – Librispeech [22] and Commonvoice [5]. We assess the effectiveness of our approaches for attack generation and frame selection using the metrics - `WER`, `Similarity`, attack `Success Rate` and `Detection score`. We also compare SPAT with a targeted whitebox state-of-the-art (SOTA) method [8] and an untargeted blackbox SOTA method [1]. It is worth noting that the scale of our evaluation is much bigger than existing work [1], [8], [23] as we use different audio datasets and ASRs. We find our approach using `OP` or `DE` for attack generation combined with `Important` or `All` frame selection was effective at attacking all three ASRs. Our techniques were $312\times$ faster than the whitebox targeted SOTA, and $7\times$ faster than blackbox targeted SOTA method. The defense system, Waveguard [14], was less effective at detecting attacks generated with our techniques compared with the other two SOTA methods.

In summary, the contributions in this paper are as follows:

1) A novel approach and framework, SPAT, for untargeted blackbox adversarial attack generation on ASRs based on frequency masking.
2) Frame selection option to selectively perturb frames in an audio.
3) Extensive empirical evaluation of the attack generation and frame selection options within SPAT on three ASRs and two audio datasets. We also compare performance against SOTA whitebox and blackbox techniques.

The source code for SPAT can be found at:
https://anonymous.4open.science/r/lalalala-9DEE.

II. Background

Most current ASRs comprise the following stages when transcribing an input audio to a text output: 1. Pre-processing to remove noise and detect human voice in the input audio, 2. Signal processing stage to extract audio features as Mel Frequency Cepstral Coefficient (MFCC) and 3. Recurrent Neural Network prediction that uses the MFCC features from the audio to predict a probability distribution of characters for every time step or audio frame. From the character sequence distributions, an output selection algorithm, such as Beam search, is used to select the most likely translated text. More details on the stages can be found in [3]. We next present a brief description of the frequency masking concept used in SPAT.

A. Frequency Masking and Masking Threshold Computation

Frequency masking is a psychoacoustic phenomenon that occurs when the perception of a sound is affected and masked by the presence of another sound, distracting the ear from being able to clearly perceive the simultaneous sounds [17]. For example, on a quiet night, consider that the sound of chirping crickets is audible but in the presence of the TV sound, we stop hearing the crickets chirping as the TV sound masks it. In Figure 1, the TV sound would be the *masker* (seen as a red bar) that creates a masking threshold [17] which is the minimum level at which other sounds in the same frequency frame can be heard. The chirping sound of the crickets falls below the masking threshold (seen as a blue bar) and therefore is not audible in the presence of the TV. The chirping sound in Figure 1 would be the *maskee*.

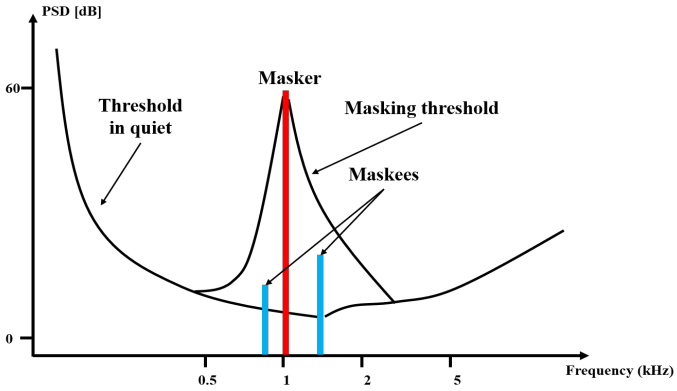

Fig. 1. Frequency masking phenomenon: the masker creates a *masking threshold* in the nearby frequency domain such that other sounds below this threshold cannot be heard.

a) Masking Threshold Computation: To calculate the masking threshold for a given audio, we need to first convert the audio from the expression in the time domain to the frequency domain (using Fast Fourier Transform), then discard the phase information in the spectrum. We then use the amplitude information of the spectrum to calculate the log-magnitude power spectral density (PSD) of this audio. The PSD characterizes the energy distribution on a unit frequency, and is used widely to describe the frequency domain results of the signal [17], [31]. The red and blue bars in Figure 1 represent the PSD (in dB) of maskers and maskees, respectively, for the given frequency bin. According to [17], [23], maskers are identified from the audio PSD using two conditions: the PSD of a masker should be greater than the absolute threshold of hearing (ATH), and it must be the highest PSD estimate within a certain surrounding frequency range. After identifying the maskers, their respective masking thresholds will be computed using a two-slope function, described in [31]. If there are several maskers and associated masking thresholds, they will be combined into a global masking threshold for the audio like in [23]. Once the maskers are identified, the other PSDs in the audio are labelled maskees. A more detailed description of the computation of masker, maskee and masking threshold can be found in [23], [31].

We use this masking phenomenon observed with simultaneous sounds to create adversarial audio that sounds similar to the original audio but has the potential to produce a different transcription. We achieve this by first taking the original audio that is composed of many sounds, identifying the maskers and maskees in it using the approach from [23], [31] (red and

blue bars in Figure 1). We then manipulate the PSD of the maskees so it stays below the masking threshold, ensuring they are not audible, like in the original audio. Nevertheless, this manipulation can still affect the transcribed text. We create the adversarial audio by composing together the unchanged maskers and manipulated maskees. In terms of our earlier example with the TV sound and crickets chirping, we identify the TV sound as the masker and the chirping crickets as the maskee. We then manipulate the PSD of the cricket sound, staying within the masking threshold, to produce an adversarial audio that composes the TV sound with the manipulated chirping sound. Section III describes the SPAT framework and the techniques used for manipulation in detail.

B. Griffin-Lim Algorithm

To construct an adversarial audio from the maskers and manipulated maskees in the amplitude spectrum, we use the Griffin-Lim (GL) algorithm that helps reconstruct audio waveforms with a known amplitude spectrum but an unknown phase spectrum [11]. Steps in the algorithm are as follows: (1) Randomly initialize a phase spectrum, (2) Use this phase spectrum and the known amplitude spectrum to synthesize a new waveform through Inverse Short-Time Fourier Transform (3) Use the synthesized speech to get new amplitude spectrum and new phase spectrum through Short-time Fourier Transform, (4) Discard the new amplitude spectrum, (5) Repeat steps 2, 3, 4 for a fixed number of iterations. Output is a waveform with an estimated phase spectrum and the known input amplitude spectrum.

III. Methodology

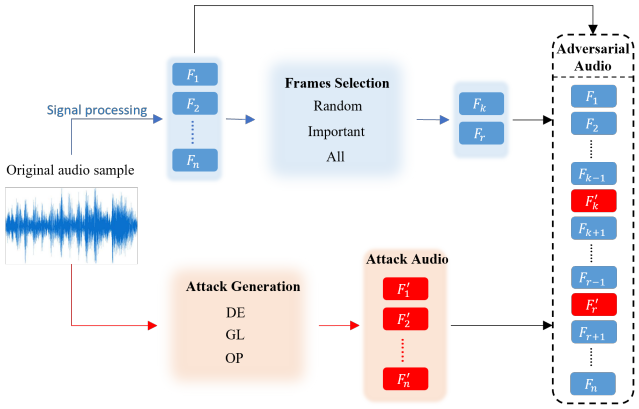

Fig. 2. Our framework, SPAT, for generating adversarial attacks comprises of three stages, 1. Frame Selection, 2. Attack generation and finally 3. Adversarial audio formed by combining information in the first two stages.

In this section, we propose techniques for generating adversarial attacks for ASRs. As seen in Figure 2, our framework, SPAT, has two important stages, 1. Audio Frame Selection and 2. Attack Generation. The general workflow in SPAT is as follows: Given an input audio example, we first select frames within it using one of the three techniques for audio frame selection – `Random`, `Important` and `All`. Independently, we generate manipulated audio from the input audio using one of three attack techniques – `GL Reconstruction (GL)`, `Original Phase (OP)`, `Deletion (DE)`. We then replace the selected frames in the original audio with corresponding manipulated audio frames while keeping the rest of the audio unchanged. The combination of original and manipulated audio frames forms the adversarial attack audio.

a) Threat Model and Assumptions: The attack techniques in SPAT assume a black-box threat model, in which an adversary has no knowledge of the internal workings or architecture of the target ASR model. We treat the ASR as a black-box to which we make requests in the form of input audio and receive responses in the form of transcriptions in text format. We also assume that an adversary can only make a limited number of requests to the target ASR. We also accommodate the scenario when the adversary cannot make any requests to the target ASR (with `All` frames selected). Finally, we assume an over the line attack. This means that digital files are sent directly to the target ASR system for transcription, as opposed to playing back audio files over the air through speakers.

A. Stage 1: Frame Selection

We explore generation of adversarial audio by modifying a subset of frames in the entire audio. We provide three approaches to select audio frames that will be later manipulated – `Random`, `Important` and `All`. We will start by describing the technique to select `Important` frames.

1) Important:: The rationale for selecting important frames is to restrict manipulation to a small number of significant frames. This allows the adversarial audio to remain similar to the original while still affecting the output transcription text. We define importance of frames based on the proportion of Word Error Rate (`WER`[2]) produced by masking that frame in the original audio. The steps involved in selecting important frames are as follows,

1. For every input audio example, record output transcription.
2. Pick one of the input audio examples. For every frame in the processed audio example, set it to zero (masked) while keeping the remaining frames unchanged. Record translated text using the ASR for the masked audio.
3. Compute `WER` between the masked and original output. Repeat this for all frames. The frames that result in a non-zero `WER` are identified as important frames for that audio example. Magnitude of `WER` change for frame selection can be altered to suit needs.
4. Repeat Steps 2 and 3 for the remaining input audio examples.

At the end of this process, every input audio example is associated with a list of important frames.

2) Random:: To enable us to compare the effectiveness of only using important frames in frame selection, we also provide a means to select frames randomly. The number of frames selected for a given audio example is set to be the same as the number of important frames in that audio.

3) All:: We simply use *all* the frames from the manipulated audio generated in Stage 2 (see Section III-B). Using `All` frames helps us assess how much `WER` was achievable. In addition it helps quantify the tradeoff in `WER` and `Similarity` when compared to frame selection with `Important` and `Random`. It is worth noting that using `All` frames requires *no* queries to the ASR. Therefore, if the threat model assumes

[2]`WER` is a common metric to evaluate the difference in ASR transcription between original versus adversarial audio. The formula is provided in Sec IV-B

no queries then we would select `All` frames in Stage 1 of our approach.

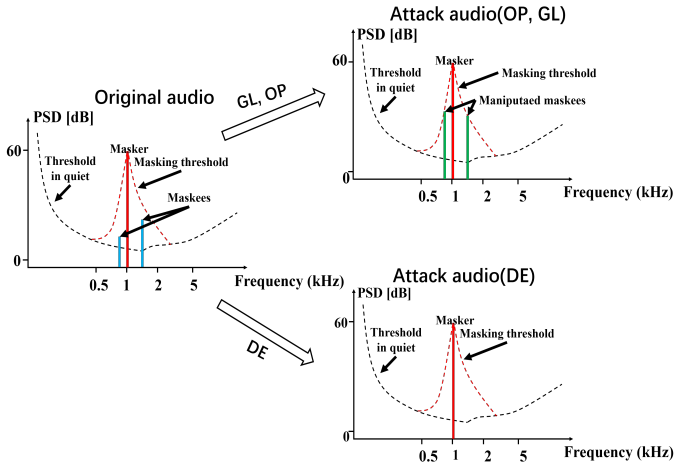

Fig. 3. Attack generation methods, GL and OP, increase the PSD of maskees to the masking threshold. Attack generation with DE suppresses the PSD of maskees to zero.

B. Stage 2: Attack Generation

We discuss three attack generation techniques within SPAT – GL, OP and DE, that manipulate the amplitude spectrum of the input audio example using the concept of frequency masking, described in Section II-A. We illustrate the manipulations in Figure 3 and describe them in the Sections below. All three techniques take the input audio, generate audio frames in the frequency domain (obtained with sampling and fast fourier transform), with each frame having amplitude and phase information. For each frame, we compute the masking threshold, maskers and maskees using established techniques discussed in Section II-A

1) GL Reconstruction (GL): As seen in the top part of Figure 3, `GL` (and `OP`) increases the PSD of all maskees (blue bars in the original audio) to the global masking threshold. Masker PSDs remain unchanged. We then compute an updated amplitude based on the maskers and altered maskees PSD inversely [31][3]. `GL` discards phase information of the input audio waveform. Instead, it estimates phase information using the GL reconstruction technique discussed in Section II-B. The estimated phase information is combined with the updated amplitude information and is used to synthesize the attack audio through inverse FFT.

2) Original Phase (OP): The primary difference between the `OP` and `GL` technique is in the phase information. Estimating phase using the GL algorithm introduces distortion and lack of consistency across multiple runs. To avoid this problem, the `OP` technique retains phase information from the original audio. We believe using phase information from the original audio to synthesize the attack audio will make it more similar to the original audio.

[3] $Amplitude(k) = N\sqrt{10^{\frac{PSD(k))}{10}}}$, where k is the index of the frequency bin and N represents the length of frame.

3) Deletion (DE): Previous methods, `OP` and `GL`, ensure the attack audio sounds no different from the original input by increasing the PSD of the maskees up to the maximum limit (which is the masking threshold) for them to remain masked. The `DE` technique, on the other hand, suppresses the PSD of the maskees to the minimum value of zero which is akin to deleting them. This manipulation will not affect the audio perception as the masking threshold is unaffected. The `DE` technique, thus, deletes all maskee PSDs that are hidden under the masking threshold. Subsequently, we use the modified amplitude after deletion and combine it with the *original phase* information from the input audio (similar to `OP`'s use of phase). We use inverse FFT as before to synthesize attack audio from the amplitude and phase information.

C. Stage 3: Combining Original and Attack Audio

In this final stage, we create an adversarial attack by taking the original audio, replacing the selected frames (identified in Stage 1) with corresponding frames from the attack audio (generated in Stage 2). Other frames from the original audio are left unchanged. This modified version of the original serves as an adversarial attack.

The source code for our adversarial attack generation framework, SPAT, with the three attack generation and three frame selection methods, can be found at https://anonymous.4open.science/r/lalalala-9DEE.

IV. Experiments

We evaluate the effectiveness of our techniques within SPAT, described in Section III, using two different datasets – (1) 200 audio samples from Librispeech [22] and (2) 200 audio samples from Commonvoice [5]. We use three ASRs in our evaluation, namely, Deepspeech [12], Sphinx [16], and Google ASR. Our choice of datasets and ASRs were inspired by their use in related work for adversarial ASR attack generation [1] [8] [23] [35]. We discuss the defense system used to assess the effectiveness of the adversarial attacks, evaluation metrics and the research questions in our experiments in the rest of this Section.

A. Detection and defense

The ability to evade defense systems is an important measure of effectiveness for adversarial attacks. Defense systems have evolved to detect and defend a significant fraction of adversarial attacks. In our experiments, we use a SOTA adversarial audio detection and defense system, Waveguard [14], proposed by Hussain et al. in 2021. We chose Waveguard as our defense system as it is demonstrated to be faster, more effective and capable of detecting both targeted and untargeted attacks compared to existing detection techniques, like Temporal Dependency Detection Method [33]. We report how well Waveguard performed (as an AUC score) in detecting adversarial attacks in our experiments.

Attack detection with Waveguard is divided into two steps. The first step is to transform the input audio using one of several functions that are meant to preserve (or closely preserve) the transcription text. For example, one of their transformations – Mel Spectrogram Extraction and Inversion – first extracts MFCC features from input audio and reconstructs the audio from MFCC features. The second step is to compare the Character Error Rate(CER) between the transcription text for

the original and transformed audio. If the difference between the texts is greater than a predefined threshold, then the input audio is classified as adversarial, and benign otherwise.

B. Evaluation Metrics

We use four metrics to measure the effectiveness of our techniques – `Word Error Rate (WER)`, `Similarity`, `Success Rate` and `Detection score`. We are interested in generating adversarial attacks that sound similar to the the original audio (high `Similarity`) but produce a transcription different from the original (high `WER`). Additionally, we would like the technique to be portable, i.e generate adversarial attacks that are usable across several ASRs (high `Success Rate`). Finally, we want the generated attacks to be robust to get past SOTA defense systems, like Waveguard [14] (lower `Detection score`). We provide definitions of each of these metrics below.

a) `WER`: is a common metric to evaluate the difference in ASR transcription from original versus adversarial audio [9] [13]. `WER` is computed using Equation (1),

$$\text{WER} = \frac{\text{Insertions} + \text{Substitutions} + \text{Deletions}}{\text{Total Words in Correct Transcript}} \quad (1)$$

b) Similarity: We use the widely used `PESQ` metric [24] that measures quality of audio relative to a reference audio to assess similarity of adversarial audio to the original. The `PESQ` algorithm accepts a noisy signal, which in our case is the adversarial attack, and an original reference signal, which is the input audio for our method. The `PESQ` score ranges from -0.5 to 4.5. The higher the score, the better the voice quality. According to [6], audio quality is deemed "good" when its `PESQ` score is above 3.0. We use this standard for classifying the quality of the adversarial audio. In this paper, we use `Similarity` metric to mean the `PESQ` score.

c) `Success Rate`: shown in Equation (2), refers to the ratio of adversarial attacks that can successfully attack a given ASR. A successful attack, as defined by Abdullah et al [1], happens when the adversarial attack results in a non-zero WER with respect to the original transcription.

$$\text{Success Rate} = \frac{\text{Number of successful attacks}}{\text{Total number of adversarial attacks}} \quad (2)$$

d) `Detection score`: refers to the effectiveness of the Waveguard defense system in correctly classifying adversarial attacks. We use the area under the curve (AUC) metric, reported by Waveguard [14], to evaluate correct classification of adversarial attacks. The AUC score ranges from 0.0 to 1.0. We aim for a lower Waveguard AUC score or `Detection score` with our techniques.

C. Research Questions

We aim to answer the following research questions (RQs) in our experiments,

RQ1: Which frame selection method in SPAT among `Random`, `Important`, `All` performs best?
We compare the `WER` and `Similarity` achieved by the different frame selection techniques across three different ASRs and two input audio datasets. Answering this research question will help us assess the value of selecting a subset of frames versus just changing the whole audio.

RQ2: Which attack generation technique among `GL`, `OP`, `DE` performs best?
We compare the `WER`, `Similarity` achieved by the different attack generation techniques across three different ASRs and two different input datasets. We also measure `Time` taken by each technique.

RQ3: Are the adversarial attacks portable across ASRs?
One of the primary selling points of our techniques is that they are blackbox and untargeted, and therefore agnostic to the structure and workings within ASRs. We validate this by evaluating the `Success Rate` of the generated adversarial attacks across three different ASRs.

RQ4: Do SPAT generated attacks perform better than SOTA techniques?
We selected representative and high-performing SOTAs in our comparison, namely a whitebox targeted technique proposed by Carlini et al [8], and a blackbox technique by Abdullah et al [1].

Carlini et al. generate adversarial attacks using Deepspeech ASR and the Commonvoice input dataset. To allow comparison, we use the same ASR and input dataset with our techniques. Owing to the targeted nature of their technique, they require the transcription text to be specified in advance. To address this need, we use the transcription from Deepspeech ASR with adversarial attacks generated by our technique as Carlini et al.'s target. We then compare our technique with Carlini et al. with respect to time taken to generate adversarial attacks, `Similarity` to original audio, `Success Rate` on other ASRs, Google and Sphinx, and `Detection score`. Since the transcription text in both techniques are the same, it is not useful to compare `WER`.

We compare our technique against Abdullah et al. using `WER`, `Similarity`, `Success Rate`, `Detection Score`, `Time` over different ASRs and both the Commonvoice and Librispeech dataset.

a) Experiment settings: We use Google Colab Pro with two NVIDIA Tesla T4 GPUs(16GB RAM, 2560 cores) to run our experiments. We use the following audio parameters in our experiments: Sampling rate of $16000HZ$, frame length of 2048 and frame shift of 512.

V. Results and Analysis

We present and discuss the results from our experiments in the context of the research questions presented earlier. It is worth noting that `WER` and `Similarity` are measured for each attack, while `Success rate` and `Detection score` are measured across an entire dataset. Techniques should try to maximise `WER`, `Similarity` and `Success rate` while minimising `Detection score` by Waveguard.

A. RQ1: Comparison of Frame Selection Techniques

The best performing frame selection technique is one that achieves high `WER` and high `Similarity` across audio examples. However, these two metrics are often conflicting. We discuss and compare `WER` and `Similarity` achieved by the three frame selection techniques in SPAT below. Figures in Table I shows the `WER` achieved by different frame section techniques for the Librispeech and Commonvoice datasets across different ASRs and attack generation techniques while Figure 4 shows the `Similarity` achieved.

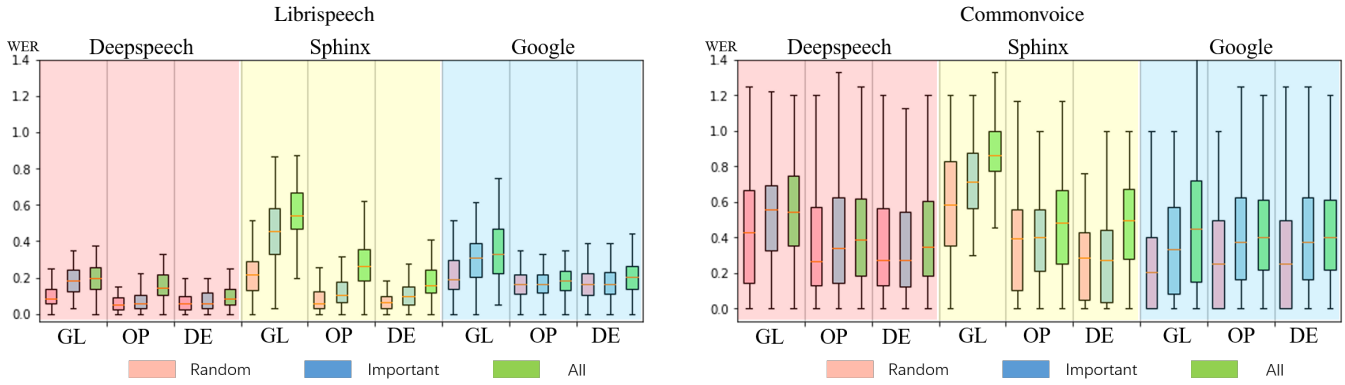

TABLE I
BOX PLOTS OF THE WER OF THE ADVERSARIAL ATTACKS GENERATED WITH TWO DIFFERENT DATASETS.

	Librispeech			Commonvoice		
	GL	OP	DE	GL	OP	DE
Deepspeech	96%	95%	91%	95%	90%	90%
Sphinx	99%	96.5%	94%	98%	89%	90%
Google	99%	97.5%	95.5%	85%	80%	80%
Average	98%	96.3%	93.5%	92%	86.3%	86.6%

TABLE II
THE SUCCESS RATES OF THE ADVERSARIAL ATTACKS WITH GL, OP, DE ATTACK GENERATION METHODS ACROSS THE THREE ASRS AND TWO DATASETS.ALL FRAMES IS USED AS THE FRAME SELECTION METHOD.

Technique	Time	Similarity	Success rate			WER			Detection score
			Deepspeech	Sphinx	Google	Deepspeech	Sphinx	Google	
Carlini [8]	780 seconds	3.63	N/A	77%	33%	N/A	N/A	N/A	0.67
Abdullah [1]	18 seconds	3.12	80%	77%	54%	0.39	0.44	0.14	0.65
OP	3.5 seconds	3.65	90%	89%	80%	0.46	0.47	0.40	0.52
DE	2.5 seconds	4.29	90%	90%	80%	0.45	0.50	0.38	0.55

TABLE III
COMPARISON OF OP, DE WITH ABDULLAH ET AL. [1] AND CARLINI ET AL. [8] WITH RESPECT TO GENERATION TIME FOR PER ADVERSARIAL ATTACK, SIMILARITY TO ORIGINAL AUDIO EXAMPLES,WER, SUCCESS RATE AND DETECTION SCORE AGAINST DEFENSE SYSTEM [14] IN ATTACKING ALL THREE ASRS

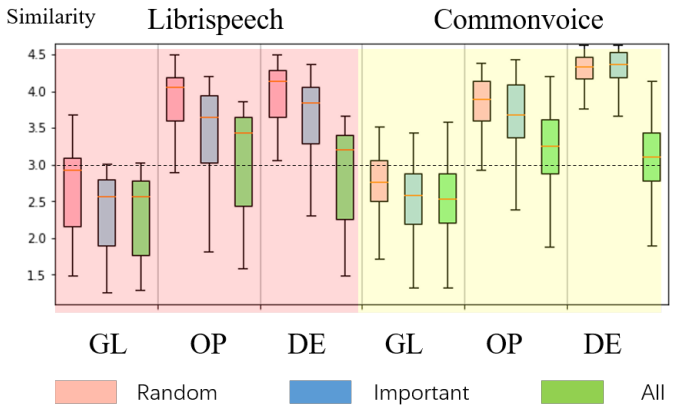

Fig. 4. Box plots of the Similarity of the adversarial attacks generated with all datasets.

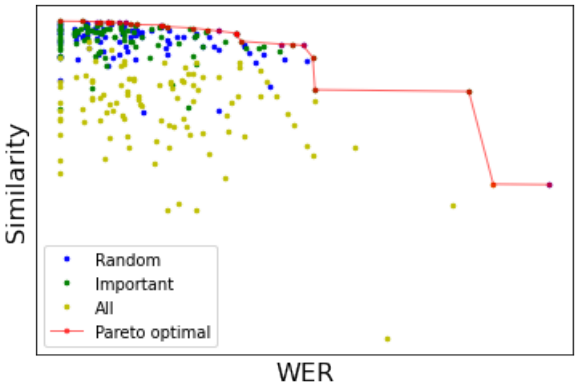

Fig. 5. Pareto front over adversarial attacks generated by Random, Important and All frame selection techniques on Commonvoice dataset and Deepspeech ASR using DE.

a) All frames: We find in Table I and Figure 4, that the All frame selection achieves the highest WER and lowest Similarity compared to Important and Random across ASRs, input datasets and attack generation methods. This is in line with our expectations as the other two frame selection techniques select a small part of the audio to introduce noise into achieving lower WER but higher Similarity to original audio.

b) Important versus Random:: For most combinations of ASR, dataset and attack generation, we find Random frame selection produces the lowest WER and the highest Similarity, while Important frame selection results in

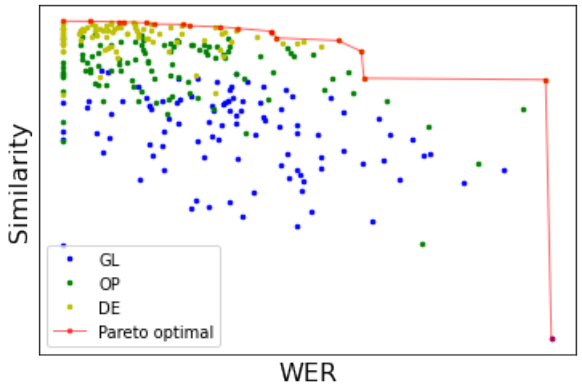

Fig. 6. Pareto front over adversarial attacks generated by GL, OP and DE on Commonvoice dataset and Deepspeech ASR using Important frames.

a WER and Similarity between Random and All.

c) Statistical Analysis.: We confirmed the statistical significance (at 5% significance level) of the difference in means between the frame selection techniques using one-way Anova and did a post-hoc Tukey's Honest Significant Difference (HSD) test to reveal which differences between pairs of means are significant. Extension file[4] Sections 1.1.1 and 1.1.2 list the P-values for pairwise comparisons of WERs and Similarities between frame selection techniques. For the WER metric, we find the All frames selection technology is significantly better than Important and Random on majority of ASR, dataset, attack technique combinations. In contrast, for Similarity measure, Random and Important frame selections significantly outperformed All.

d) Pareto front: Owing to the conflicting nature of the WER and Similarity metrics, all three frame selection techniques achieve a trade-off between them. We use the Pareto front with these two metrics, shown in Figure 5 for one of the datasets and ASRs, to determine the number of non-dominated attack examples (that fall on the Pareto front) from each frame selection. We find Important frame selection has the most number of non-dominated attacks (17 examples); Random was second with 12 examples, while All frames only had 1 non-dominated attack example. This trend is observed across all ASRs, attack technologies and datasets (see results in Extension file Section 1.1.3). Based on the number of non-dominated examples, we believe that Important frames is effective at achieving a trade-off between WER and Similarity.

e) Summary: In terms of WER, we find All frames performs best. However, Important and Random frames perform better in terms of Similarity. We find Important is the best at optimising trade-off between the two metrics, achieving reasonable performance in both WER and Similarity.

B. RQ2: Comparison of Attack Generation Techniques

We present WER achieved by GL, OP, DE using different ASRs and datasets in Table I, while we show Similarity achieved in Figure 4. Best performing attack generation technique is one that results in a high WER and high Similarity to original audio.

a) WER Performance: GL attack generation performs better than both OP and DE in terms of WER achieved. We confirm the differences are significant using One-way Anova and Tukey's HSD test (see P-values in Section 1.2.1 of the Extension file). Between OP and DE attacks, OP outperforms DE with DeepSpeech and Sphinx ASRs over the Librispeech dataset. There is no significant difference between the two techniques over the other dataset and ASRs.

b) Similarity Performance: Both OP and DE significantly outperform GL in terms of Similarity, confirmed with pairwise comparison using one-way Anova followed by Tukey's HSD test (P-value tables in Extension file Section 1.2.2). The median Similarity or PESQ score for GL tends to be below the value of 3.0 (shown by the dashed line), irrespective of frame selection used. According to Beuran et al. [6], the standard for good quality audio is a PESQ score of greater than 3 and GL technique does not meet this standard in our experiments. We believe this is because GL uses estimated, rather than actual, phase information which causes distortion that reduces the PESQ score.

Between OP and DE, there is no significant difference in their Similarity performance. The benefit with using DE lies in faster generation of an adversarial attack. The average time to generate a single adversarial attack using DE is $2.5 seconds$, a second faster than the OP technique ($3.5 seconds$ on average) as OP relies on calculating the masking threshold for every input example.

c) Pareto Front: As with RQ1, we draw the Pareto front using WER and Similarity, shown in Figure 6. For Deepspeech ASR using Important frames in Figure 6, we find DE technique has the most number of non-dominated attacks (16 examples); OP is second with 5 examples, while GL only has 2 non-dominated attack example. This trend is observed across most ASRs, frame selections and datasets. However, for the All frame selection, OP has the most number of non-dominated attacks on Deepspeech and Google (Results available in Section 1.1.3 of the Extension file).

d) Summary: Based on the number of non-dominated examples, we believe that when using Important and Random frame selection, DE is a suitable choice for optimising both WER and Similarity. When using All frame selection, OP is a better choice. Independently, DE is the fastest attack generation among the three techniques.

C. RQ3: Portability across ASRs

We evaluate portability of the adversarial attacks generated by OP,GL,DE across the three ASRs using the Success Rate metric, described in Section IV-B. Table II presents Success Rates achieved with the Librispeech and Commonvoice datasets.

We find GL achieves the best success rates over all ASRs, with both the Librispeech dataset (average of 98%) and the Commonvoice dataset (average of 92%). OP comes next, performing better than DE on the Librispeech dataset (96% versus 93.5%, respectively). OP and DE have similar performance over the Commonvoice dataset (average of 86%).

[4]Extension file is available at https://anonymous.4open.science/r/lalalala-9DEE/apsec2022_extension.pdf

a) Summary: All three attack generation techniques have high success rates across the three ASRs producing portable adversarial attacks. GL outperforms OP and DE in portability but the magnitude of difference is small (on average 2% to 5%). OP and DE have comparable performance on the ASRs, especially with the Commonvoice dataset.

D. RQ4: Comparison to Existing Techniques

We compare performance of SPAT against a whitebox targeted technique proposed by Carlini et al. [8] and a blackbox untargeted technique proposed by Abdullah et al. [1] using the metrics – WER, Similarity, Success rate, Time, Detection score. Within SPAT, we use OP and DE for attack generation as they perform best in terms of Similarity and WER[5]

1) Comparison with Carlini et al: We fix the ASR to Deepspeech and input dataset to Commonvoice to match the experiments in Carlini et al. [8]. We show results in Table III. We do not compare WER as the target text for Carlini et al. [8] is the transcription text from our adversarial attacks, so there will be no difference.

a) Time and Similarity: We find time taken to generate attack examples is faster with our approaches, OP and DE, compared to Carlini et al. with a maximum speedup of 312× achieved with DE. We also achieve higher Similarity scores – 4.3 (DE) and 3.7 (OP), compared to 3.6 by Carlini et al. We confirm the statistical significance (at 5% significance level) of the observed differences in Similarity using one-way Anova and Tukey's Honest Significant Difference (HSD) test. We find our techniques are a clear winner in terms of time taken, and outperform Carlini at al. in Similarity.

b) Success Rate: To evaluate portability of adversarial attacks, we transcribe the adversarial attacks using Google and Sphinx (since DeepSpeech is used by Carlini et al.). We find when used with Google ASR, adversarial attacks generated by Carlini et al. have a much lower Success Rate than our techniques (33% versus 80%), respectively. For Sphinx, the difference in Success Rate is smaller but the trend remains (77% Carlini versus 89% to 90% for ours). The lower Success Rate observed with Carlini et al. is because their technique specifically targets the neural network inside Deepspeech, and may not be as effective when used on other ASRs with different NNs. This is a drawback also encountered with other whitebox attacks. However, since our method is blackbox, we find it is easier to port our adversarial attacks to different ASRs.

c) Detection score: Attack examples generated by Carlini et al. are more easily detected by Waveguard, with a higher Detection score score of 0.67, compared to techniques in SPAT, whose Detection scores are 0.52 for OP and 0.55 for DE. We believe this is because Carlini et al use noise in their attack generation which is detected more easily by Waveguard. We find SPAT attack generation with OP and DE performs better than Carlini et al at evading the Waveguard defense.

Across all four evaluation metrics, we find one of the two techniques from SPAT is the winner (highlighted in red in Table III), outperforming Carlini et al [8] across all metrics.

[5]We use the best performance between All and Important frames.

2) Comparison with Abdullah et al: Like SPAT, Abdullah et al. [1] use a blackbox, untargeted attack generation technique that is meant to be fast and portable on different ASRs. Unlike the comparison with Carlini et al., we can include WER as a performance metric (in addition to the other 4 metrics) and Deepspeech ASR in our comparison. We discuss performance for each of the metrics below using the Commonvoice dataset[6].

a) Time and Similarity: We find attack generation with OP and DE is much faster than Abdullah et al. (5× and 7× faster, respectively). For the Similarity metric, SPAT outperforms Abdullah et al. with both OP and DE attack generation (at 5% significance level, P-value tables in the Extension file.)

b) Success rate, WER and Detection score: Attack examples generated with OP and DE have a higher Success rate than Abdullah et al. across all ASRs. We see a similar trend with WER, where OP and DE outperform Abdullah et al. (at 5% statistical significance). Finally, OP and DE surpass Abdullah et al. with respect to getting past Waveguard's defense system by achieving lower detection scores of 0.52 and 0.55, respectively, versus 0.65 for Abdullah et al..

In summary, we find our attack techniques, OP and DE, surpass Abdullah et al. for each of the five evaluation metrics (best performing is highlighted in red in Table III).

3) Threats to Validity: There are three threats to validity in our experiments based on the selected ASRs, speech datasets and the metrics used in evaluation.

Firstly, we only use three ASRs among dozens of commercial and non-commercial ASRs in our experiments to evaluate the effectiveness of our attacks. Results may vary on other ASRs. However, our techniques are meant to be ASR agnostic so we believe they will be applicable to other ASRs. It is worth noting that the number of ASRs in our experiments is at par or exceeds that used in Section VI. We plan on conducting a more extensive evaluation in the future.

Secondly, we use audio samples from two common speech datasets – Librispeech [22] and Commonvoice [5]. The adversarial examples we generate are a manipulation of the input audio. It would be interesting to evaluate our technique on audio samples in other speech datasets. Given the time consuming nature of the experiments, the number of samples from the different attack generation techniques and their combination with frame selection techniques, we were unable to scale our experiments further.

Thirdly, we use metrics WER, PESQ score for Similarity, Success Rate and attack Detection Score in our evaluation of adversarial examples. These metrics have been used separately in other related work [1], [9], [13], [24] which led to their selection in our experiments. We have also tried Cosine Similarity of MFCC features in place of PESQ score and the trends were similar between the different techniques in SPAT. The choice of metrics for evaluating adversarial examples in this field have not been standardised and there is a range of metrics across several papers. We have tried our best to capture several metrics in our evaluation to avoid bias along any one dimension.

Attack Type	Existing work
Whitebox-Targeted	Vaidya et al. [30], Carlini et al, [7], [8], Qin et al. [23],Yuan et al. [34],Yakura et al. [32], Schönherr et al. [25], [26], Szurley et al. [28]
Blackbox-Targeted	Zhang et al. [35], Alzantot et al. [4], Taori et al. [29]
Blackbox-Untargeted	Abdullah et al. [1]

TABLE IV
EXISTING WORK ON ADVERSARIAL ASR ATTACK GENERATIONS.

VI. RELATED WORK

As mentioned in Section I, existing adversarial attack generation on ASR models can be classified along two dimensions: 1. Targeted for a given transcription or untargeted, and 2. Whitebox, with knowledge of the internal ASR structure or Blackbox. Table IV lists the existing techniques using these two dimensions and they are discussed in more detail in the rest of this Section.

A. Targeted Attacks

Vaidya et al. [30] pioneered the first whitebox targeted method for attacking ASR in 2015. Given the transcription to target, they gradually approach the target by continuously fine-tuning the parameters of the extracted MFCC features. Once the goal is reached, they use the obtained adversarial MFCC features to reconstruct the speech waveform. On the basis of Vaidya's work and in an effort to improve the efficiency of their approach, Carlini et al. [7] proposed Hidden Voice Command in 2016, adding noise that is often encountered in real life. However, neither of these two types of attacks can conceal the existence of noise, and such adversarial attacks can be easily detected as noise rather than effective commands.

Yuan et al. [34] proposed a method for embedding commands into songs so that when these songs are played, the commands will be translated by an ASR. Additionally, they improve the realistic nature of adversarial attacks by introducing noise generated by hardware devices. This approach, however, is restricted to songs as the carrier of commands, and is, therefore, limited in application scenarios.

Carlini et al. [8] in 2018 used a whitebox approach that applies gradient descent to modify the original audio so that the difference between the transcription and the target text is smaller. Their experimental results show their attack `Success Rates` reached 100% on Deepspeech ASR. However, their approach faces the following drawbacks: First, it can take up to several hours to generate attacks; second, the gradient descent method requires the attacker to have a good understanding of all the internal parameters and structures of the attacked system before it can be used; and finally the adversarial attacks generated will be invalid over other ASRs.

Yakura et al. [32] proposed some improvements to [8] to maintain attack performance under over-the-air conditions (mixed with sound of the surrounding environment). They generate adversarial attacks accounting for noise caused by echo and recording in real life, so as to obtain more robust adversarial attacks. However, other shortcomings in Carlini et al. [8] (such as long generation time and weak transferability) have not been addressed.

[6]Results for Librispeech dataset follow a similar trend and can be viewed in Extension file Section 1.3.2.

In 2018, Schönherr et al. [26] developed a whitebox approach that applies the knowledge of masking threshold to generate adversarial attacks. They proposed to limit the generated noise below the masking threshold of the original audio to ensure that the obtained perturbation is not audible to the human ear. In more recent work [25], they introduced room impulse response (RIR) simulator to improve the robustness of examples that produces different types of noise for different environment configurations.

Inspired by Schönherr et al., Qin and Carlini et al. [23] developed a whitebox method and optimized perturbations to make it lower than the masking threshold of the original audio. This method achieved a 100% attack `Success Rate` on the Lingvo system. However, their algorithms only study the attack on traditional signal-processing-based ASRs, and has not studied the end-to-end ASRs that have emerged in recent years. Adbullah et al [2] adapted the algorithm for end-to-end ASRs on the basis of those two. But again, it's also a targeted white-box attack, and those limitations are still unresolved.

Like other whitebox targeted approaches, their work lacks portability to other ASRs and is time consuming for attack generation.

Around the same time, Szurley et al. [28] proposed a white-box method similar to Schönherr et al. [25], [26] and Carlini et al. [8], [23] that constructed an optimization based on masking threshold and combined it with room reverberation. Their method reached a 100% `Success Rate` on Deepspeech but still suffers from limitations of lack of portability and time consuming attack generation.

a) Blackbox-targeted approaches: Few Blackbox Targeted adversarial attack generation techniques exist in the literature [4], [29], [35]. Zhang et al. [35] in 2017 modulated the voice on the ultrasonic carrier to insert preset commands(like "Open the window") into the original audio. However, this method is not easy to reproduce as it uses hardware characteristics of the microphone to complete the attack. Alzantot et al. [4] proposed a iterative optimization method that adds a small amount of noise iteratively to a benign example until the ASR outputs a target label. Taori et al. [29] used a genetic algorithm to achieve iterative optimization, mutating benign examples until the ASR output matches a target label. These approaches for blackbox targeted attacks suffer from the following two weaknesses: First, they require thousands of queries to ASRs to generate one adversarial attack, which is unrealistic. Secondly, these attacks are only applicable to ASRs that aim to classify audios, not translate audios.

B. Untargeted Attacks

The only known untargeted blackbox adversarial ASR attack generation approach is that proposed by Abdullah et al. [1] in 2019. They construct an adversarial attack by decomposing and reconstructing the original audio. Specifically, they decompose the original audio into components called eigenvectors via Singular Spectrum Analysis (SSA). These eigenvectors represent the various trends and noises that make up the audio. They believe that eigenvectors with smaller eigenvalues convey limited information. They choose a threshold to classify eigenvalues as small and subsequently eliminate small eigenvectors. They then reconstruct an audio from the remaining components as the adversarial attack. We

compare performance of our techniques against their approach in Section V-D.

VII. Conclusion

We proposed a blackbox untargeted adversarial attack generation technique for ASRs using frequency masking to make the adversarial audio sound similar to the original while producing a change in the transcription. Our framework, SPAT, provides three attack generation options – `GL`, `OP` and `DE`. We also provide the option of selectively introducing perturbations to a small fraction of audio frames using three frame selection options — `Random`, `Important` and `All`. Evaluation of our techniques over three ASRs and two audio datasets showed that our techniques can be effective at achieving high `WERs` (average of 44% with `OP+All`) while also achieving high `Similarity` (average of 3.93 with `OP+Important`). The choice in attack generation and frame selection helps achieve a good balance between these two metrics, with `DE` attack generation and `Important` frames achieving the best trade-off. We also confirmed that our techniques were portable across ASRs and superior to existing whitebox targeted technique [8] and blackbox untargeted technique [1] in terms of `WER`, `Similarity`, `Success Rate`, `Time` and `Detection score`.

References

[1] Hadi Abdullah, Muhammad Sajidur Rahman, Washington Garcia, Logan Blue, Kevin Warren, Anurag Swarnim Yadav, Tom Shrimpton, and Patrick Traynor. Hear "no evil", see "kenansville": Efficient and transferable black-box attacks on speech recognition and voice identification systems, 2019.

[2] Hadi Abdullah, Muhammad Sajidur Rahman, Christian Peeters, Cassidy Gibson, Washington Garcia, Vincent Bindschaedler, Thomas Shrimpton, and Patrick Traynor. Beyond l_p clipping: Equalization based psychoacoustic attacks against ASRs. In Vineeth N. Balasubramanian and Ivor Tsang, editors, *Proceedings of The 13th Asian Conference on Machine Learning*, volume 157 of *Proceedings of Machine Learning Research*, pages 672–688. PMLR, 17–19 Nov 2021.

[3] Hadi Abdullah, Kevin Warren, Vincent Bindschaedler, Nicolas Papernot, and Patrick Traynor. Sok: The faults in our asrs: An overview of attacks against automatic speech recognition and speaker identification systems. *CoRR*, abs/2007.06622, 2020.

[4] Moustafa Alzantot, Bharathan Balaji, and Mani B. Srivastava. Did you hear that? adversarial examples against automatic speech recognition. *CoRR*, abs/1801.00554, 2018.

[5] R. Ardila, M. Branson, K. Davis, M. Henretty, M. Kohler, J. Meyer, R. Morais, L. Saunders, F. M. Tyers, and G. Weber. Common voice: A massively-multilingual speech corpus. In *Proceedings of the 12th Conference on Language Resources and Evaluation (LREC 2020)*, pages 4211–4215, 2020.

[6] Razvan Beuran, Mihail Ivanovici, and Bob Dobinson. Ntwork quality of service measurement system for application requirements evaluation. In *International Symposium on Performance Evaluation of Computer and Telecommunication Systems, SPECTS'03*, pages 380–387, 2003.

[7] Nicholas Carlini, Pratyush Mishra, Tavish Vaidya, Yuankai Zhang, Micah Sherr, Clay Shields, David Wagner, and Wenchao Zhou. Hidden voice commands. In *25th USENIX Security Symposium (USENIX Security 16)*, pages 513–530, Austin, TX, August 2016. USENIX Association.

[8] Nicholas Carlini and David A. Wagner. Audio adversarial examples: Targeted attacks on speech-to-text. *CoRR*, abs/1801.01944, 2018.

[9] Cristina España-Bonet and José A. R. Fonollosa. Automatic speech recognition with deep neural networks for impaired speech. In Alberto Abad, Alfonso Ortega, António Teixeira, Carmen García Mateo, Carlos D. Martínez Hinarejos, Fernando Perdigão, Fernando Batista, and Nuno Mamede, editors, *Advances in Speech and Language Technologies for Iberian Languages*, pages 97–107, Cham, 2016. Springer International Publishing.

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

[11] D. Griffin and Jae Lim. Signal estimation from modified short-time fourier transform. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 32(2):236–243, 1984.

[12] Awni Hannun, Carl Case, Jared Casper, Bryan Catanzaro, Greg Diamos, Erich Elsen, Ryan Prenger, Sanjeev Satheesh, Shubho Sengupta, Adam Coates, and Andrew Y. Ng. Deep speech: Scaling up end-to-end speech recognition, 2014.

[13] Jean-Paul Haton. Automatic speech recognition: A review. In Olivier Camp, Joaquim B. L. Filipe, Slimane Hammoudi, and Mario Piattini, editors, *Enterprise Information Systems V*, pages 6–11, Dordrecht, 2005. Springer Netherlands.

[14] Shehzeen Hussain, Paarth Neekhara, Shlomo Dubnov, Julian J. McAuley, and Farinaz Koushanfar. Waveguard: Understanding and mitigating audio adversarial examples. *CoRR*, abs/2103.03344, 2021.

[15] Alexey Kurakin, Ian Goodfellow, Samy Bengio, et al. Adversarial examples in the physical world, 2016.

[16] Paul Lamere, Philip Kwok, William Walker, Evandro B Gouvêa, Rita Singh, Bhiksha Raj, and Peter Wolf. Design of the cmu sphinx-4 decoder. In *Interspeech*. Citeseer, 2003.

[17] Yiqing Lin and Waleed H. Abdulla. *Principles of Psychoacoustics*, pages 15–49. Springer International Publishing, Cham, 2015.

[18] Sapna Maheshwari. Burger king 'ok google'ad doesn't seem ok with google. *The New York Times. https://www. nytimes. com/2017/04/12/business/burger-king-tv-ad-google-home. html*, 2017.

[19] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, Omar Fawzi, and Pascal Frossard. Universal adversarial perturbations. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1765–1773, 2017.

[20] Paarth Neekhara, Shehzeen Hussain, Prakhar Pandey, Shlomo Dubnov, Julian J. McAuley, and Farinaz Koushanfar. Universal adversarial perturbations for speech recognition systems. *CoRR*, abs/1905.03828, 2019.

[21] Shaun Nichols. Tv anchor says live on-air 'alexa, order me a dollhouse'-guess what happens next. *The Register*, 7, 2017.

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

[23] Yao Qin, Nicholas Carlini, Ian Goodfellow, Garrison Cottrell, and Colin Raffel. Imperceptible, robust, and targeted adversarial examples for automatic speech recognition, 2019.

[24] Antony W Rix, John G Beerends, Michael P Hollier, and Andries P Hekstra. Perceptual evaluation of speech quality (pesq)-a new method for speech quality assessment of telephone networks and codecs. In *2001 IEEE international conference on acoustics, speech, and signal processing. Proceedings (Cat. No. 01CH37221)*, volume 2, pages 749–752. IEEE, 2001.

[25] Lea Schönherr, Thorsten Eisenhofer, Steffen Zeiler, Thorsten Holz, and Dorothea Kolossa. Imperio: Robust over-the-air adversarial examples for automatic speech recognition systems, 2020.

[26] Lea Schönherr, Katharina Kohls, Steffen Zeiler, Thorsten Holz, and Dorothea Kolossa. Adversarial attacks against automatic speech recognition systems via psychoacoustic hiding, 2018.

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

[28] Joseph Szurley and J. Zico Kolter. Perceptual based adversarial audio attacks, 2019.

[29] Rohan Taori, Amog Kamsetty, Brenton Chu, and Nikita Vemuri. Targeted adversarial examples for black box audio systems, 2019.

[30] Tavish Vaidya, Yuankai Zhang, Micah Sherr, and Clay Shields. Cocaine noodles: Exploiting the gap between human and machine speech recognition. In *9th USENIX Workshop on Offensive Technologies (WOOT 15)*, Washington, D.C., August 2015. USENIX Association.

[31] Qing Wang, Pengcheng Guo, and Lei Xie. Inaudible adversarial perturbations for targeted attack in speaker recognition. *CoRR*, abs/2005.10637, 2020.

[32] Hiromu Yakura and Jun Sakuma. Robust audio adversarial example for a physical attack. *CoRR*, abs/1810.11793, 2018.

[33] Zhuolin Yang, Bo Li, Pin-Yu Chen, and Dawn Song. Characterizing audio adversarial examples using temporal dependency. *CoRR*, abs/1809.10875, 2018.

[34] Xuejing Yuan, Yuxuan Chen, Yue Zhao, Yunhui Long, Xiaokang Liu, Kai Chen, Shengzhi Zhang, Heqing Huang, Xiaofeng Wang, and Carl A. Gunter. Commandersong: A systematic approach for practical adversarial voice recognition. *CoRR*, abs/1801.08535, 2018.

[35] Guoming Zhang, Chen Yan, Xiaoyu Ji, Tianchen Zhang, Taimin Zhang, and Wenyuan Xu. Dolphinattack. *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, Oct 2017.