Audio Adversarial Examples: Targeted Attacks on Speech-to-Text

Nicholas Carlini David Wagner
University of California, Berkeley

***Abstract*—We construct targeted audio adversarial examples on automatic speech recognition. Given any audio waveform, we can produce another that is over 99.9% similar, but transcribes as any phrase we choose (recognizing up to 50 characters per second). We apply our white-box iterative optimization-based attack to Mozilla's implementation DeepSpeech end-to-end, and show it has a 100% success rate. The feasibility of this attack introduce a new domain to study adversarial examples.**

I. Introduction

As the use of neural networks continues to grow, it is critical to examine their behavior in adversarial settings. Prior work [6] has shown that neural networks are vulnerable to *adversarial examples* [41], instances x' similar to a natural instance x, but classified incorrectly by a neural network.

While neural networks are being used in increasingly many new domains, existing work on adversarial examples has focused largely on the space of images, be it image classification [41], generative models on images [26], image segmentation [1], face detection [37], or reinforcement learning by manipulating the images the RL agent sees [5, 21]. In the discrete domain, there has been some study of adversarial examples over text classification [23] and malware classification [16, 20].

Neural networks are commonly used used for speech recognition — given an audio waveform x, perform the speech-to-text transform that gives the transcription y of the phrase being spoken (as used in, e.g., Apple Siri, Google Now, and Amazon Echo). On audio, untargeted adversarial examples are uninteresting, as simply causing word-misspellings would be regarded as a successful attack. Instead, the more powerful *targeted* attack asks not only that x and x' have different labels, but that the network assigns a specific target label (chosen by the adversary) to x'.

While there have been several attempts at producing targeted adversarial examples on audio, so far none have succeeded. Hidden voice commands [9, 40, 42] are targeted attacks, but require synthesizing new audio (analogous to the observation that neural networks can make high confidence predictions for unrecognizable images [33]). Other work has constructed standard untargeted adversarial examples on different audio systems [13, 24]. The current state-of-the-art targeted attack on automatic speech recognition is Houdini [10], which can only construct audio adversarial examples targeting phonetically similar phrases, leading the authors to state

> targeted attacks seem to be much more challenging when dealing with speech recognition systems than when we consider artificial visual systems [10]

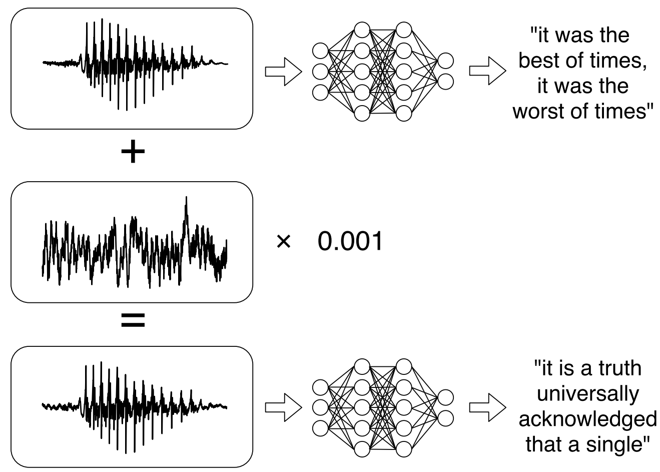

Fig. 1. Illustration of our attack: given any waveform, adding a small perturbation makes the result transcribe as any desired target phrase.

Contributions. In this paper, we demonstrate that targeted adversarial examples exist in the audio domain by attacking DeepSpeech [18], a state-of-the-art speech-to-text transcription neural network. Figure 1 illustrates our attack: given a natural waveform x, we are able to construct a perturbation δ that is nearly inaudible but so that $x+\delta$ is recognized as **any** desired phrase. We are able to achieve this by making use of strong, iterative, optimization-based attacks based on the work of [8].

Our white-box attack is end-to-end, and operates directly on the raw samples that are used as input to the classifier. This requires optimizing through the the MFC pre-processing transformation, which is has been proven to be difficult [9]. Our attack works with 100% success, regardless of the desired transcription, or initial source phrase being spoken.

By starting with an arbitrary waveform instead of speech (such as music), we can embed speech into audio that should not be recognized as speech; and by choosing silence as the target, we can hide audio from a speech-to-text system.

Audio adversarial examples give a new domain to explore these intriguing properties of neural networks. We hope others will build on our attacks to further study this field. To facilitate future work, we make our code available, along with the dataset we use for evaluation[1]. Additionally, we encourage the reader to listen to our audio adversarial examples.

[1] http://nicholas.carlini.com/code/audio_adversarial_examples

II. BACKGROUND

Neural Networks & Speech Recognition A neural network is a parameterized function $f(\cdot)$ consisting of many simple layers $f_i(\cdot)$. The function $f(\cdot)$ is differentiable and its parameters can be updated by gradient descent to learn any given mapping.

We represent audio as a N-dimensional vector x. Each element x_i is a signed 16-bit value, sampled at 16KHz. To reduce the input dimensionality, the Mel-Frequency Cepstrum (MFC) transform is often used as a preprocessing step. The MFC splits the waveform into 50 *frames* per second, and maps each frame to the frequency domain through an FFT.

Simple neural networks for classification take a single input and produce a single output probability distribution over all possible output labels. However, in the case of speech-to-text systems, there are exponentially many possible labels (i.e., there are over 26^N possible phrases of length N). This makes it computationally infeasible to enumerate all possible phrases.

Therefore, speech recognition systems often use Recurrent Neural Networks (RNNs) to map an audio waveform to a sequence of probability distributions over individual characters, instead of over complete phrases An RNN is a function which maintains a state vector s with $s_0 = \mathbf{0}$ and

$$(s_{i+1}, y^i) = f(s_i, x_i),$$

where the input x_i is one frame of input, and each output y^i is a probability distribution over which character was being spoken during that frame.

We use the DeepSpeech [18] speech-to-text system, which has recently been reproduced and made open source by Mozilla [32]. Internally, it consists of a preprocessing layer which computes the MFC followed by a recurrent neural network using LSTMs [19].

Connectionist Temporal Classication (CTC) [15] is a method of training a sequence-to-sequence neural network when the alignment between the input and output sequences is not known. In DeepSpeech, CTC is used because the inputs are an audio sample of a person speaking, and the labels the transcribed sentences, but the exact position of each word in the audio sample is not known.

We briefly summarize the key details, and give our notation. We refer readers to [17] for an excellent survey of CTC.

Let $\mathcal{X}$ be the input domain — a single frame of input — and $\mathcal{Y}$ be the range — the the characters a-z, space, and the special ϵ token (described below). Our neural network $f : \mathcal{X}^N \to [0,1]^{N\cdot|\mathcal{Y}|}$ takes a sequence of N frames $x \in \mathcal{X}$ and returns a probability distribution over the output domain for each frame. We write $f(\boldsymbol{x})^i_j$ to mean that the probability of frame $x_i \in \mathcal{X}$ having label $j \in \mathcal{Y}$. We use $\boldsymbol{p}$ to denote a phrase: a sequence of characters $\langle p_i \rangle$, where each $p_i \in \mathcal{Y}$.

While $f(\cdot)$ maps every frame to a probability distribution over the characters, this does not directly give us a probability distribution over all *phrases*. We must define the probability of a phrase as a function of the probability of each character.

To do this, we first make two short definitions. We say that a sequence π *reduces to* $\boldsymbol{p}$ if starting with π and making the following two operations (in order) yields $\boldsymbol{p}$:

1) Remove all sequentially duplicated tokens.
2) Remove all ϵ tokens.

For example, the sequence $a\ a\ b\ \epsilon\ \epsilon\ b$ reduces to $a\ b\ b$.

Further, we say that π is an alignment of $\boldsymbol{p}$ with respect to $\boldsymbol{y}$ (formally: $\pi \in \Pi(\boldsymbol{p}, \boldsymbol{y})$) if (a) π *reduces to* $\boldsymbol{p}$, and (b) the length of π is equal to the length of $\boldsymbol{y}$. The probability of alignment π under $\boldsymbol{y}$ is defined as the product of the likelihoods of each of its elements:

$$\Pr(\pi|\boldsymbol{y}) = \prod_i \boldsymbol{y}^i_{\pi^i}$$

With these definitions, we can now define the probability of a given phrase $\boldsymbol{p}$ under the distribution $\boldsymbol{y} = f(\boldsymbol{x})$ as

$$\Pr(\boldsymbol{p}|\boldsymbol{y}) = \sum_{\pi \in \Pi(\boldsymbol{p},\boldsymbol{y})} \Pr(\pi|\boldsymbol{y}) = \sum_{\pi \in \Pi(\boldsymbol{p},\boldsymbol{y})} \prod_i \boldsymbol{y}^i_{\pi^i}$$

As is usually done, the loss function used to train the network is the negative log likelihood of the desired phrase:

$$\text{CTC-Loss}(f(\boldsymbol{x}), \boldsymbol{p}) = -\log \Pr(\boldsymbol{p}|f(\boldsymbol{x})).$$

Despite the exponential search space, this loss can be computed efficiently with dynamic programming.

Finally, to *decode* a vector $\boldsymbol{y}$ to a phrase $\boldsymbol{p}$, we search for the phrase $\boldsymbol{p}$ that best aligns to $\boldsymbol{y}$.

$$C(\boldsymbol{x}) = \arg\max_{\boldsymbol{p}} \Pr(\boldsymbol{p}|f(\boldsymbol{x})).$$

Because computing $C(\cdot)$ requires searching an exponential space, it is typically approximated in one of two ways.

- *Greedy Decoding* searches for the most likely alignment (which is easy to find) and then reduces this alignment to obtain the transcribed phrase:

$$C_{\text{greedy}}(\boldsymbol{x}) = \text{reduce}(\arg\max_{\pi} \Pr(\pi|f(\boldsymbol{x})))$$

- *Beam Search Decoding* simultaneously evaluates the likelihood of multiple alignments π and then chooses the most likely phrase $\boldsymbol{p}$ under these alignments. The exact method used is out of scope of this paper. We refer the reader to [15] for a complete algorithm description.

Adversarial Examples. Evasion attack have long been studied on machine learning classifiers [3, 4, 29]. These attacks are practical against many types of models [6].

When discussion neural networks, these evasion attacks are referred to as *adversarial examples* [41]: for any input x, it is possible to construct an input x' that is similar to x (according to some metric) but so that the network assigns x and x' different labels [6]. In the audio domain, these untargeted adversarial example are usually not interesting: causing a speech-to-text system to transcribe "test sentence" as the misspelled "test sentense" does little to help an adversary.

Targeted Adversarial Examples are a more powerful attack: not only must the classification of x and x' differ, but the network must assign a specific label (chosen by the adversary) to the instance x'. The purpose of this paper is to show that targeted adversarial examples are possible with minimal distortion on speech-to-text systems.

III. Audio Adversarial Examples

A. Threat Model & Evaluation Benchmark

Threat Model. Given an audio waveform x, and target transcription y, our task is to construct another audio waveform $x' = x + \delta$ so that x and x' sound similar (formalized below), but so that $C(x') = y$. We report success only if the output of the network matches exactly the target phrase (i.e., contains no misspellings, extra whitespace, or any otherwise sentiment-preserving modifications); if we allow for these types of changes, we can obtain substantially lower distortion.

We assume a white-box setting where the adversary has complete knowledge of the model and its parameters. This is the threat model taken in most prior work [14] and allows the adversary the most power. Just as later work in the space of images showed black-box attacks are possible [22, 35]; we expect that our attacks can be extended to black-box attacks.

Distortion Metric. How should we quantify the amount of distortion introduced by a perturbation δ? In the space of images, despite some debate about what metrics truly mirror perceptual distance [36], most of the community has settled on l_p metrics [8], most often using l_∞ [14, 30], the maximum amount any pixel has been changed; we follow this convention.

We measure distortion in Decibels (dB): a logarithmic scale that measures the relative loudness of an audio sample:

$$dB(x) = \max_i \; 20 \cdot \log_{10}(x_i).$$

To say that some signal is "10 dB" is only meaningful when comparing it relative to some other reference point. In this paper, we compare dB level of the the distortion δ to the original waveform x. To make this explicit, we often write

$$dB_x(\delta) = dB(\delta) - dB(x).$$

Because the perturbation introduced is *quieter* than the original signal, the distortion is a negative number, where smaller values indicate quieter distortions.

While this metric may not be a perfect measure of distortion, as long as the perturbation is small enough, it will be imperceptible to humans. We encourage the reader to listen to our adversarial examples to hear how similar they sound. Alternatively, in Figure 2, we visualize two waveforms which transcribe to different phrases overlaid.

Evaluation Benchmark. To evaluate the effectiveness of our attack, we construct targeted audio adversarial examples on the first 100 test instances of the Mozilla Common Voice dataset. For each sample, we target 10 different *incorrect* transcriptions, chosen uniformly at random such that (a) the transcription is incorrect, and (b) it is theoretically possible to reach that target[2].

[2]Since audio is sampled at 50 frames per second, we can not make a 1 second audio clip target a 60 character transcription.

B. An Initial Formulation

As is commonly done [6, 41], we formulate the problem of constructing an adversarial example as an optimization problem: given a natural example x, and any target phrase t, we attempt to solve the formulation

$$\begin{aligned} &\text{minimize } dB_x(\delta) \\ &\text{such that } C(x+\delta) = t \\ &\qquad\qquad x + \delta \in [-M, M] \end{aligned}$$

Here M represents the maximum representable value (2^{15} in our case). This constraint can be handled by clipping the values of δ and for notational simplicity we omit it from future formulation. Due to the non-linearity of the constraint $C(x + \delta) = t$, standard gradient-descent techniques do not work well with this formulation.

Prior work [8, 41] has worked around this difficulty by reformulating the problem as

$$\text{minimize } dB_x(\delta) + c \cdot \ell(x + \delta, t)$$

where the loss function $\ell(\cdot)$ is constructed so that

$$\ell(x', t) \leq 0 \iff C(x') = t$$

and c is a parameter that trades off the importance of being adversarial and remaining close to the original example.

Constructing a loss function $\ell(\cdot)$ with this property is much simpler in the space of images than in the domain of audio; on images, $f(x)_y$ directly corresponds to the probability of the input x having label y. In contrast, for audio, we use a second decoding step to compute $C(x')$, and so constructing a loss function is nontrivial.

To begin, we use the CTC loss as the loss function:

$$\ell(x', t) = \text{CTC-Loss}(x', t).$$

For this loss function, one direction of the implication holds true (i.e., $\ell(x', t) \leq 0 \implies C(x') = t$) but the converse does not. Fortunately, this means that the resulting solution will still be adversarial, it just may not be minimally perturbed.

The second difficulty we must address is that when using a l_∞ distortion metric (as we do in this paper), this optimization process will often oscillate around a solution without converging [8]. Therefore, we use a similar trick to what is done in [8] and initially solve the formulation

$$\begin{aligned} &\text{minimize } |\delta|_2^2 + c \cdot \ell(x + \delta, t) \\ &\text{such that } dB_x(\delta) \leq \tau \end{aligned}$$

or equivalently we could require that $\delta_i \in [-10^{\tau/20}, 10^{\tau/20}]$. This bounds δ to have distortion at most τ. We initialize τ to some sufficiently large constant, and upon obtaining a solution δ^* to the above problem, we reduce τ and resume minimization until no solution can be found, and return the solution to the optimization problem with minimum distortion.

To ensure that our audio adversarial examples retain their label after discretizing to 16-bit audio, we add a small amount of Gaussian noise with standard deviation 1.0 to the input before classifying it.

To solve this formulation, we differentiate through the entire classifier to generate our adversarial examples — starting from the entire audio sample, through the MFC, and neural network, all the way to the final loss. We solve the minimization problem over the complete audio sample simultaneously. This is in contrast with prior work on hidden voice commands [9], which were generated sequentially, one frame at a time. We solve the minimization problem with the Adam [25] optimizer using a learning rate of 10, for a maximum of $5,000$ iterations.

Evaluation. We are able to generate targeted adversarial examples with 100% success for each of the source-target pairs with a mean perturbation magnitude of -31dB. For comparison, this is roughly the difference between ambient noise in a quiet room and a person talking [39]. We encourage the reader to listen to our audio adversarial examples[3]. The 95% confidence interval for distortion ranged from -15dB to -45dB.

The longer a phrase is, the more difficult it is to target: every extra character requires approximately a 0.1dB increase in distortion. However, conversely, we observe that the longer the initial source phrase is, the *easier* it is to make it target a given transcription. These two effects roughly counteract each other (although we were not able to measure this to a statistically significant degree of certainty).

Generating a single adversarial example requires approximately one hour of compute time on commodity hardware (a single NVIDIA 1080Ti). However, due to the massively parallel nature of GPUs, we are able to construct 10 adversarial examples simultaneously, reducing the time for constructing any given adversarial example to only a few minutes.[4]

C. Improved Loss Function

Carlini & Wagner [8] demonstrate that the choice of loss function impacts the final distortion of generated adversarial examples by a factor of 3 or more. We now show the same holds in the audio domain. While CTC loss is highly useful for training the neural network, we show that a careful design of the loss function allows generating better lower-distortion adversarial examples. For the remainder of this section, we focus on generating adversarial examples that are effective when using *greedy decoding*.

In order to minimize the CTC loss (as done in § III-B), an optimizer will make *every* aspect of the transcribed phrase more similar to the target phrase. That is, if the target phrase is "ABCD" and we are already decoding to "ABCX", minimizing CTC loss will still cause the "A" to be more "A"-like, despite the fact that the only important change we require is for the "X" to be turned into a "D".

[3] http://nicholas.carlini.com/code/audio_adversarial_examples

[4] Due to implementation difficulties, after constructing multiple adversarial examples simultaneously, we must fine-tune them individually afterwards.

This effect of making items classified more strongly as the desired label despite already having that label led Carlini & Wagner to design a more effective loss function:

$$\ell(x,t) = \max\left(f(x)_t - \max_{t' \neq t} f(x)_{t'}, 0\right).$$

Once the probability of item y is larger than any other item, the optimizer no longer sees a reduction in loss by making it more strongly classified with that label.

We now adapt this loss function to the audio domain. Assume we were given an alignment π that aligns the phrase $\boldsymbol{p}$ with the probabilities $\boldsymbol{y}$. Then the loss of this sequence is

$$\ell(\boldsymbol{x},\pi) = \sum_i \max\left(f(\boldsymbol{x})^i_{\pi^i} - \max_{t' \neq \pi^i} f(\boldsymbol{x})^i_{t'}, 0\right).$$

We can make one further improvement on this loss function. The constant c used in the minimization formulation determines the relative importance of being close to the original symbol versus being adversarial. A larger value of c allows the optimizer to place more emphasis on reducing $\ell(\cdot)$.

In audio, consistent with prior work [9] we observe that certain characters are more difficult for the transcription to recognize. When we choose only one constant c for the complete phrase, it must be large enough so that can make the most difficult character be transcribed correctly. This forces c to be larger than necessary for the easier-to-target segments. To resolve this issue, we instead use the following formulation:

$$\begin{aligned}&\text{minimize } |\delta|_2^2 + \sum_i c_i \cdot \ell_i(x+\delta, t)\\ &\text{such that } dB_x(\delta) < \tau\end{aligned}$$

where

$$\ell_i(\boldsymbol{x},\pi) = \max\left(f(\boldsymbol{x})^i_{\pi^i} - \max_{t' \neq \pi^i} f(\boldsymbol{x})^i_{t'}, 0\right).$$

Computing the loss function requires choice of an alignment π. If we were not concerned about runtime efficiency, in principle we could try all alignments $\pi \in \Pi(\boldsymbol{p})$ and select the best one. However, this is computationally prohibitive.

Instead, we use a two-step attack:

1) First, we generate an adversarial example using the CTC loss (using the methods in § III-B). Let x_0 denote the adversarial example found in this way. Note that the CTC loss explicitly constructs an alignment as part of decoding. Therefore, we extract the alignment π that is induced by x_0 (by computing $\pi = \arg\max_i f(x_0)_i$). We fix this alignment π and use it as the target in the second step.
2) Next, holding the alignment π fixed, we generate a better adversarial example x targeting the alignment π. In particular, we use gradient descent using the improved loss function above to minimize $|\delta|_2^2 + \sum_i c_i \cdot \ell_i(x+\delta,\pi)$, starting gradient descent at the initial point $\delta_0 = x_0 - x$.

Evaluation. We repeat the evaluation from Section III-B (above), and generate targeted adversarial examples for the

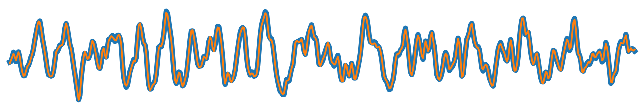

Fig. 2. Original waveform (blue, thick line) with adversarial waveform (orange, thin line) overlaid; it is nearly impossible to notice a difference. The audio waveform was chosen randomly from the attacks generated and is 500 samples long.

first 100 instances of the Common Voice test set. We are able to reduce the mean distortion from -31dB to -38dB. However, the adversarial examples we generate are now only guaranteed to be effective against a greedy decoder; against a beam-search decoder, the transcribed phrases are often more similar to the target phrase than the original phrase, but do not perfectly match the target.

Figure 2 shows two waveforms overlaid; the blue line is the original waveform, and the orange the modified adversarial waveform. This sample was chosen randomly from among the training data, and corresponds to a distortion of -30dB. Even visually, these two waveforms are nearly indistinguishable.

D. Audio Information Density

Recall that the input waveform is converted into 50 *frames* per second of audio, and DeepSpeech outputs a probability distribution of output characters for each frame. This places the maximum density of audio at 50 characters per second. We are able to generate adversarial examples that produce output at this maximum rate, i.e., 50 characters per second of audio. Thus, short audio clips can transcribe to a long textual phrase.

The loss function can be significantly simpler in this setting. The *only* alignment of $\boldsymbol{p}$ to $\boldsymbol{y}$ is the assignment $\pi = \boldsymbol{p}$. This means that the CW loss function can be applied directly without first heuristically finding an alignment; any other alignment would require omitting some character.

We perform this attack and find it is effective, although it requires a mean distortion of -18dB.

E. Starting from Non-Speech

Not only are we able to construct adversarial examples converting a person saying one phrase to that of them saying a different phrase, we are also able to begin with arbitrary non-speech audio sample and make that recognize as any target phrase. No technical novelty on top of what was developed above is required to mount this attack: we only change the initial audio waveform to be non-speech.

To evaluate the effectiveness of this attack, we take five-second clips from classical music (which contain no speech) and target phrases contained in the Common Voice dataset. We have found that this attack requires more computational effort (we perform $20,000$ iterations of gradient descent) and the total distortion is slightly larger, with a mean of -20dB.

F. Targeting Silence

Finally, we find it is possible to *hide* speech by adding adversarial noise that causes DeepSpeech to transcribe nothing. While performing this attack without modification (by just targeting the empty phrase) is effective, we can slightly improve on this if we define silence to be an arbitrary length sequence of only the space character repeated. With this definition, to obtain silence, we should let

$$\ell(\boldsymbol{x}) = \sum_i \max\left(\max_{t \in \{\epsilon, \text{``\,''}\}} f(\boldsymbol{x})^i_t - \max_{t' \notin \{\epsilon, \text{``\,''}\}} f(\boldsymbol{x})^i_{t'}, 0 \right).$$

We find that targeting silence is substantially easier than targeting a specific phrase: with distortion less than -45dB below the original signal, we can turn any phrase into silence.

This partially explains why it is easier to construct adversarial examples when starting with longer audio waveforms than shorter ones: because the longer sentence contains more sounds, the adversary can just silence the ones that are not required and obtain a subsequence that nearly matches the target. In contrast, for a shorter phrase, the adversary must synthesize new characters where ones did not exist previously.

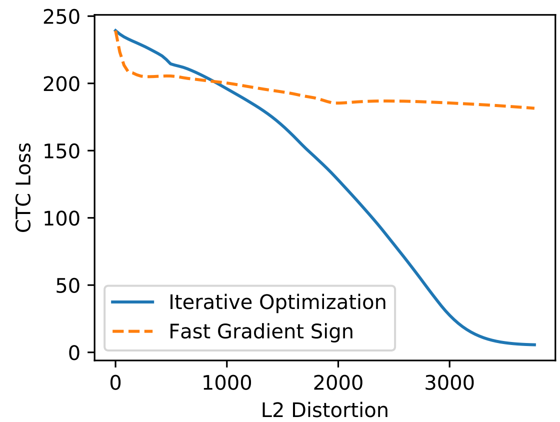

Fig. 3. CTC loss when interpolating between the original audio sample and the adversarial example (blue, solid line), compared to traveling equally far in the direction suggested by the fast gradient sign method (orange, dashed line). Adversarial examples exist far enough away from the original audio sample that exploiting the local linearity of neural networks is insufficient to construct targeted adversarial examples.

IV. Audio Adversarial Example Properties

A. Evaluating Single-Step Methods

In contrast to prior work which views adversarial examples as "blind spots" of a neural network, Goodfellow *et al.* argue that adversarial examples are largely effective due to the locally linear nature of neural networks.

The Fast Gradient Sign Method (FGSM) [14] demonstrates that this is true in the space of images. FGSM takes a single step in the direction of the gradient of the loss function. That is, given network F with loss function J, compute

$$x' \leftarrow x - \epsilon \cdot \text{sign}(\nabla_x J_f(x, y)).$$

Intuitively, for each pixel, this attack asks "in which direction should we modify this pixel to minimize the loss?" and then taking a small step in that direction for every pixel simultaneously. This attack has been shown to be effective on the domain of images [14].

However, we find that this type of single-step attack is not effective on audio adversarial examples: the inherent non-linearity introduced in computing the MFCCs, along with the depth of many rounds of LSTMs, introduces a large degree of non-linearity in the output.

In Figure 3 we compare the value of the CTC loss when traveling in the direction of a known adversarial example, compared to traveling in the fast gradient sign direction. While initially (near the source audio sample), the fast gradient direction is more effective at reducing the loss function, it quickly plateaus and does not decrease afterwards. On the other hand, iterative optimization chooses a direction that eventually finds an adversarial example (only when the CTC loss is below 10 does the phrase have the correct transcription).

We do, however, observe that the FGSM is effective at producing *untargeted* adversarial examples. For the same amount of distortion, adding random noise is five times less effective than taking a step in the the FGSM direction.

B. Robustness of Adversarial Examples

The minimally perturbed adversarial examples we construct in Section III-B can be made non-adversarial by trivial modifications to the input. And, indeed, simply increasing the beam search size will cause our even-more-minimally distorted adversarial examples from Section III-C from being classified as the adversarial target label.

However, we demonstrate here that it is possible to construct adversarial examples robust to synthetic noise. As a consequence, these noise models will not be effective as defenses.

Robustness to pointwise noise. Given an adversarial example x', adding pointwise random noise σ to x' and returning $C(x+\sigma)$ will cause x' to lose its adversarial label, even if the distortion σ is small enough to allow normal examples to retain their classification.

In practice, we find that letting $dB_x(\sigma) = -30$ is approximately the threshold of making minor modifications to natural examples, and is sufficient to corrupt adversarial perturbations.

However, we can construct an adversarial example that is robust to this kind of noise, using a combination of two methods. First, we generate a high-confidence adversarial example x' [6, 8], which we find by modifying the objective from finding a sample that is just barely classified as the target to one that is strongly classified as the target. Second, we increase the noise added during adversarial example generation to be the same -30dB noise distribution we would like to be resilient to. In particular, in each step of gradient descent, we randomly sample noise at -30dB, add it the audio before classification, and compute the gradient of this process. When we do this, $x'+\sigma$ remains adversarial at a cost of increasing the distortion by approximately 10dB.

Robustness to MP3 compression. On images, lossy JPEG compression has been shown to significantly degrade the effectiveness of adversarial examples [11, 12]. An adaptive adversary can generate JPEG compression resistant adversarial examples [38]. We have found that without any modifications, our high-confidence audio adversarial examples generated above are MP3-compression resistant.

V. Open Questions

Can these attacks be played over-the-air? Image-based adversarial examples have been shown to be feasible in the physical world [27], and can even be placed as textures on three dimensional objects such that pictures of the object will remain adversarial [2]. In the audio space, both hidden voice commands and inaudible voice commands are effective over-the-air when played by a speaker and recorded by a microphone [9, 42].

The audio adversarial examples we construct in this paper do not remain adversarial after being played over-the-air, and therefore present a limited real-world threat; however, just as the initial work on image-based adversarial examples did not consider the physical channel and only later was it shown to be possible, we believe further work will be able to produce audio adversarial examples that are effective over-the-air.

Do universal adversarial perturbations [31] exist? One surprising observation is that on the space of images, it is possible to construct a single perturbation δ that when applied to an arbitrary image x will make its classification incorrect. These attacks would be incredibly powerful on audio, and would correspond to a perturbation that could be played to cause any other waveform to recognize as a target phrase.

Are audio adversarial examples transferable? We can ask whether our attacks transfer: given an audio sample x, can we generate a single perturbation δ so that for multiple classifiers f_i, we have $f_i(x+\delta) = y$? Transferability is believed to be a fundamental property of neural networks [34], which significantly hinders constructing robust defenses [7] and allows attackers to mount black-box attacks [28]. Evaluating transferability on the audio domain is an important direction for future work.

Which existing defenses can be applied audio? To the best of our knowledge, all existing defenses to adversarial examples have only been evaluated on image domains. If the defender's objective is to produce a robust neural network, then it should improve resistance to adversarial examples on all domains, not just on images. Audio adversarial examples give a second point of comparison.

VI. Conclusion

We introduce audio adversarial examples: targeted attacks on automatic speech recognition. With powerful iterative optimization-based attacks applied completely end-to-end, we are able to turn any audio waveform into any target transcription with 100% success by only adding a slight distortion. We can cause audio to transcribe up to 50 characters per second (the theoretical maximum), cause music to transcribe as arbitrary speech, and hide speech from being transcribed.

We give preliminary evidence that audio adversarial examples have different properties from those on images by showing that linearity does not hold on the audio domain. We hope that future work will continue to investigate audio adversarial examples, and separate the fundamental properties of adversarial examples from properties which occur only on image recognition.

ACKNOWLEDGEMENTS

This work was supported by the National Science Foundation award CNS-1514457, Qualcomm, and the Hewlett Foundation through the Center for Long-Term Cybersecurity.

REFERENCES

[1] A. Arnab, O. Miksik, and P. H. Torr. On the robustness of semantic segmentation models to adversarial attacks. *arXiv preprint arXiv:1711.09856*, 2017.

[2] A. Athalye and I. Sutskever. Synthesizing robust adversarial examples. *arXiv preprint arXiv:1707.07397*, 2017.

[3] M. Barreno, B. Nelson, R. Sears, A. D. Joseph, and J. D. Tygar. Can machine learning be secure? In *Proceedings of the 2006 ACM Symposium on Information, computer and communications security*, pages 16–25. ACM, 2006.

[4] M. Barreno, B. Nelson, A. D. Joseph, and J. Tygar. The security of machine learning. *Machine Learning*, 81(2):121–148, 2010.

[5] V. Behzadan and A. Munir. Vulnerability of deep reinforcement learning to policy induction attacks. *arXiv preprint arXiv:1701.04143*, 2017.

[6] B. Biggio, I. Corona, D. Maiorca, B. Nelson, N. Šrndić, P. Laskov, G. Giacinto, and F. Roli. Evasion attacks against machine learning at test time. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 387–402. Springer, 2013.

[7] N. Carlini and D. Wagner. Magnet and "efficient defenses against adversarial attacks" are not robust to adversarial examples. *arXiv preprint arXiv:1711.08478*, 2017.

[8] N. Carlini and D. Wagner. Towards evaluating the robustness of neural networks. In *Security and Privacy (SP), 2017 IEEE Symposium on*, pages 39–57. IEEE, 2017.

[9] N. Carlini, P. Mishra, T. Vaidya, Y. Zhang, M. Sherr, C. Shields, D. Wagner, and W. Zhou. Hidden voice commands. In *25th USENIX Security Symposium (USENIX Security 16), Austin, TX*, 2016.

[10] M. Cisse, Y. Adi, N. Neverova, and J. Keshet. Houdini: Fooling deep structured prediction models. *arXiv preprint arXiv:1707.05373*, 2017.

[11] N. Das, M. Shanbhogue, S.-T. Chen, F. Hohman, L. Chen, M. E. Kounavis, and D. H. Chau. Keeping the bad guys out: Protecting and vaccinating deep learning with jpeg compression. *arXiv preprint arXiv:1705.02900*, 2017.

[12] G. K. Dziugaite, Z. Ghahramani, and D. M. Roy. A study of the effect of jpg compression on adversarial images. *arXiv preprint arXiv:1608.00853*, 2016.

[13] Y. Gong and C. Poellabauer. Crafting adversarial examples for speech paralinguistics applications. *arXiv preprint arXiv:1711.03280*, 2017.

[14] I. J. Goodfellow, J. Shlens, and C. Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

[15] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In *Proceedings of the 23rd international conference on Machine learning*, pages 369–376. ACM, 2006.

[16] K. Grosse, N. Papernot, P. Manoharan, M. Backes, and P. McDaniel. Adversarial perturbations against deep neural networks for malware classification. *arXiv preprint arXiv:1606.04435*, 2016.

[17] A. Hannun. Sequence modeling with ctc. *Distill*, 2017. doi: 10.23915/distill.00008. https://distill.pub/2017/ctc.

[18] A. Hannun, C. Case, J. Casper, B. Catanzaro, G. Diamos, E. Elsen, R. Prenger, S. Satheesh, S. Sengupta, A. Coates, et al. Deep speech: Scaling up end-to-end speech recognition. *arXiv preprint arXiv:1412.5567*, 2014.

[19] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[20] W. Hu and Y. Tan. Generating adversarial malware examples for black-box attacks based on gan. *arXiv preprint arXiv:1702.05983*, 2017.

[21] S. Huang, N. Papernot, I. Goodfellow, Y. Duan, and P. Abbeel. Adversarial attacks on neural network policies. *arXiv preprint arXiv:1702.02284*, 2017.

[22] A. Ilyas, L. Engstrom, A. Athalye, and J. Lin. Query-efficient black-box adversarial examples. *arXiv preprint arXiv:1712.07113*, 2017.

[23] R. Jia and P. Liang. Adversarial examples for evaluating reading comprehension systems. *arXiv preprint arXiv:1707.07328*, 2017.

[24] C. Kereliuk, B. L. Sturm, and J. Larsen. Deep learning and music adversaries. *IEEE Transactions on Multimedia*, 17(11):2059–2071, 2015.

[25] D. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[26] J. Kos, I. Fischer, and D. Song. Adversarial examples for generative models. *arXiv preprint arXiv:1702.06832*, 2017.

[27] A. Kurakin, I. Goodfellow, and S. Bengio. Adversarial examples in the physical world. *arXiv preprint arXiv:1607.02533*, 2016.

[28] Y. Liu, X. Chen, C. Liu, and D. Song. Delving into transferable adversarial examples and black-box attacks. *arXiv preprint arXiv:1611.02770*, 2016.

[29] D. Lowd and C. Meek. Adversarial learning. In *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*, pages 641–647. ACM, 2005.

[30] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017.

[31] S.-M. Moosavi-Dezfooli, A. Fawzi, O. Fawzi, and P. Frossard. Universal adversarial perturbations. *arXiv preprint arXiv:1610.08401*, 2016.

[32] Mozilla. Project deepspeech. https://github.com/mozilla/DeepSpeech, 2017.

[33] A. Nguyen, J. Yosinski, and J. Clune. Deep neural networks are easily fooled: High confidence predictions for unrecognizable images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 427–436, 2015.

[34] N. Papernot, P. McDaniel, and I. Goodfellow. Transferability in machine learning: from phenomena to black-box attacks using adversarial samples. *arXiv preprint arXiv:1605.07277*, 2016.

[35] N. Papernot, P. McDaniel, I. Goodfellow, S. Jha, Z. B. Celik, and A. Swami. Practical black-box attacks against deep learning systems using adversarial examples. *arXiv preprint arXiv:1602.02697*, 2016.

[36] A. Rozsa, E. M. Rudd, and T. E. Boult. Adversarial diversity and hard positive generation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 25–32, 2016.

[37] M. Sharif, S. Bhagavatula, L. Bauer, and M. K. Reiter. Accessorize to a crime: Real and stealthy attacks on state-of-the-art face recognition. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pages 1528–1540. ACM, 2016.

[38] R. Shin and D. Song. Jpeg-resistant adversarial images. *MLSec Workshop*, 2017.

[39] S. W. Smith et al. The scientist and engineer's guide to digital signal processing. 1997.

[40] L. Song and P. Mittal. Inaudible voice commands. *arXiv preprint arXiv:1708.07238*, 2017.

[41] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus. Intriguing properties of neural networks. *ICLR*, 2013.

[42] G. Zhang, C. Yan, X. Ji, T. Zhang, T. Zhang, and W. Xu. Dolphinatack: Inaudible voice commands. *CCS*, 2017.