Adversarial Attacks Against Automatic Speech Recognition Systems via Psychoacoustic Hiding

Anonymous Author(s)

Affiliation
Address
email

Abstract

The improvements of automatic speech recognition (ASR) base on an ongoing evolution of DNNs as the computational core of ASR. However, recent research results show that DNNs are vulnerable to adversarial perturbations, which allow attackers to force the transcription into a malicious output. In this paper, we introduce a new type of adversarial examples based on psychoacoustic hiding. Our attack exploits the characteristics of DNN-based ASR systems, where we use backpropagation to learn the degrees of freedom for the adversarial perturbation of the input signal, i.e., we apply a psychoacoustic model and manipulate the acoustic signal below the thresholds of human perception. To minimize the perceptibility of the perturbations, we use forced alignment to find the best fitting temporal alignment between the original audio sample and the malicious target transcription. These extensions allow us to embed an arbitrary audio input with a malicious voice command that is then transcribed by the ASR system, with the audio signal remaining barely distinguishable from the original signal. In an experimental evaluation, we attack the state-of-the-art speech recognition system Kaldi and determine the best performing parameter and analysis setup for different types of input. Our results show that we are successful in up to 98% of cases.

1 Introduction

DNNs have evolved into the state-of-the-art approach for many machine learning tasks, including automatic speech recognition (ASR) systems [1][2]. In practice, the importance of DNN-based ASR systems is steadily increasing, e.g., within smartphones or stand-alone devices. On the downside, their success also comes at a price: the number of necessary parameters is significantly larger than that of the previous Gaussian-Mixture-Model probability densities within Hidden Markov Models (so-called GMM-HMM systems) [3]. As a consequence, this high number of parameters gives an adversary much space to explore (and potentially exploit) blind spots that enable her to mislead an ASR system.

In the audio domain, Carlini et al. introduced hidden voice commands and demonstrated that targeted attacks against HMM-only ASR systems are feasible [4]. They use inverse feature extraction to create adversarial audio samples. The resulting audio samples are not intelligible by humans (in most of the cases) and may be considered as noise, but may make listeners suspicious. To overcome this limitation, Zhang et al. proposed DolphinAttacks: they showed that it is possible to hide a transcription by utilizing non-linearities of microphones to modulate the baseband audio signal with ultrasound higher than 20 kHz [5]. The drawback of this and similar ultrasound-based attacks [6][7] is that the attack is costly as the information to manipulate the input features needs to be retrieved from recordings of audio signals with the specific microphone and the modulation is tailored to this specific microphone. Concurrently, Carlini and Wagner published a technical report in which they
introduce a general targeted attack on ASR systems. Similarly to previous adversarial attacks on image classifiers, they work with a gradient-descent-based minimization [8], but replace the loss function by the CTC-loss. However, the constraint for the minimization of the difference between original and adversarial sample is borrowed from adversarial attacks on images and therefore does not consider the limitations and sensitivities of human auditory perception. Additionally, the algorithm often does not converge. This is solved by multiple initializations of the algorithm, which leads to high run-time requirements. Also recently, Yuan et al. described CommanderSong, which is able to hide transcripts within music [9]. However, this approach is only shown to be successful in music and it does not contain a human-perception-based noise reduction.

In this paper, we introduce a novel type of adversarial examples against ASR systems based on psychoacoustic hiding. We utilize psychoacoustic modeling, as in MP3 encoding, in order to reduce the perceptible noise. For this purpose, hearing thresholds are calculated based on psychoacoustic experiments by Zwicker et al. [10]. This limits the adversarial perturbations to those parts of the original audio sample, where they are not (or hardly) perceptible by a human. Furthermore, we use backpropagation as one part of the algorithm to find adversarial examples with minimal perturbations. This algorithm has already been successfully used for adversarial examples in other settings [8][11]. To show the general feasibility of psychoacoustic attacks, we feed the audio signal directly into the recognizer.

A key feature of our approach is the integration of the preprocessing step into the backpropagation. As a result, it is possible to change the raw audio signal without further steps. In addition, ASR highly depends on temporal alignment as it is a continuous process. We enhance our attack by computing an optimal alignment with the forced alignment algorithm, which calculates the best starting point in time for the backpropagation. Hence, we make sure to move the target transcription into parts of the original audio sample which are the most likely to not be perceivable by a human. We evaluated it against the state-of-the-art DNN-HMM-based ASR system Kaldi [12], which is one of the most popular toolchains for ASR among researchers [13][14][15][16][9] and is also used in commercial products such as Amazon’s Echo/Alexa and by IBM and Microsoft [17][18]. We analyze the optimal parameter settings, including different phone rates, allowed deviations from the hearing thresholds, and the number of iterations for the backpropagation. The experiments show that in comparison to other targeted attacks [9], the amount of noise is significantly reduced. This observation is confirmed during a audibility study, where test listeners transcribe adversarial examples. The results of the user study indicate that it is impossible to comprehend the target transcription of adversarial perturbations and only the original transcription is recognized by human listeners. A demonstration of our attack is available online[1] where we present several adversarial audio files generated for different kinds of attack scenarios.

2 Technical Background

Neural networks, as complex functions with millions of parameters and a limited number of training examples, have become prevalent in many ML tasks, greatly improving the accuracy of modern ASR systems. At the same time, they represent the Achilles’ heel of these systems that we are going to exploit for our ASR attack. In the following, we provide the technical background as far as it is necessary to understand the details of our approach.

2.1 Speech Recognition Systems

There is a variety of commercial and non-commercial ASR systems available. In the research community, Kaldi [12] is very popular, which is an open-source toolkit and provides a wide range of state-of-the-art algorithms for ASR. Given Kaldi’s popularity and its accessibility, this ASR system hence represents an optimal fit for our experiments. In the following, we describe the system in more detail:

Preprocessing Audio Input. For the feature extraction, we divide the input waveform into overlapping frames of fixed length. Each frame is transformed individually using the discrete Fourier transform (DFT) to obtain a frequency domain representation. We calculate the logarithm of the magnitude spectrum, a very common feature representation for ASR systems. A detailed description

is given in Section 3.2, where we explain the necessary integration of this particular preprocessing into our ASR system.

Neural Network. Kaldi takes a different route from an end-to-end-system: it uses a more conventional Hidden Markov Model (HMM) representation in the decoding stage and uses the DNN to model the probability of all HMM states (modeling context-dependent phonetic units) given the acoustic input signal. Therefore, the outputs of the DNN are pseudo-posteriors, which we use during the decoding step in order to find the most likely word sequence.

Decoding. Decoding in ASR systems, in general, utilizes some form of graph search for the inference of the most probable word sequence from the acoustic signal. In Kaldi, a static decoding graph is constructed as a composition of individual transducers (i.e., graphs with input/output symbol mappings attached to the edges of the graph). These individual transducers describe, for example, the grammar, the lexicon, context dependency of context-dependent phonetic units, and the transition and output probability functions of these phonetic units. The transducers and the pseudo-posteriors (i.e., the output of the DNN) are then used to find an optimal path through the word graph.

2.2 Adversarial Machine Learning

Adversarial attacks can, in general, be applied to any kind of machine learning system [19, 20, 21], but they are successful especially for DNNs [22, 23].

A trained DNN maps an input \( x \) to an output \( y = F(x) \). In the case of a trained ASR system, this is a mapping of the features into estimated pseudo-posteriors. Unfortunately, this mapping is not well defined in all cases due to the high number of parameters in the DNN, which leads to a very complex function \( F(x) \). Insufficient generalization of \( F(x) \) can lead to blind spots, which may not be obvious during system development or even deployment. We exploit this weakness by using a manipulated input \( x' \) that closely resembles the original input \( x \), but leads to a different mapping:

\[
x' = x + \delta, \text{ such that } F(x) \neq F(x'),
\]

where we minimize any additional noise \( \delta \) such that it stays close to the hearing threshold. For the minimization, we use a model of human audio signal perception.

2.3 Psychoacoustic Modeling

Psychoacoustic hearing thresholds describe how the interdependencies between different frequencies lead to masking effects in human perception. Probably the best-known example is MP3 compression [24], where the compression algorithm applies a set of hearing thresholds to the input signal to minimize the audio file size.

MP3 compression depends on an empirical set of hearing thresholds that define how dependencies between certain frequencies can mask, i.e., make imperceptible, other parts of an audio signal. When applied to the input signal, the thresholds indicate which parts of the signal can be altered in the following quantization step, and hence, help to compress the input. We utilize this psychoacoustic model for our manipulations of the signal, i.e., we apply it as a rule set to add inaudible noise. We derive the respective set of thresholds for an audio input from the psychoacoustic model of MP3 compression.

We use the fast Fourier transform to derive 32 frequency bands and to break the spectrum into MPEG ISO [24] specified scale factor bands. This segmentation of frequency bands helps to analyze the input signal according to its acoustic characteristics, as the hearing thresholds and masking effects directly relate to the individual bands. We measure this segmentation of bands in bark, a subjective measurement of frequency. Using this bark scale, we estimate the relevance of each band and compute its energy. In the following steps of the MP3 compression, the thresholds for each band indicate which parts of the frequency domain can be removed while maintaining a certain audio quality during quantization. In the context of our work, we use the hearing thresholds as a guideline for acceptable manipulations of the input signal. They describe the amount of energy that can be added to the input in each individual window of the signal. An example of such a matrix is visualized in Figure 1d. The matrices are always normalized in such a way that the largest time-frequency-bin energy is limited to 95 dB.
3 Attacking ASR via Psychoacoustic Hiding

In the following, we show how the audible noise can be limited by applying hearing thresholds during the creation of adversarial examples. The algorithm for the calculation of adversarial examples can be divided into three parts.

3.1 Forced Alignment

One major problem of attacks against ASR systems is that they require the recognition to pass through a certain sequence of HMM states in such a way that it leads to the target transcription. However, due to the decoding step—which includes a graph search—for a given transcription, many valid pseudo-posteriors combinations exist. For example, when the same text is spoken at different speeds, the sequence of the HMM states is correspondingly shorter or longer. We can benefit from this fact by using that version of pseudo-posteriors which best fits the given audio signal and the desired target transcription. We use forced alignment as an algorithm for finding the best possible temporal alignment between the acoustic signal that we manipulate and the transcription that we wish to obtain. This algorithm is provided by the Kaldi toolkit. Note that it is not always possible to find an alignment that fits an audio file to any target transcription. In this case, we define the alignment by dividing the audio sample equally into the number of necessary states and set the target according to this division.

3.2 Integrating Preprocessing

We integrate the preprocessing step and the DNN step into one joint DNN. This design choice does not affect the accuracy of the ASR system, but it allows for manipulating the raw audio data by also applying backpropagation to the preprocessing steps, directly giving us the optimally adversarial audio signal as result.

All preprocessing steps are summarized in $F_p(x)$ and the function returns the input features for the DNN. For the backpropagation, it is necessary to know the derivatives $\frac{\partial F_p(x)}{\partial x}$ of each of the four preprocessing steps:

**Framing and Window Function.** In the first step, the raw audio data is divided into $T$ frames of length $N$ and a window function is applied to each frame. A window function is a simple, element-wise multiplication with fixed values $w(n)$

$$x_w(t, n) = x(t, n) \cdot w(n), \quad n = 0, \ldots, N-1,$$

with $t = 0, \ldots, T-1$. Thus, the derivative is just $\frac{\partial x_w(t, n)}{\partial x(t, n)} = w(n)$.

**Discrete Fourier Transform.** For transforming the audio signal into the frequency domain, we apply a DFT to each frame $x_w$. This transformation is a common choice for audio features. The DFT is defined as $X(t, k) = \sum_{n=0}^{N-1} x_w(t, n) e^{-i2\pi \frac{kn}{N}}, \quad k = 0, \ldots, N - 1$.

Since the DFT is a weighted sum with fixed coefficients $e^{-i2\pi \frac{kn}{N}}$, the derivative for the backpropagation is simply the corresponding coefficient

$$\frac{\partial X(t, k)}{\partial x_w(t, n)} = e^{-i2\pi \frac{kn}{N}}, \quad k, n = 0, \ldots, N - 1.$$

**Magnitude.** The output of the DFT is complex valued, but as the phase is not regarded as relevant for speech recognition, we just use the magnitude of the spectrum, which is defined as $|X(t, k)|^2 = \text{Re}(X(t, k))^2 + \text{Im}(X(t, k))^2$, with $\text{Re}(X(t, k))$ and $\text{Im}(X(t, k))$ as the real and imaginary part of $X(t, k)$. For the backpropagation, we need the derivative of the magnitude. In general, this is not well defined and allows two solutions for

$$\frac{\partial |X(t, k)|^2}{\partial X(t, k)} = \begin{cases} 2 \cdot \text{Re}(X(t, k)) \\ 2 \cdot \text{Im}(X(t, k)) \end{cases}.$$

We circumvent this problem by considering the real and imaginary parts separately and calculate the derivatives for both cases

$$\nabla X(t, k) = \begin{pmatrix} \frac{\partial |X(t, k)|^2}{\partial \text{Re}(X(t, k))} \\ \frac{\partial |X(t, k)|^2}{\partial \text{Im}(X(t, k))} \end{pmatrix} = \begin{pmatrix} 2 \cdot \text{Re}(X(t, k)) \\ 2 \cdot \text{Im}(X(t, k)) \end{pmatrix}.$$
This is possible, as real and imaginary parts are stored separately during the calculation of the DNN.

**Logarithm.** The last step is taking the logarithm of the squared magnitude \( F_P(x) = \log(|X(t, k)|^2) \), which yields a common feature representation in speech recognition systems.

### 3.3 Hearing Thresholds

Psychoacoustic hearing thresholds allow us to limit audible distortions from all signal manipulations. More specifically, we use the hearing thresholds during the manipulation of the input signal in order to limit audible distortions. For this purpose, we use the original audio signal to calculate the hearing thresholds \( \mathbf{H} \) as described in Section 2.3. We limit the differences \( \mathbf{D} \) between the original signal spectrum \( \mathbf{S} \) and the modified signal spectrum \( \mathbf{M} \) to the threshold of human perception for all times \( t \) and frequencies \( k \)

\[
D(t, f) \leq H(t, k), \quad \forall t, k, \quad \text{with} \quad D(t, k) = 20 \cdot \log_{10} \left( \frac{|S(t, k) - M(t, k)|}{\max_{t,k}(|S|)} \right).
\]

The maximum value of the power spectrum \( |\mathbf{S}| \) defines the reference value for each utterance, which is necessary to calculate the difference in dB. Examples for \( |\mathbf{S}| \), \( |\mathbf{M}| \), \( |\mathbf{D}| \), and \( \mathbf{H} \) in dB are plotted in Figure 1 where the power spectra are plotted for one utterance. We calculate the amount of distortion that is still acceptable via \( \Phi = \mathbf{H} - \mathbf{D} \). The resulting matrix \( \Phi \) contains the difference in dB to the hearing thresholds.

In the following step, we use the matrix \( \Phi \) to derive scaling factors. First, because the thresholds are tight, an additional variable \( \lambda \) is added, to allow the algorithm to differ from the hearing thresholds by small amounts \( \Phi^* = \Phi + \lambda \). In general, a negative value for \( \Phi^*(t, k) \) indicates that we crossed the threshold. As we want to avoid more noise for these time-frequency-bins, we set all \( \Phi^*(t, k) < 0 \) to zero. We then obtain a time-frequency matrix of scale factors \( \hat{\Phi} \) by normalizing \( \Phi^* \) to values between zero and one, via

\[
\hat{\Phi}(t, k) = \frac{\Phi^*(t, k) - \min_{t,k}(\Phi^*)}{\max_{t,k}(\Phi^*) - \min_{t,k}(\Phi^*)}, \quad \forall t, k.
\]

The scaling factors are applied during each backpropagation iteration. Using the resulting scaling factors \( \hat{\Phi}(t, k) \) typically leads to good results, but especially in the cases where only very small changes are acceptable, this scaling factor alone is not enough to satisfy the hearing thresholds. Therefore, we use another, fixed scaling factor, which only depends on the hearing thresholds \( \mathbf{H} \). For this purpose, \( \mathbf{H} \) is also scaled to values between zero and one, denoted by \( \hat{\mathbf{H}} \).

Therefore, the gradient \( \nabla X(t, k) \) calculated via Equation (1) between the DFT and the magnitude step is scaled by both scaling factors

\[
\nabla X^*(t, k) = \nabla X(t, k) \cdot \hat{\Phi}(t, k) \cdot \hat{H}(t, k), \quad \forall t, k.
\]
4 Experiments and Results

With the help of the following experiments, we verify and assess the proposed attack. We target the ASR system Kaldi and use it for our speech recognition experiments. We assess the influence of significant parameter settings on the success of the adversarial attack.

4.1 Experimental Setup

To verify the feasibility of our targeted adversarial attacks, we have used the default settings for the Wall Street Journal (WSJ) training recipe of the Kaldi toolkit [12]. Only the preprocessing step was adapted for the integration into the DNN. The WSJ data set is well suited for large vocabulary ASR: it is phone-based and contains more than 80 hours of training data, composed of read sentences of the Wall Street Journal recorded under mostly clean conditions.

4.2 Word Error Rate

As the adversarial examples are primarily designed to fool an ASR system, a natural metric for our success is the accuracy with which the target transcription was actually recognized. For this purpose, we use the Levenshtein distance [25] to calculate the word error rate (WER). A dynamic-programming algorithm is employed to count the number of deleted \( D \), inserted \( I \), and substituted \( S \) words in comparison to the total number of words \( N \) in the sentence, which together allows for determining the word error rate via

\[
WER = \frac{D + I + S}{N}.
\]

When the adversarial example is based on audio samples with speech, it is possible that the original text is transcribed instead of—or in addition to—the target transcription. Therefore, it can happen that many words are inserted, possibly even more words than contained in the target text. This can lead to WERs larger than 100 %, which can also be observed in Table 1.

4.3 Evaluation

In the next steps, the optimal settings are evaluated, considering the hearing thresholds and the success rate to generate valid adversarial examples.

4.3.1 Evaluation of Hearing Thresholds

In Table 1, the results for speech and music samples are shown for 500 and for 1000 iterations of backpropagation, respectively. The value in the first row shows the setting of \( \lambda \). For comparison, the case without the use of hearing thresholds is shown in the column ‘None.’ We applied all combinations of settings on a test set of speech containing 72 samples and a test set of music containing 70 samples. The test set of speech was the same as for the previous evaluations and the target text was the same for all audio samples.

The results in Table 1 show the dependence on the number of iterations and on \( \lambda \). The higher the number of iterations and the higher \( \lambda \), the lower the WER becomes. The experiments with music show some exceptions to this rule, as a higher number of iterations slightly increases the WER in some cases. However, this is only true where no thresholds were employed or for \( \lambda = 50 \). As is to be expected, the best WER results were achieved when the hearing thresholds were not applied. However, the results with applied thresholds show that it is indeed feasible to find a valid adversarial example very reliably even when minimizing human perceptibility.

<table>
<thead>
<tr>
<th>Iter.</th>
<th>None</th>
<th>50 dB</th>
<th>40 dB</th>
<th>30 dB</th>
<th>20 dB</th>
<th>10 dB</th>
<th>0 dB</th>
</tr>
</thead>
<tbody>
<tr>
<td>Speech</td>
<td>500</td>
<td>2.14</td>
<td>6.96</td>
<td>11.07</td>
<td>16.43</td>
<td>36.43</td>
<td>92.69</td>
</tr>
<tr>
<td></td>
<td>1000</td>
<td>1.79</td>
<td>3.93</td>
<td>5.00</td>
<td>7.50</td>
<td>22.32</td>
<td>76.96</td>
</tr>
<tr>
<td>Music</td>
<td>500</td>
<td>1.04</td>
<td>8.16</td>
<td>13.89</td>
<td>22.74</td>
<td>31.77</td>
<td>60.07</td>
</tr>
<tr>
<td></td>
<td>1000</td>
<td>1.22</td>
<td>10.07</td>
<td>9.55</td>
<td>15.10</td>
<td>31.60</td>
<td>56.42</td>
</tr>
</tbody>
</table>
4.3.2 Number of Required Repetitions

We also analyzed the number of iterations needed to obtain a successful adversarial example for a randomly chosen audio input and target text. The results are shown in Figure 2. We tested our approach for speech and music, setting $\lambda = 0$, $\lambda = 20$, and $\lambda = 40$, respectively. For the experiments, we randomly chose speech files from 150 samples and music files from 72 samples. For each sample, a target text was chosen randomly from 120 predefined texts. The only constraint was that we used only audio-text-pairs with fewer than 7 phones per second of audio. In the case of a higher phone rate, we selected a new audio file. We repeated the experiment 100 times for speech and for music and used these sets for each value of $\lambda$.

For each round, we ran 100 iterations and checked the transcription. If the target transcription was not recognized successfully, we started the next 100 iterations and re-checked, repeating until either the maximum number of 5000 iterations was reached or the target transcription was successfully recognized. An adversarial example was only counted as a success if it had a WER of 0%. There were also cases where no success was achieved after 5000 iterations. This varied from only 2 cases for speech audio samples with $\lambda = 40$ up to 9 cases for music audio samples with $\lambda = 0$.

4.4 Comparison

We compare the amount of noise with CommanderSong [9], as their approach is also able to create targeted attacks using Kaldi. Additionally, it is the only recent approach, which reported signal-to-noise-ratio (SNR) for their results. The SNR measures the amount of noise $\sigma$, added to the original signal $x$, computed via $\text{SNR}(\text{dB}) = 10 \cdot \log_{10} \frac{P_x}{P_\sigma}$, where $P_x$ and $P_\sigma$ are the energies of the original signal and the noise.

Table 2 shows the SNR for successful adversarial samples, where no hearing thresholds are used (None) and for different values of $\lambda$ (40 dB, 20 dB, and 0 dB) in comparison to CommanderSong. Note, that the SNR does not measure the perceptible noise. Nevertheless, the results show, that in all cases, even if no hearing thresholds are used, we achieve higher SNRs, meaning, far less noise was added to create a successful adversarial example.

4.5 Transcription Listening Test

While the original text of a speech audio sample should still be understandable by human listeners, we aim for a result where the hidden command cannot be transcribed or even identified as speech. Therefore, we performed the transcription test, in which listeners were asked to transcribe the utterances of original and adversarial audio samples.

The utterances were the same for everyone, but with randomly chosen conditions: 9 original utterances, 3 adversarial examples with $\lambda = 0$, $\lambda = 20$, and $\lambda = 40$ respectively and 3 difference signals of the original and the adversarial example, one for each value of $\lambda$. For the adversarial utterances, we made
sure that all samples were valid, such that the target text was successfully hidden within the original utterance. An example for these successful adversarial utterances is the following:

Original audio transcription:

“SPECIFICALLY THE UNION SAID IT WAS PROPOSING TO PURCHASE ALL OF THE ASSETS OF THE OF UNITED AIRLINES INCLUDING PLANES GATES FACILITIES AND LANDING RIGHTS.”

Adversarial audio transcription:

“DEACTIVATE SECURITY CAMERA AND UNLOCK FRONT DOOR.”

For the evaluation, we have collected data from 22 listeners during an internal study. For the original utterances and the adversarial utterances, an average human-transcription WER of 12.59% and 12.61% was calculated. The marginal difference shows that the difference in the audio does not influence the intelligibility of the utterances. A two-sided t-test with a significance level of 1% shows no difference for the distributions of original and adversarial utterances.

5 Discussion

The choice of the parameters highly affects the amount of perceptible noise. The evaluation has shown that a higher number of iterations increases the success rate, but simultaneously the amount of noise. However, for up to 500 iterations, the success rate is already very high and therefore, 500 should not be exceeded. Additionally, by this choice, the required calculation time is reduced as well. If the success rate needs to be raised, the increase of $\lambda$ has a higher effect. Additionally, the phone rate should be set to an optimum value as this highly affects the success of the attack. Besides improving the success of the attack, the choice of the original audio sample greatly influences the quality of the adversarial example. We recommend using music or other unsuspicious audio samples, like bird twittering, which do not contain speech, as speech has to be obfuscated, typically leading to larger required adversarial perturbations.

As a general countermeasure, one approach could be to consider human perception. A very simple version would be to apply MP3 encoding to the input data. However, the DNN is not trained for that kind of data. Nevertheless, we did run some tests on our adversarial examples. With this setup, the original transcription could not be recovered, but the target transcription was also distorted. We assume that training the ASR-DNN with MP3-encoded audio files will only move the vulnerability into the perceptible region of the audio files, but will not circumvent blind spots of DNNs completely.

6 Conclusion

We have presented a new method for creating adversarial attacks on ASR systems, which explicitly take dynamic human hearing thresholds into account. In this way, borrowing the mechanisms of MP3 encoding, the audibility of the added noise is clearly reduced. We perform our attack against the state-of-the-art Kaldi ASR system and feed the adversarial input directly into the recognizer in order to show the general feasibility of psychoacoustics-based attacks.

By applying forced alignment and backpropagation to the DNN-HMM system, we were able to create inconspicuous adversarial perturbations very reliably. It is possible to hide any target transcription within any audio file, with a success rate up to 98%, and, with the correct attack vectors, it was possible to hide the noise below the hearing threshold and make the changes psychophysically almost imperceptible. The choice of the original audio sample, an optimal phone rate, and forced alignment give the optimal starting point for the creation of adversarial examples. Additionally, we have evaluated different algorithm setups, including the number of iterations and the allowed deviation from the hearing thresholds. The comparison with another approach in [9], which is also able to create targeted adversarial examples, shows that our approach needs far lower distortions. Listening tests have proven that the target transcription was incomprehensible for human listeners. Future work should investigate the hardening of ASR systems by considering psychoacoustic models, in order to prevent these presently fairly easy attacks.
References


