Hear me if you can!
Audio Steganography

Mithilesh Vaidya
Department of Electrical Engineering
IIT Bombay
mithilesh.vaidya@iitb.ac.in

Rishabh Dahale
Department of Electrical Engineering
IIT Bombay
dahalerishabh1@iitb.ac.in

Samyak Shah
Department of Electrical Engineering
IIT Bombay
samyakshah@iitb.ac.in

***Abstract*—Audio steganography is a technique for concealing the existence of information by embedding it within non-secret audio, called the carrier audio signal. There is a trade-off between the amount of information encoded and the imperceptibility of the change in the encoded audio. In this work, we implement a deep-learning based audio steganography technique. An ASR model is trained on the TIMIT dataset. We exploit the susceptibility of the trained model to adversarial examples. Given an input recording and a list of phones to encode, the technique searches for a small imperceptible perturbation. When this perturbation is added to the original audio and passed through the model, we recover the encoded text as the ASR output. PESQ score is used as the evaluation metric to quantify the amount of degradation. We also study the time required for encoding the text as a function of it's length, nature and PESQ. Lastly, we study the effect of a Gaussian Noise channel on our encoding technique.**

I. Introduction

Due to rapid adoption of internet among the public and easy and abundant availability of digital data, there is a growing demand for data protection. Currently, the popular methods are cryptography, watermarking and steganography. In cryptography, the structure of a message is scrambled to make it meaningless unless a decryption key is available whereas in watermarking the information is hidden to convey information like copyright and ownership. Steganography does not alter the structure of the message, but hides information so that it cannot be seen. It prevents an unintended recipient from even suspecting that data exists.

Audio steganography is useful for transmitting sensitive information hidden in an audio signal. Audio signals provides a lot of redundant space for embedding hidden information and have a good imperceptibility in the transmission of information. Two main criteria for successful embedding of a concealed message are:

1) Resulting audio signal should be perceptually indistinguishable from host audio signal.
2) Embedded message can be successfully recovered.

Traditional audio information hiding techniques can be classified into two major categories: time domain based and frequency domain based. Time domain techniques embed information on the carrier in time domain. They generally have large hiding capacity and low imperceptibility. Examples of such techniques are hiding information in LSB (least significant bit) [2], spread spectrum [3] and echo hiding [4]. Frequency domain techniques generally modify the transform domain to hide information. Such techniques generally posses high imperceptibility but low poor hiding capacity. Such techniques generally make use of discrete cosine transform (DCT), phase coding [5] and discrete wavelet transform (DWT) [6].

This work is based on a paper by Kong et al. [1] in which the authors used a DNN-based ASR model to hide information in the audio signal by adding small deviations in time domain. We implement their algorithm and carry out various experiments to study the tradeoff between hiding capacity, imperceptibility and nature of encoded text. We also test the ability of the model to learn an optimal perturbation in presence of Gaussian noise. Finally, we extend this work by presenting preliminary results of a similar method in the frequency domain. The main reason for studying this technique is to bring in psycho-acoustic concepts e.g. encoding the perturbation is specific regions along the frequency to ensure imperceptibility is high. This could lead to better PESQ scores.

II. Methodology

A. ASR model

We train a simple convolutional-recurrent neural network for the ASR task. The architecture is given in figure 1. The model accepts log spectrogram as input and outputs a distribution over 48 phones at each time step. The model is trained using CTC loss. We train, validate and test on the TIMIT dataset. Note that once the model is trained, the weights are **frozen**.

B. Encoding

1) Sample-based: Given an input recording in time domain (x) and a list of phones to encode (t), we follow the encoding procedure shown in figure 2. We search for a perturbation δ such that $f(x+\delta)=t$ where $f(.)$ denotes the final output of the ASR model after CTC decoding. We need to also ensure that the perturbation remains imperceptible. One way to achieve this is to minimise the amplitude of δ or in other words, the L_∞ norm. Thus, our goal becomes:

$$\delta = \underset{k}{\operatorname{argmin}} \|k\|_\infty \text{ s.t. } f(x+k)=t$$

Note that minimising the L_∞ norm need not necessarily give the best PESQ score (our evaluation metric). This is something we plan to study in the future.

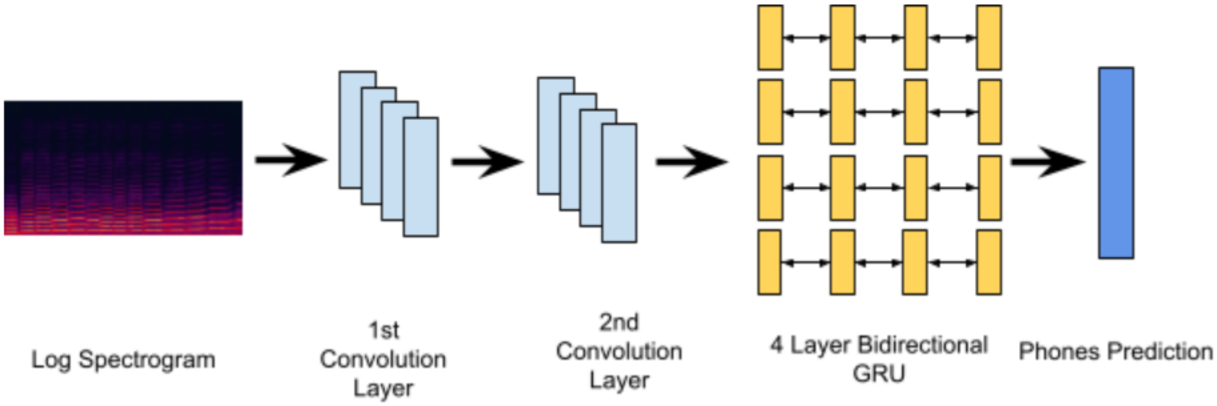

Fig. 1. Architecture of the ASR model trained on TIMIT.

Refer to Algorithm 1 for the pseudo-code. We initialise δ to 0. The maximum deviation τ is set to the maximum value of the audio signal. For every iteration, we feed the log spectrogram of the perturbed audio to the ASR model and obtain the phone distribution at each time step. The CTC loss with respect to the text to encode *t* is calculated. Our goal is to minimise this loss so that on decoding, we recover *t*. When the decoded text matches the target text, we decay τ by a predefined constant factor γ. This step ensures that the perturbation keeps getting smaller in amplitude as iterations increase.

At every time step, we use the Adam optimiser to update δ. Also, δ is clipped so that it lies between the maximum allowed amplitude τ in magnitude.

We use PESQ as the evaluation metric. It quantifies the amount of degradation with respect to the original clean audio. The quantification is based on various human hearing attributes. We report PESQ whenever the decoded text matches the target text.

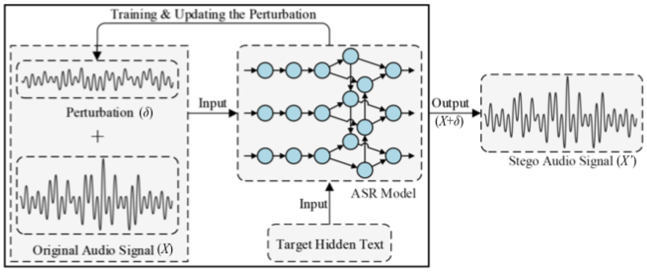

Fig. 2. The process of embedding the hidden text in the given audio signal. Figure reproduced from the main reference [1]

Algorithm 1: Estimating δ

```
Input  : Original signal x, Text to encode t
Output: Perturbation $\delta$
Initialise: $\delta$ = best = 0;
Parameters: $\gamma$ - decay factor, N - max iterations
$\tau$ = max(x) - amplitude of perturbation
for $i$ = 1, 2, ..., N do
    input = logspectrogram(x + $\delta$)
    output = model(input)
    //Calculate loss
    L = CTCLoss(output, t)
    //Decode
    pred = decode(output)
    //Check if prediction matches target
    if pred == t then
        //Store current best
        best = $\delta$
        //Decay the allowed amplitude
        $\tau \leftarrow \tau * \gamma$
    end
    //Update delta
    $\delta \leftarrow Adam.minimise(L,\delta)$
    $\delta = clip(\delta, -\tau, \tau)$
end
return best
```

2) *Spectrogram-based:* Instead of calculating a perturbation in the time domain, we can also look for perturbations in the log spectrogram domain. The perturbed audio can be recovered by inverting the log spectrogram. As far as number of parameters are concerned, framing reduces them by a factor of 100 along time (assuming a hop size of 10 ms) but an N-point FFT increases them by a factor N. Thus, if N $>$ 100, the number of parameters increases. However, we can exploit the psycho-acoustic properties to encode information in specific regions of the spectrogram [7], [9]. We could use a more nuanced loss function which is inspired by properties of the human hearing system e.g. deviations at higher frequencies are less perceptible as compared to those at lower frequencies. This could potentially lead to better PESQ scores since PESQ and amplitude of the perturbation need not always show an inverse relationship. The algorithm for computing the perturbation is very similar to algorithm 1. We used the Griffin-Lim

inversion algorithm found in the torchaudio library to invert the perturbed spectrogram.

C. Decoding

For extracting the encoded text, we simply pass the perturbed audio through our ASR model. After CTC decoding, we recover the text. It has been observed that passing the perturbed signal through any other ASR model (different architecture or same architecture with different weights) does not give the encoded text as output [1]. In other words, the perturbation is highly sensitive to the model weights and hence the encoding technique is very secure.

III. Experiments

A. Details

We deal with audio files sampled at 16 KHz. Log spectrogram of this audio, sampled using a Hann window of width 20ms and hop size of 10ms is given as input to the ASR model. The model consist of 2 layers of CNN followed by 4 layers of bidirectional GRU. Both the CNN layers use 32 filters with kernel size $(5, 32)$. The first layer has a stride of 1, while the second layer has a stride of 2. Both the layers use ReLU activation function. Dropout probability of 0.4 is used in both the CNN layers. This is followed by a 256 dimensional bidirectional GRU whose outputs are finally passed through a 48 dimensional fully connected layer with Softmax activation. Phone error rate (PER) of 19.7% was reached when the model was trained on CTC loss using SGD as the optimizer with a learning rate of 0.003 and momentum of 0.95. Batch size was set to 64 and we trained the model for 1000 epochs.

In algorithm 1, τ is initialised to max(x) while decay factor γ is set to 0.75. We run the algorithm for N = 10,000 iterations. Every 50 iterations, we check whether the CTC decoded text matches the target text. If yes, we store the perturbation and decay the threshold by γ. The learning rate for Adam is set to 0.01 and we decay it with by a factor of 0.995 every 50 iterations.

B. Results

Figure 3 shows the variation of loss with iterations. The loss plotted is the CTC loss between the predicted phone sequence by our ASR model and the target audio. At certain iterations, a spike in the loss can be observed. This is because at these iterations, for a given amplitude of perturbation, the target phone sequence was correctly predicted. As a result, the amplitude of the perturbation was reduced, resulting in sudden spike in the loss function.

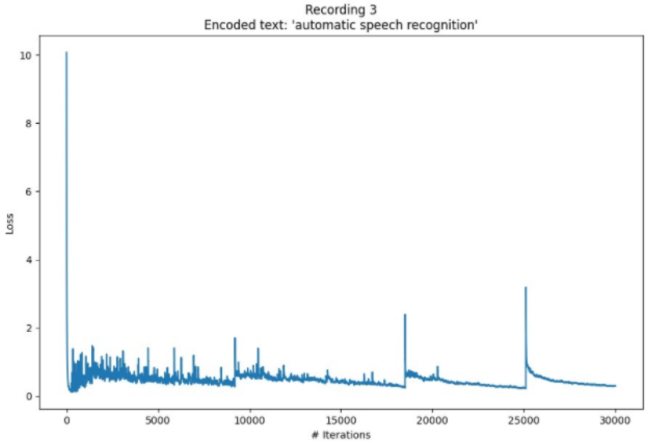

Fig. 3. Variation of loss b/w predicted phone sequence and target phone sequence v/s iteration. The carrier audio has the dialogue: ”Now I am become death, the destroyer of worlds” while the encoded text is ”automatic speech recognition”.

Table I lists the PESQ for various experiments we carried on a few audio files. It can be seen that for noisy carriers (Breaking Bad, Oppenheimer and Lion King which are obtained from the internet), we were able to achieve a higher PESQ as compared to the clean audio files from TIMIT. Any perturbation to a recording from TIMIT is highly perceptible.

Source	Sr No	Audio Duration	PPS	PESQ WB (best)	PESQ NB (best)
TIMIT	1	3	6.66	1.7	1.2
TIMIT	2	3	5	1.27	1.04
Breaking Bad	3	11	2.7	3.48	2.4
Breaking Bad	4	11	1.36	3.22	1.95
Oppenheimer	5	4	8	4.39	3.42
Oppenheimer	6	4	1.75	4.52	4.13
Lion King	7	19	5.42	2.79	1.98
Lion King	8	19	0.68	3.67	2.9

TABLE I
Results on different audio files. PPS: Phones Per Second, Sr. No correspond to the audio-target pair sr no in table V. Details about the text present in the audio files and the text encrypted in it can be found in V in Appendix A. Higher PPS correspond to the longer encrypted text.

Figure 4 shows the variation of wide band and narrow band PESQ with number of iterations. Although we ran experiments for 10,000 iterations, the range of x axis is restricted. This is because PESQ was noted only when the target phone sequence was decoded and after a certain iteration, due to gamma decay of the amplitude of perturbation, no perturbation was found so as to recover the target sequence after decoding.

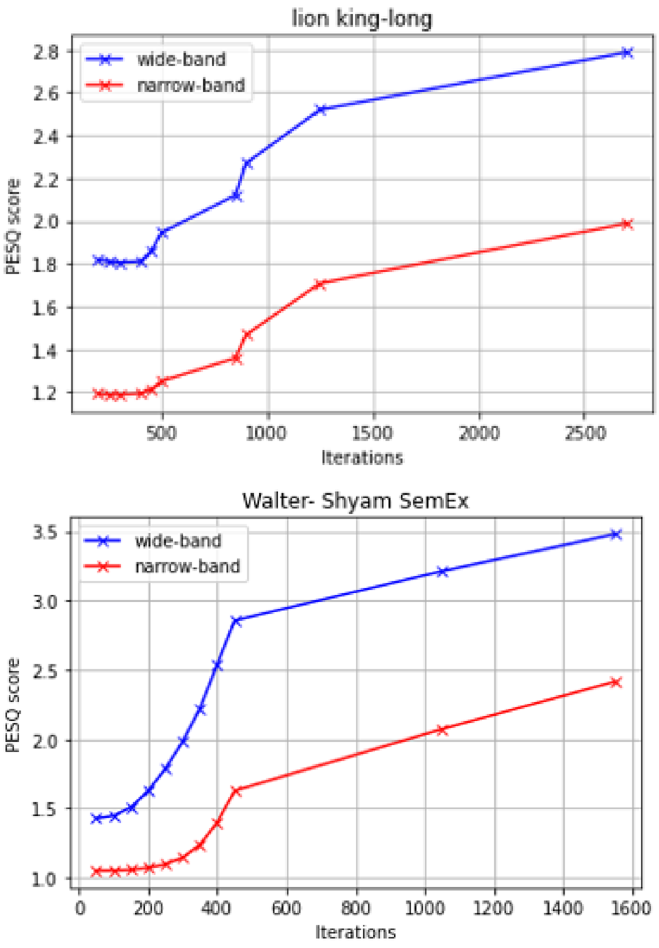

Fig. 4. Variation of PESQ v/s iterations

We also ran few of the above experiments by adding Gaussian noise with SNR 10dB to simulate transmission through a noisy channel. Results can be seen in Table II. When we compare the results (same carrier audio and encrypted text) in presence of Gaussian noise with that without (Table I), we can see a sharp drop in PESQ. This is expected because when we add noise to the perturbations, it acts as a completely different perturbation. As a result, it becomes difficult to reduce the amplitude of original perturbation in order to improve the PESQ score. Figure 5 compares the PESQ variation with and without noise for same audio files and target text pair.

Source	Sr No	Audio Duration	PPS	PESQ WB (best)	PESQ NB (best)
TIMIT	1	3	6.66	1.42	1.1
Breaking Bad	3	11	2.7	2.13	1.41
Oppenheimer	6	4	1.75	2.17	1.4

TABLE II
PESQ OBTAINED BY ADDING GAUSSIAN NOISE. SR. NO. CORRESPOND TO THE AUDIO-TARGET SR. NO. IN TABLE V

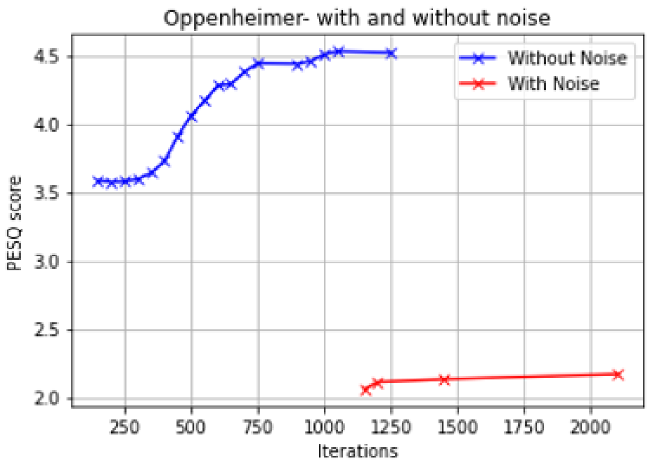

Fig. 5. Variation of PESQ when encrypted with and without the presence of Gaussian noise.

Lastly, we encode a random string of phones and check the PESQ. We found that it is very difficult to encode this random target. Thus, higher the perplexity of the text to encode, the more difficult it is to find the right perturbation. This has important implications for the kind of text that can be encoded. Check figure 6.

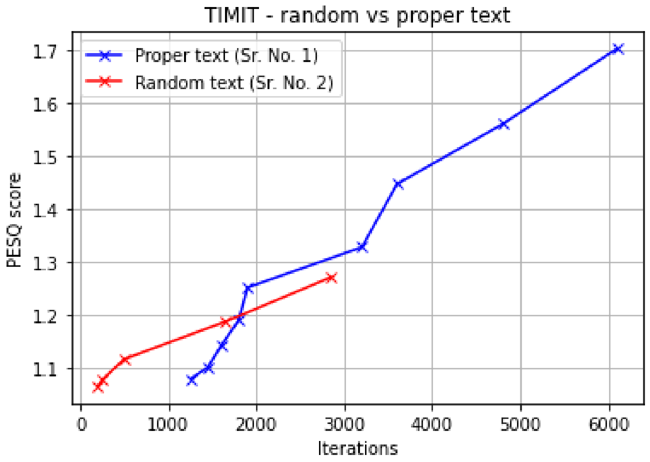

Fig. 6. Trends in PESQ as a function of the nature of encoded text. Proper text refers to a grammatically-correct English sentence while random refers to a random string of phones which is encoded.

We also ran some experiments with perturbation in the log spectrogram domain instead of the time domain (Table III). When compared to the results of perturbation in time domain, we can see a degradation in PESQ value. This is because the phase information of the spectrum is lost, and the Griffin-Lim inverse transform present in the torchaudio library can only approximate it. Circumventing this issue is a potential direction for future work.

Source	Sr No	Audio Duration	PPS	PESQ WB (best)	PESQ NB (best)
Oppenheimer	5	4	8	2.52	1.97
Oppenheimer	6	4	1.75	2.97	2.44

TABLE III
RESULTS OF EXPERIMENTS WITH PERTURBATIONS IN LOG SPECTROGRAM. SR NO CORRESPOND TO THE AUDIO-TARGET PAIR IN TABLE V

IV. CONCLUSION

In this work, we successfully implemented the algorithm presented in [1]. Moreover, we studied various trends related to the number of iterations, the nature of the encoded text, duration of carrier audio and the PESQ score. We observed that noisy signals serve as excellent carrier audio because the perturbation can be easily hidden in the existing noise. On the other hand, it is very difficult to encode the perturbation in a clean carrier. Also, texts with high perplexity are difficult to encode.

V. FUTURE WORK

We plan to extend the work in the following directions:

- Make our scheme immune to noisy channels i.e. the model should be able to recover the encoded text even after addition of some noise to the perturbed signal e.g. Gaussian noise. [8] talks about a few suggestions.
- Train a discriminator to distinguish between the perturbed audio and the clean audio. This could help improve the imperceptibility of the perturbed signal. Alternatively, we could also make our scheme robust to Gaussian noise by training a discriminator to distinguish between Gaussian noise samples and the perturbations. By doing so, our scheme can generate a perturbation which does not resemble Gaussian noise and can hence be immune to noisy channels.
- Try out other ASR models. It is important to note that a better model (in terms of metrics such as PER) need not always be the best model for our task. In fact, we want our model to be *highly susceptible to adversarial attacks* so that any given text can be quickly encoded with a small perturbation. This could be achieved by training the ASR model on some stego pairs.
- Extend the spectrogram-based technique by incorporating the work done in [9].

ACKNOWLEDGMENT

We would like to thank Prof. Preethi Jyothi for the references, feedback regarding the experiments and guiding us through the exciting field of Automatic Speech Recognition.

REFERENCES

[1] Kong, Yehao, and Jiliang Zhang. ”Adversarial audio: A new information hiding method and backdoor for dnn-based speech recognition models.” arXiv preprint arXiv:1904.03829 (2019).

[2] Jadhav, Shwetavinayakarao, and A. M. Rawate. ”A New Audio Steganography with Enhanced Security based on Location Selection Scheme.” International Journal of Performability Engineering 12.5 (2016).

[3] Xiang, Yong, et al. ”Spread spectrum audio watermarking using multiple orthogonal PN sequences and variable embedding strengths and polarities.” IEEE/ACM Transactions on Audio, Speech, and Language Processing 26.3 (2017): 529-539.

[4] Hua, Guang, Jonathan Goh, and Vrizlynn LL Thing. ”Time-spread echo-based audio watermarking with optimized imperceptibility and robustness.” IEEE/ACM Transactions on Audio, Speech, and Language Processing 23.2 (2015): 227-239.

[5] Ngo, Nhut Minh, and Masashi Unoki. ”Method of audio watermarking based on adaptive phase modulation.” IEICE transactions on information and systems 99.1 (2016): 92-101.

[6] Avci, Derya, Turker Tuncer, and Engin Avci. ”A new information hiding method for audio signals.” 2018 6th International Symposium on Digital Forensic and Security (ISDFS). IEEE, 2018.

[7] Mendes, Ethan and Kyle Hogan. “Defending Against Imperceptible Audio Adversarial Examples Using Proportional Additive Gaussian Noise.” (2020).

[8] Ren, Kui Zheng, Tianhang Qin, Zhan Liu, Xue. (2020). Adversarial Attacks and Defenses in Deep Learning. Engineering. 6. 10.1016/j.eng.2019.12.012.

[9] Qin, Y., Carlini, N., Cottrell, G., Goodfellow, I. Raffel, C.. (2019). Imperceptible, Robust, and Targeted Adversarial Examples for Automatic Speech Recognition. Proceedings of the 36th International Conference on Machine Learning, in Proceedings of Machine Learning Research 97:5231-5240 Available from http://proceedings.mlr.press/v97/qin19a.html

[10] Carnegie Mellon University, CMUdict- The CMU Pronouncing Dictionary version 0.7b. Available from http://www.speech.cs.cmu.edu/cgi-bin/cmudict

Appendix A

Table IV contains more details about the original audios and the encrypted text mentioned in Table I.

Dataset	Original Audio	Sr. No	Encrypted Text
TIMIT	Shaving cream is a popular item on halloween	**1**	Nuclear strike authorized (20 phones)
		2	15 random phones
Breaking Bad	I am not in danger Skylar; I am the danger	**3**	Shyam, how is your semester exchange (30 phones)
		4	15 random phones
Oppenheimer	Now I am become death, the destroyer of worlds	**5**	Hey Alexa, add a TV to my shopping list (32 phones)
		6	Code red (7 phones)
Lion King	To change the future you gotta put the past behind you, way behind. Look kid, bad things happen but you cannot do anything about it. Right? WRONG! When the world turns its back on you, you turn your back on the world and only embrace what's next and turn the what into so what	**7**	In winter that seat is close enough to the radiator so it's warm yet not so close that he sweats. In summer it's directly in the path of cross breeze. It faces the television at an angle that isn't direct so he can still talk to everybody yet not so wide that the picture looks distorted (103 phones)
		8	That's my spot (13 phones)

TABLE IV