Transferable and Configurable Audio Adversarial Attack from Low-Level Features

Anonymous Author(s)
Affiliation
Address
email

Abstract

Recent works revealed that state-of-the-art machine learning based Automatic Speech Recognition systems (ASR) have a considerable vulnerability to the crafted adversarial examples. However, limited by individual ASR system's specific machine learning models, the current audio adversarial attacks still lack certain model transferability as well as configurability for different deployment scenarios. In this work, we propose a novel untargeted adversarial example generation method to ASR systems, which shifts the adversarial example generation from the high-level machine learning models to the low-level feature extraction stage. By taking advantage of the fundamental impact and direct configuration of the low-level features, the proposed method can generate transferable and configurable adversarial examples for ASR system perturbation. During the evaluation, we use 6 commercial ASR models to test the proposed attack method. The results show that the proposed method can achieve strong transferability and outstanding perturbation effectiveness. Also, it can configure the adversarial examples with desired audio attributes for better scenario adaptation capability.

1 Introduction

Rapid progress in the machine learning technologies have largely promoted the performance of Automatic Speech Recognition systems (ASR). However, recent research works have shown that the machine learning models in the ASR systems can be easily perturbed by the adversarial examples and therefore mislead the systems to incorrect recognition results. Many works have been proposed [1,2,3,4,5,6,7], and most of them share a same methodology, which applies the backpropagation algorithm through the ASR machine learning models (e.g. Recurrent Neural Network (RNN)) to cast the logit errors to input data. However, because of the huge variance among different machine learning models and the indirect backpropagation casting process through the models, these methods fail to generate adversarial examples with strong model transferability to attack arbitrary ASR systems, and can't directly configure the adversarial examples with desired audio attributes.

In this paper, we propose a novel untargeted adversarial attack method to address these two issues. Different with previous works focusing on the machine learning models, we apply the adversarial example generation on the low-level feature extraction stage. Specifically, we use Mel-Frequency Cepstral Coefficient (MFCC) features as low-level features which transfer the input audio waveform to MFCC feature vectors. Shared by all the ASR systems, low-level feature extraction stage like MFCC has the fundamental impact to later high-level machine learning models. Therefore, the adversarial examples generated from the low-level features are expected to have strong transferability for different ASR systems. Meanwhile, the adversarial example generation over the MFCC stage can direct regulate the audio attributes and achieve flexible attack configuration. During the evaluation, we evaluate our proposed attack method on multiple commercial ASR systems (e.g. Google Voice). The results show that the proposed method can achieve strong transferability and and outstanding

Submitted to 32nd Conference on Neural Information Processing Systems (NIPS 2018). Do not distribute.

perturbation effectiveness. Also, it can configure the adversarial examples with desired audio attributes for better scenario adaptation capability.

2 Low-Level Feature based Attack Method

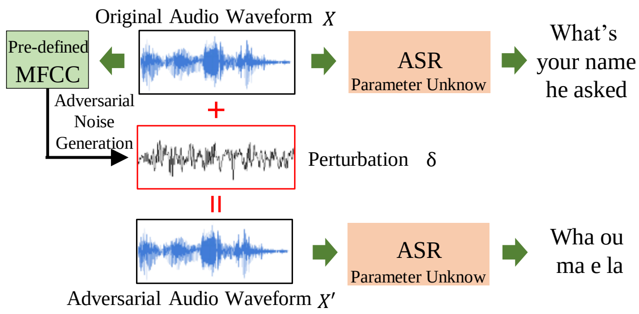

Figure 1: Adversarial Example Generation Process

2.1 General Adversarial Attack Definition Typically, the ASR process can be formulated as $f(\cdot)$, while the original input waveform is X, and the recognition result is $f(X) = Y$. When superposing a human imperceptible noise δ on the original input waveform X, we can get adversarial example $X + \delta$, and the recognition result is expected to be changed as $f(X + \delta) \neq Y$.

According to our preliminary experiment, by considering both perturbation performance and method implementation convenience, Basic Iterative Method (BIM) [9] is utilized in there to iteratively generate adversarial noise through backpropagation and can be formulated as:

$$\begin{aligned} X_0 &= X, \\ X_{N+1} &= Clip\{X_N + \delta_N\}, \\ \delta_N &= \bigtriangledown J(\theta, X_N, Y_w), \end{aligned} \tag{1}$$

where θ is the parameters for ASR process $f(\cdot)$, Y_w indicates the desired manipulation result. δ_N is the adversarial perturbation generated in the Nth iteration. *Clip* means that the generated adversarial examples will be limited in a certain strength range. J is the cost function that measures the difference between $f(X_N + \delta_N)$ and Y_w, and $\bigtriangledown$ represents the partial differentiate process. Previous works generate δ_N by attacking $f(\cdot)$ through the high-level machine learning models as Eq. 1 shows, thus the δ_N is highly related to specific model. Therefore, such a methodology defects the transferability of the adversarial attacks.

2.2 MFCC Based Adversarial Example Generation

In this part, we will describe our attack approach including method formulation, MFCC process analysis and final generation method design in detailed.

In our method, which can be shown in the Fig. 1, instead of attacking $f(\cdot)$, we use a MFCC process which formulates as $f_P(\cdot)$ to generate adversarial examples. The parameter of $f_P(\cdot)$ is pre-defined. Since $f_P(\cdot)$ is not part of $f(\cdot)$, our method can be considered as a black-box attack.

During the adversarial example generation, we need to use backpropagation to differentiate the cost function J. Therefore, it is necessary to formulate and integrate MFCC process to facilitate our method design. For MFCC process, it transforms an input speech waveform into feature vectors composed of coefficients of Mel-Frequency Cepstrum by following 6 steps. *1)* Speech waveform X is preprocessed to X_s and further be pre-emphasized as speech vector y^{pre} according to the equation $y^{pre} = X_s - \alpha X_{s-1} \quad (WIN)$. *2)* y^{pre} is further divided into N^{fra} frames y^{fra} with frame length n. *3)* A hamming windowing function is applied to each frame: $y^{win} = \left\{0.54 - 0.46cos(\frac{2\pi(n-1)}{N^{fra}-1})\right\} \times y^{fra} \quad (WIN)$. *4)* Each frame y^{win} do a N^{FFT} points Fast Fourier-Transforming and calculate the power spectrum by using equation: $y^{FFT} = \frac{1}{N^{FFT}} |(\sum_{n=1}^{N^{FFT}} y^{win} e^{-j2\pi kn/N^{FFT}})^2|, \quad 1 \leq k \leq K \quad (FFT)$, where K is the total frequency points. *5)* A set of Mel-Filter vectors $MB(f_1, f_2, ..., f_L)$ are applied to the power spectrum y^{FFT} and the Mel-power spectrum y^{Mel} can be obtained according to: $y^{Mel} = y^{FFT} \times MB(f_1, f_2, ..., f_L), \quad 1 \leq l \leq L \quad (MFB)$, where L is number of filters. *6)* We apply Discrete Cosine Transform to calculate MFCC features Y^{MFCC}: $y^{MFCC} = y^{Mel} cos[(l - 0.5)\frac{\pi l}{L}] \quad (DCT)$. With the aforementioned 6 steps, the input speech waveform X can be transfered as MFCC feature vectors of Y^{MFCC}, which offers speech perceptive features for the ASR process.

In our case, we first take Y^{MFCC} into J in Eq. 1. Then we replace $f_P(\cdot)$ and θ with parameters in MFCC process. Finally, we set Y_w to 0 for maximizing the perturbation effectiveness regardless the

Table 1: WER Performance of MFCC and CTC

Iteration	1	10	100	1000
MFCC	**56.5%**	**84.6%**	**100%**	100%
CTC	36.0%	77.1%	93.3%	100%

speech content. Furthermore, according to the chain rule of partial differentiation, we differentiate J from *DCT* process to *PRE* process step by step. Such a differentiation can be formulated as:

$$\begin{aligned} \delta^{MFCC} =& DCT'(\cdot){\cdot}MFB'(\cdot){\cdot}FFT'(\cdot){\cdot}WIN'(\cdot){\cdot}PEM'(\cdot), \\ &(\quad X{+}\delta^{MFCC} \to 0 \quad and \quad \delta^{MFCC} < T_{adv}). \end{aligned} \tag{2}$$

As the derivation value of equations in step 3 and step 5 are constants, which can be obtained directly, Eq. 2 can be simplified as:

$$\begin{aligned} \delta^{MFCC} =& MB(f_1, f_2, ..., f_L){\times}\left\{0.54{-}0.46cos(\frac{2\pi(n{-}1)}{N^{fra}{-}1})\right\} \\ &{\times}DCT'(\cdot){\cdot}FFT'(\cdot){\cdot}PEM'(\cdot), \\ &(\quad X{+}\delta^{MFCC} \to 0 \quad and \quad \delta^{MFCC} < T_{adv}), \end{aligned} \tag{3}$$

where δ^{MFCC} is the generated adversarial noise and T_{adv} is its strength constraints.

By use Eq. 3, we can obtain the adversarial noise in each iteration. With the number of iteration increase, the adversarial noise will approach to a best perturbation performance which we will show in the experiment section.

2.3 Adversarial Example Configuration In this part, we further explore the configurability of generated adversarial example in our proposed method. During the process of adding adversarial noise δ into original input waveform X, many practical constraints should be taken into consideration such as frequency range configuration. Therefore, generated adversarial examples should have strong configurability with respect to these different practical constraints.

However, because of indirect and long casting process through the machine learning model, traditional methods cannot regulate input waveform precisely. On the contrary, MFCC process has simple and short casting during the backpropagation. Therefore, we can accurately regulate audio attributes in the input waveform by adding regulation directly in the MFCC process.

The configurability of adversarial example has significant potential for different applications. We can take human hearing perception quality as a case study. We leverage two auditory masking effects to reduce the impact of adversarial noise on human hearing perception: 1) The significant sensitivity frequency range of human hearing perception is from 200*Hz* to 5*KHz*. 2) Frequency component with higher sound intensity C^H may prevent its adjacent lower frequency component C^L from human perception, which can be formulates as:

$$\mathbf{D}(C^H) = 0, \quad if \quad C^L < C^H, \tag{4}$$

where **D** represents the human perception system.

During the FFT step of MFCC process, we can get multiple y^{FFT} and each of them represents certain frequency band. So, we first prevent y^{FFT} which represent the frequency range from 20*Hz* to 20k*Hz* from being differentiated during the backpropagation process. Then, we locate the top t% frequency component with the highest sound intensity (empirically, t≈10). Adversarial noises are further generated around those frequency components. We will evaluate the performance of this configurability in the experiment section.

3 Evaluation

The proposed method is implemented on the Tensorflow platform [11], and evaluated on a desktop server equipped with Intel Xeon and NIVIDA 1080. During the implementation, the MFCC parameter configuration is adopted from [2][10], the original speech data is from the Common Voice Dataset [12], and the rest of ASR system is based on DeepSpeech platform [10]. During the evaluation, 6 different state-of-the-art ASR systems are considered to evaluate the effectiveness and transferability.

3.1 Perturbation Effectiveness and Efficiency To evaluate the proposed method, we first compare the perturbation of our low-level feature based to one machine learning model based adversarial attack,

Table 2: WER of MFCC, CTC and Original on 6 ASR Models

	Google	Sphinx	Wit.ai	Microsoft	Houndify	IBM
Original	8%	21.2%	19.2%	15.8%	20.9%	18%
MFCC	**51%**	**77.8%**	**62.3%**	**67.8%**	**72.1%**	**63.9%**
CTC	16.3%	51.1%	33.8%	38.5%	40.9%	31.5%

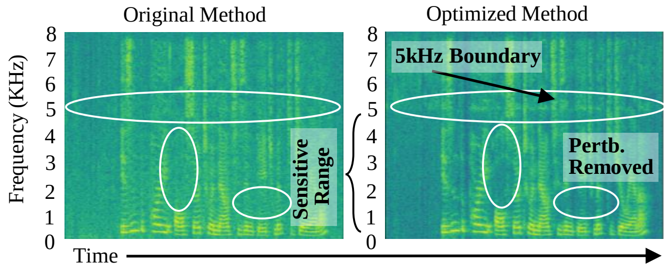

Figure 2: Feature configuration Example with Human Hearing Quality

which is referred as the CTC attack. The CTC attack is a state-of-the-art method that attacks the high-level CTC features and generates adversarial examples from RNN models [2]. As the proposed MFCC based method is designed as an untargeted attack, the CTC is also configured for misleading the ASR results to random noises. Table 1 shows the perturbation effectiveness on DeepSpeech model by comparing the WER achieved by 2000 adversarial examples from each method, among which every 500 examples are generated with different amounts of iterations. For the limitation of the adversarial noise strength T_{adv}, we set it as $28dB$. With the iteration incremented from 1 to 1000, the proposed MFCC based attack could achieve an WER of 56.5%$\sim$100%, and the CTC attack achieves 36.0%$\sim$100%. However, the MFCC based attack only takes 100 iterations to achieve the 100% WER, while the CTC attack needs 10 times more effort. From such a comparison, we can tell that the MFCC based attack can effective achieve the same perturbation effectiveness as the machine learning based method, but with significant efficiency improvement.

3.2 Attack Transferability to Different ASR Systems To further evaluate the transferability of the proposed MFCC based attack, another 6 different state-of-the-art ASR systems are also tested (*i,e,* Google Voice, Sphinx, Wit.ai, Microsoft, Houndify, and IBM). 500 adversarial examples are generated respectively from the proposed MFCC based and CTC attack method with 1000 iterations on DeepSpeech system. The original examples are also tested as the baseline. Table 2 illustrates the experiment results. In Table 2, different ASR systems have different recognition performance with varying WERs of 8%$\sim$21.2%. The proposed method can effectively maintain a high WER over 50% (51%$\sim$77.8%) over different ASR systems, while the WER of CTC attack drops to 16.3%$\sim$51.1%. Therefore, the proposed MFCC based attack method demonstrates strong model transferability.

3.3 Attack Configurability In this part, we will evaluate the configurability under the case of human hearing perception quality. Since the sampling rat of audio samples in Common Voice Dataset is 16kHz, fully frequency range for each audio samples will be from $0Hz$ to 8kHz. During the FFT stage in MFCC process, there are 257 value numbers and each of them represents a frequency bands around $31Hz$. Then we do the configuration according to the Section 2.3 and the result is shown in the Fig. 2. The left one is the frequency spectrum of generated adversarial example without configuration, while the right one is generated after configuration. We can clearly find that much adversarial noise are restricted outside of human sensitive range which indicated by a clear boundary presents around 5kHz and white circles in sensitive frequency range. Also, the adversarial noise remain in the sensitive range is more concentrated to the speech component with high sound intensity.

4 Conclusion

In this work, we proposed a transferable and configurable audio adversarial attack method. By generating adversarial examples from low-level features in the ASR system, we show that transferable adversarial examples can be well generated in the fundamental stage before the machine learning models. Also, without complex backpropagation process through the machine learning models, the proposed method can directly configure the adversarial examples with desired audio attributes for better scenario adaptation. The proposed attack method can be well utilized as an effective and efficient non-target attack method, which can be well deployed in scenarios of transferable attack, black-box attack, and even audio encryption against undesired analysis.

References

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

[2] Carlini, N. and Wagner, D. (2018) Audio adversarial examples: Targeted attacks on speech-to-text. *arXiv:1801.01944*.

[3] Alzantot, M., Balaji, B. and Srivastava, M. (2018) Did you hear that? adversarial examples against automatic speech recognition. *arXiv:1801.00554*.

[4] Taori, R., Kamsetty, A., Chu, B. and Vemuri, N. (2018) Targeted Adversarial Examples for Black Box Audio Systems. *arXiv:1805.07820*.

[5] Yuan, X., Chen, Y., Zhao, Y., Long, Y., Liu, X., Chen, K., Zhang, S., Huang, H., Wang, X. and Gunter, C.A. (2018) CommanderSong: A Systematic Approach for Practical Adversarial Voice Recognition. *arXiv:1801.08535*.

[6] Kreuk, F., Adi, Y., Cisse, M. and Keshet, J. (2018) Fooling End-to-end Speaker Verification by Adversarial Examples. *arXiv:1801.03339*.

[7] Gong, Y. and Poellabauer, C. (2017) Crafting Adversarial Examples For Speech Paralinguistics Applications. *arXiv:1711.03280*.

[8] Këpuska, V.Z. and Elharati, H.A. (2015) Robust speech recognition system using conventional and hybrid features of mfcc, lpcc, plp, rasta-plp and hidden markov model classifier in noisy conditions. Journal of Computer and Communications, 3(06), p.1.

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

[10] Hannun, A., Case, C., Casper, J., Catanzaro, B., Diamos, G., Elsen, E., Prenger, R., Satheesh, S., Sengupta, S., Coates, A. and Ng, A.Y. (2014) Deep speech: Scaling up end-to-end speech recognition. *arXiv:1412.5567*.

[11] Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M. and Kudlur, M. (2016) Tensorflow: a system for large-scale machine learning. in Proceedings of *the USENIX Symposium on Operating Systems Design and Implementation* (OSDI), Vol. 16, pp.265-283

[12] Mozilla. (2018) Common Voice. https://voice.mozilla.org/en

[13] Google. (2018) Google Cloud Speech-to-Text. https://cloud.google.com/speech-to-text/

[14] Sphinx. (2018) CMUSphix. https://cmusphinx.github.io/wiki/

[15] Wit.ai. (2018) Wit.ai. https://wit.ai/

[16] Microsoft. (2018) Microsoft Bing Voice Recognition. https://azure.microsoft.com/zh-cn/services/cognitive-services/speech/

[17] SoundHound. (2018) Houndify API. https://www.houndify.com/

[18] IBM. (2018) IBM Speech to Text. https://www.ibm.com/watson/developercloud/speech-to-text.html