Adversarial Training for Automatic Speech Recognition Systems

Gian Guido Parenza
Department of Information Technology and Electrical Engineering (D-ITET)
Zurich, Switzerland
parenzag@student.ethz.ch

Tim Gretler
Department of Information Technology and Electrical Engineering (D-ITET)
Zurich, Switzerland
tgretler@student.ethz.ch

Hendrik Klug
Department of Information Technology and Electrical Engineering (D-ITET)
Zurich, Switzerland
klugh@student.ethz.ch

***Abstract*—In the past years, it has been shown that neural networks are vulnerable to adversarial examples: inputs specifically designed to produce a misclassification. Previous work has shown that any given audio, including music, can be perturbed such that it is recognised as any desired phrase or silence by an automatic speech recognition system. Furthermore, recently it has been shown that such adversarial samples can be made imperceptible to humans. This was a breakthrough in the field. Before it, the samples generated could be easily identified as adversarial by humans. To the best of our knowledge, it has only been shown that these adversarial examples can be successfully generated, however no work has yet shown that an automatic speech recognition system can be trained and made robust to such attacks. In this work, we first produce imperceptible audio adversarial examples on arbitrary full-sentence targets by means of a simple but effective. Next, we train Deep Speech 2, a state-of-the-art model in speech-to-text conversion tasks. Finally, we show how the performance of the model trained with adversarial audios did not worsen, whilst it has achieved higher scores with respect to the original model on the adversarial samples.**

I. Introduction

Deep learning models have achieved state-of-the-art results on a large set of fields, from computer vision to natural language processing. Some have also been shown to be working in different settings on a wide variety of datasets with little modifications to the original structure. One of these is Deep Speech 2, a state-of-the-art model in speech-to-text conversion tasks developed by Baidu AI Research Lab, which has been originally designed for end-to-end speech recognition in both English and Mandarin, but can also be easily modified and used with other languages [1].

These models, despite having achieved high generalisation performances, are also known to learn uninterpretable solutions that could have counter-intuitive properties. More specifically, two intriguing properties of neural networks have been brought to the attention of the machine learning community by Szegedy et al: the entire space of activation, rather than the individual units, contains the semantic information, and that neural networks learn input-output mappings that are discontinuous. In particular, the latter has left a footprint in the research field introducing the now well-known term of "adversarial examples" [2].

Since the work of Szegedy et al, more studies have tried to understand the nature and explain the existence of these adversarial examples. Two attempts to answer the questions posed by Szegedy et al were proposed by Schmidt et al. 2018 and Bubeck et al. [3], [4]. The former suggests that for classifiers to be robust to such attacks, they require to be tailored to the specific dataset, whereas the latter suggests that classifiers are vulnerable to these perturbations not because of theoretical limitations, but because of computational constrains.

More recently, adversarial examples became of interest also in the audio domain. Of particular relevance was the work of Nicholas Carlini and David Wagner, who have demonstrated how it is possible to generate inaudible adversarial audio examples [5]. The task of making the distortion inaudible is analogue to the one on images, where minimising the distortion between an image and the nearest misclassified example yields a visually indistinguishable image. For the audio samples this

is not the case [6]. Thus, to make the audios sound like the original ones, the human perceptibility to audio has to be taken into account. The way in which Carlini and Wagner were generating the samples was making use of an iterative method that maximises the Connectionist Contemporal Classification (CTC) loss under the constraint of keeping the distortion as "quiet" as possible. In particular, the distortion metric they were using for their tests was measured in Decibels (dB):

$$dB_x(\delta) = dB(\delta) - dB(x) \tag{1}$$

$$dB(x) = \max_i 20 \log_{10}(x_i) \tag{2}$$

which was set to be smaller than a certain amount of Decibels.

In a second paper, Yao Qin et al. made advancements in both making imperceptible audio adversarial examples and constructing perturbations that are effective also when played over-the-air [7]. The major change from the previous work is that they have made use of the psychoacoustic principle of auditory masking [8]. More specifically, they make use of *frequency masking*, where a louder signal - the "masker" - can make signals of the nearby frequencies imperceptible. In practice, given the Short-Time Fourier Transform of the audio, the power spectral density is computed. This is then used to identify the global masking threshold $\theta_x(k)$. Given this parameter, when a perturbation is added to the audio and it is below the frequency masking threshold of the naïve audio, this will then be inaudible. Lastly, given the difficulty in generating the adversarial examples due to the lack of a constraint on the magnitude of the perturbation, a two-stage attack is applied.

Similarly to the work of Qin et al, we designed our attacker in a way such that it takes into account the psychoacoustic principles of auditory masking. Additionally, we make use of a single-stage optimisation and consider a ℓ_∞-bounded distortion. Furthermore, we perform adversarial training of the model as described by Madry et al. and evaluate the results [9]. Lastly, we discuss the results and show the benefits of adversarial training.

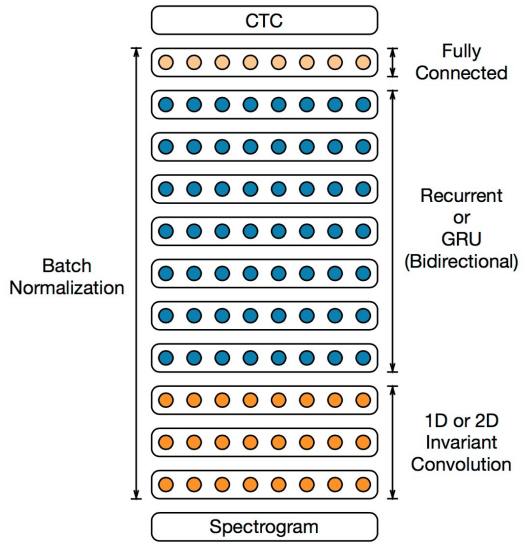


Fig. 1. Architecture of DeepSpeech 2. Various variants of this architectures have been explored in the original paper. Image taken from [1]

II. Models and Methods

In this work, our aim is to generate imperceptible untargeted adversarial samples. In agreement with the definition of Qin et al., this corresponds to finding a perturbation δ which when added to the input induces classifier to produce general spelling errors. Furthermore, the perturbation produced is such that a human cannot distinguish between adversarial and the original audio.

A. The ASR Model

The model we use to investigate the adversarial training is Deep Speech 2, more precisely the implementation of Sean Naren[1] which has already been pretrained on the dataset Libri Speech[2] [10]. The model used is shown in Figure 1 and consists of a serial stack of 2 convolutional layers, which take as input the magnitude of the spectrogram of the audio, followed by 5 bidirectional Long Short-Term Memory (LMST) units. The output consists of a Fully Connected Layer containing 29 nodes that correspond to the 26 characters of the English alphabet and 3 additional tokens corresponding to *blank*, which is used to denote a stretch of silence, *space* and *apostrophe*, to determine word boundaries.

[1]https://github.com/SeanNaren/deepspeech.pytorch
[2]http://www.openslr.org/12/

B. The CTC Loss

To train the model, we make use of the Connectionist Temporal Classification (CTC) loss [5], [11]. This loss is suited for a particular class of neural networks that perform sequence-to-sequence tasks where the alignment between input and output is not known.

The loss is computed in the following way: given a sequence as input (the spectrogram), it returns the probability over the output domain for each element of the sequence (the 29 tokens). Next, the sequence is reduced by removing duplicated tokens (e.g. repetition of the same character $h\ e\ l\ l\ o \rightarrow h\ e\ l\ o$), and the special tokens (e.g. *blank*). Once the sequence is reduced and aligned to the label, it is possible to compute the loss as follows:

$$\textit{CTC-Loss}(f(\mathbf{x}), \mathbf{p}) = -\log Pr(\mathbf{p}|f(\mathbf{x})) \quad (3)$$

Where $f(\mathbf{x})$ gives the probability over the characters given the input and $Pr(\mathbf{p}|f(\mathbf{x}))$ is the probability of a given phrase $\mathbf{p}$ under the distribution $\mathbf{y} = f(\mathbf{x})$. This is described in more detail in the original paper [11]. To decode the output of the network, we make use of a *Greedy Decoder*, which searches for the most likely alignment. This solution is not the best possible, an alternative is to use a *Beam Search Decoder*. which evaluates multiple alignments and evaluates the most likely one [5].

C. Adversarial Examples Generation

To generate the adversarial examples we make use of the "Robustness" library to perform gradient ascent on the CTC-Loss [12]. This procedure defines a particular class of adversary, a *first-order adversary*. The distinction arises from the need to discriminate adversaries that have access to the gradients of the loss function with respect to the input, and the ones that don't. This distinction is also at the base of the difference between *white-box* (complete knowledge of the model and gradient information) and *black-box* attacks. The procedure to generate the adversarial examples follows the standard procedure which is described in Algorithm 1, also known as projected gradient descent (PGD) algorithm.

Of particular interest is the implementation of the projection of the adversarial examples onto a feasible set. This is not implemented in the same way as for computer vision tasks. More specifically, when someone wants to generate adversarial examples to attack models designed for image classification tasks, the adversarial examples are generated by changing the RGB values that form the image within a certain constraint, which is defined based on the ability of humans to distinguish between different shades of the same colour. Here, this metric would not work as intended. Humans have a well-developed ability to distinguish between sounds; hence, to make the adversarial examples indistinguishable Qin et al.

Algorithm 1: Adversarial Example Generation

```
Result: Adversarial Example
initialise the adversarial example as the original input;
for number of steps do
    compute the loss given the adversarial input;
    compute the gradient;
    perform gradient ascent (descent) and update the
      adversarial input;
    project the adversarial input on feasible set;
end
```

Here, we propose a simpler implementation to achieve similar results. The idea is to define a ℓ_∞ ball not directly on each element of the spectrogram, but on the relative difference. Practically, given the ratio between the additional adversarial component and the original spectrogram, this has to be bounded within the "pixel-specific" ℓ_∞-ball. A Pytorch based implementation is shown here below:

```
1 def project(**kwargs):
2     diff = adv_example - original_input
3     diff = torch.clamp(
4             torch.div(diff, original_input + 1e-3),
5             - eps, + eps)
6             )
7     return (diff + original_input).clamp_min(0.0)
```

On line 7, we clip the value of the adversarial example to be positive, because we want to manipulate only the magnitude of the spectral component. In case we allowed for negative values, we would have introduced a π phase shift which would have compromised the reconstruction of the audio.

D. Adversarial Training

The formulation of the problem we are aiming to tackle has been defined by Madry et al. The problem can be viewed as a composition of a *inner*

maximization and *outer maximization problem*. Where the attacker aims to find a perturbation of the original input that results in a high loss (inner maximization), whilst the model is optimized in a way, such that it aims to modify its parameters to minimize the loss given by the inner attack (outer minimization) [9]. Such construction of the problem gives rise to the well-known saddle point problem which can be tackled by making use of the projected gradient descent (PGD) to solve the inner maximization problem, whilst the inner minimization problem can be solved by means of the simple gradient descent algorithm. Practically, to generate our set of adversarial examples we make use of the dedicated and well-documented library: "Robustness", which has been developed by the MIT group "Mardy Lab"[3] and that we have adapted to our needs [12].

The training has been performed in the following way, we have taken a set of δ as constraint, more specifically we have chosen the values: 0.01, 0.1, 1 and 10. The training was limited to 15 epochs due to the large dataset and computational costs. In these epochs, we have set for the first 10 a linear increase of the δ, ranging from 0 to the δ set for training. During the evaluation, we have tested the trained models and the native one on the original audios and the adversarial examples produced for each of the chosen δ.

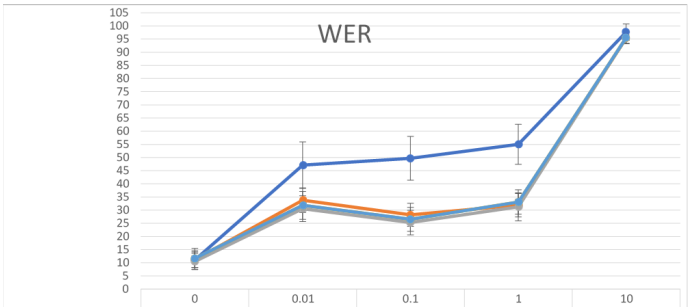
III. Results

A. Dataset

The dataset that we use is Libri Speech, a publicly available dataset derived from audiobooks. The data selected for training have been randomly sampled from the test-clean dataset without restrictions on the duration of the audio.

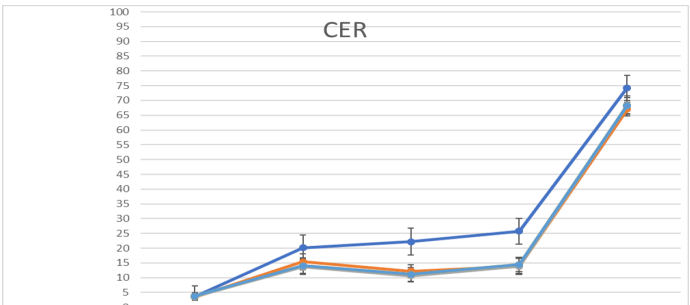
B. Evaluation Metrics

To evaluate the performances of the trained network and compare it to the native version of Deep Speech 2, we report two metrics that are typically used fot ASR systems: the word error rate (WER) and the character error rate (CER). The CER is a metric that provides information about the accuracy of the transcriptions. Given an audio, the characters are reduced to a string where the spaces have been removed. Once the two transcripts are aligned, the characters are compared one-to-one and the CER is calculated as follows: $CER = 100 * \frac{E_C}{N_C}$, where E_C are the misclassified and M_C the total number of characters. The WER[4] is a metric based on the "Levenshtein distance" and is calculated as follows: let S, D and I be the number of substitutions, deletion and insertions of words, then the $WER = 100 * \frac{S+D+I}{N_W}$, where N_W is the total number of words. Even though the WER is not particularly meaningful to define the quality of the model. This metric is important when somebody wants to make a full-text search since it provides the success rate. The results of the previously mentioned evaluation are shown in Figure 2 and 3.



	0	0.01	0.1	1	10
native	11.05329414	47.09987324	49.69124699	55.02490466	97.72828522
adversarial 1	11.18587156	33.80144473	28.19742678	31.90153294	95.42947205
adversarial 10	10.50195705	30.57747368	25.27838452	31.27824481	95.54347417
adversarial 0.1	11.63639384	31.8173666	26.52557496	33.09787967	95.500989
adversarial 0.01	11.63639384	31.8173666	26.52557496	33.09787967	95.50403897

Fig. 2. WER resulting from the evaluation of the different models.



	0	0.01	0.1	1	10
native	3.659772116	20.14118826	22.2115483	25.78570738	74.24767956
adversarial 1	3.519055615	15.36653775	12.16532013	13.9749509	67.07204198
adversarial 10	3.555802815	13.66130776	10.70276901	13.89803585	68.41676544
adversarial 0.1	3.801315783	14.08008352	11.09358399	14.47064745	68.318395
adversarial 0.01	3.801315783	14.08008352	11.09358399	14.47064745	68.31256845

Fig. 3. CER resulting from the evaluation of the different models.

[3]https://github.com/MadryLab/robustness

[4]Contrary to the CER, the WER can be greater than 100%

C. Evaluation of the Adversarial Examples

Here we report the results of the adversarial attack. As described in the previous section we have set the boundary of the ℓ_∞-ball to be "pixel-specific" to emulate the effect of frequency masking achieved by Qin et al. In Figure 4 the magnitude of the Short-Time Fourier Transform of the original audio is shown, whereas Figure 5 shows the difference the adversarial example and the original spectrogram for $\delta = 1$. For the other values, similar results have been obtained. To be specific, for smaller values of δ the differential image is mainly white and is difficult to appreciate the localization of the attack. On the other hand, for $\delta = 10$, the attacker starts to prefer certain locations of the spectrum which make the color bar saturate and make less noticeable the attack in the other regions.

IV. Discussion

In this work we successfully generated inaudible adversarial examples (for $\delta = 0.01$ and 0.1) emulating the results of Carlini et al. but using a different approach which led to an easier implementation and a less costly generation of those. In our experiments, we have trained 4 models, each on the same set of data but with different values for the constraint. These models have been tested together with the original model, that we have assumed as the baseline, on 5 sets of data: the original set, and 4 sets generated each with the aforementioned δ values for the constraint.

The results we have obtained show how the adversarial training made the models more robust to a set of adversarial examples without significantly worsening the performance on the original dataset. Even though the results look promising, for the future more should be done in terms of the parameters' optimization. It would be of interest to see how these models performed with the audios generated using the technique from [7]. Since these are not generated using an ℓ_∞-bounded, they belong to another class of adversarial attacks. Given this difference, even though the method implemented tries to emulate the characteristics of the adversarial examples generated by Qin et al., the model might not be robust to those.

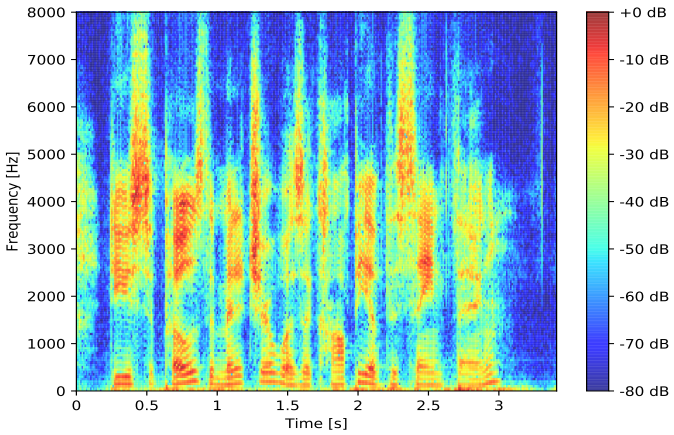


Fig. 4. The spectrogram of the native audio examples.

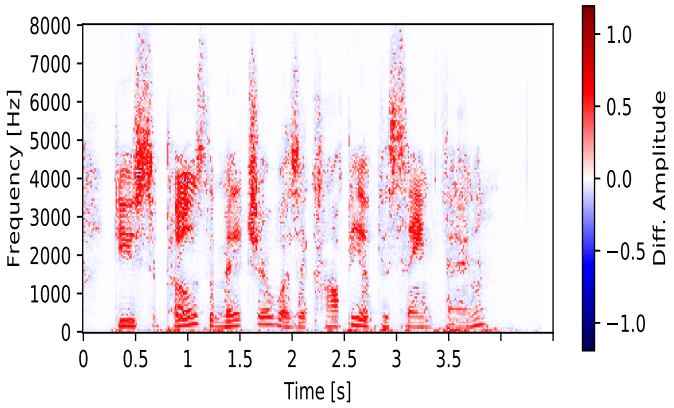


Fig. 5. The spectogram of the difference between the native and the adversary example.

V. Summary

We have shown that it is possible to improve an Automatic Speech Recognition System by training it with adversarial examples and making it more robust to such a class of attacks. The adversarially trained model has significantly lower CER and WER values for adversarial inputs than the original Deep Speech 2 model and shows no significant decrease in accuracy for native inputs.

REFERENCES

[1] D. Amodei, S. Ananthanarayanan, R. Anubhai, J. Bai, E. Battenberg, C. Case, J. Casper, B. Catanzaro, Q. Cheng, G. Chen, *et al.*, "Deep speech 2: End-to-end speech recognition in english and mandarin," in *International conference on machine learning*, pp. 173–182, 2016.

[2] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus, "Intriguing properties of neural networks," *arXiv preprint arXiv:1312.6199*, 2013.

[3] L. Schmidt, S. Santurkar, D. Tsipras, K. Talwar, and A. Madry, "Adversarially robust generalization requires more data," 2018.

[4] S. Bubeck, E. Price, and I. Razenshteyn, "Adversarial examples from computational constraints," 2018.

[5] N. Carlini and D. Wagner, "Audio adversarial examples: Targeted attacks on speech-to-text," *2018 IEEE Security and Privacy Workshops (SPW)*, May 2018.

[6] L. Schönherr, K. Kohls, S. Zeiler, T. Holz, and D. Kolossa, "Adversarial attacks against automatic speech recognition systems via psychoacoustic hiding," *CoRR*, vol. abs/1808.05665, 2018.

[7] Y. Qin, N. Carlini, I. Goodfellow, G. Cottrell, and C. Raffel, "Imperceptible, robust, and targeted adversarial examples for automatic speech recognition," *arXiv preprint arXiv:1903.10346*, 2019.

[8] Y. Lin and W. H. Abdulla, "Principles of psychoacoustics," in *Audio Watermark*, pp. 15–49, Springer, 2015.

[9] 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.

[10] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, "Librispeech: an asr corpus based on public domain audio books," in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 5206–5210, IEEE, 2015.

[11] 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*, pp. 369–376, ACM, 2006.

[12] L. Engstrom, A. Ilyas, S. Santurkar, and D. Tsipras, "Robustness (python library)," 2019.