Assessing and enhancing adversarial robustness in context and applications to speech security

Raphael Olivier

May 2023

Abstract

Designing artificial intelligence models that are robust to adversarial perturbations of their inputs is a highly desirable objective, as such models would not only be less prone to security breaches but also better aligned with human reasoning and perception. Neural Automatic Speech Recognition (ASR) is vulnerable to these adversarial perturbations but currently lacks defenses with strong evidence for robustness to state-of-the-art attacks. This contrasts with image tasks for which several defense algorithms have shown increasingly good results even against adaptive adversaries. Another odd aspect of speech robustness is the lack of transferability between different models for most adversarial attacks. The specifics of ASR as a task certainly play an important role in this odd situation: at their core speech utterances are time series in an infinite-dimensional Hilbert space with an underlying semantic structure, while transcription outputs belong to an infinite metric space.

The objective of this thesis is to explore the dependencies between adversarial perturbations and the context in which it is studied (nature of the task, set of outputs, etc.), in order to assess their threat against speech recognition models and the possibility for robust ASR. Specifically, (1) we show that adversarial robustness is closely related to training methods and model architectures. We propose novel methods to study this relationship without the cost of training robust models, by measuring the randomness in robustness on undefended models and by quantifying the number of adversarial perturbations on a hypersphere. (2) we quantify the threat posed by adversarial attacks on ASR, both under white-box and black-box threat models. We introduce an evaluation framework and apply it notably to show that self-supervised speech models are uniquely vulnerable to transferable attacks. We also take into account the specifics of voice assistants trained on user data and investigate applications of adversarial perturbations for privacy attacks. (3) we investigate two approaches to increase ASR robustness. First, we show that the Randomized Smoothing defense paradigm enables us to combine the strengths of general machine learning defense guarantees and domain-specific speech processing tools. Using it we achieve state-of-the-art ASR robustness against state-of-the-art white-box attacks for several model architectures. Then we introduce a novel framework called *adversarial masked prediction*, that lets us robustly pretrain self-supervised speech models.

Contents

Appendices 128

Chapter 1

Introduction

Artificial intelligence models nowadays show impressive performance in many domains. When evaluated on standardized benchmarks, they may display comparable if not superior performance to humans. Yet in some aspects, these models are extremely different from humans. Perhaps the most famous of these differences is their vulnerability to *adversarial examples*, modified inputs that would seem indistinguishable from naturally occurring inputs but would trigger erratic behaviors from the models. Designing systems that are robust to such perturbations is considered a desirable achievement for several reasons, the main one being security concerns. Agents can generate adversarial examples using gradient-based or black-box methods, and an agent with malicious intent could use them to force models to fail in critical situations. The possibilities are many and worrying: fooling voice activation systems, making cars crash by hiding road signs, or having army drones confuse hospitals and military bases. More generally, adversarially robust models have also been shown to better align with human reasoning and perception and can be seen as a weak but well-defined form of *alignment* between the AI model's actual and intended behavior. Unfortunately, even in simple settings, adversarial robustness is very difficult to achieve, and partial successes have always come with a drop in performance.

One important hurdle on the road to robust models is the fact that adversarial examples are still not completely understood to this day. Numerous works have proposed "explanations" of the phenomenon, but they tend to contradict each other on core aspects of the problems. Are adversarial examples artifacts of imperfect learning algorithms, or relevant patterns that play a role in model performance? Do they arise from overparametrized model classes like deep neural networks, or from the datasets these models are trained on? Is adversarial robustness fundamentally at odds with performance or contributing to it? Most of those points of view are supported by empirical evidence, despite contradicting each other. That apparent paradox reveals a lack of knowledge, which is illustrated when considering the task of Automatic Speech Recognition (ASR).

Among Machine Learning tasks, ASR has multiple distinctive characteristics. At their core speech utterances are time series with an underlying semantic structure. They are not part of a finite-dimensional vector space, but rather of an infinite-dimensional Hilbert space. The outputs that ASR models predict are also elements of an infinite set of text transcriptions. Despite these particularities, ASR models are vulnerable to adversarial attacks, but with important differences compared to more straightforward tasks such as image classification. For instance, the transferability of adversarial examples between models plays a major role in the practical concerns they cause. However, this transferability is largely absent when it comes to targeted attacks on ASR models. Inversely, general-purpose defenses like adversarial training have yet to show success in ASR. Therefore adversarial examples, while affecting all fields of AI via very similar attack algorithms, also display properties that are specific to a *context*: nature of the task, set of outputs, etc. The current state of knowledge does not explain the mechanisms behind or extent of such context-specific aspects.

The objective of this thesis is to explore these aspects and help disentangle the seemingly contradictory nature of adversarial examples, to assess their threat against speech recognition models and the possibility for robust ASR. We carried out our work with two key principles in mind: the search for adversarially robust models should be conducted *in-depth* and *in-context*. By *in-depth* we mean that commonly accepted results or methodologies should not be taken for granted. While recent works tend to focus on defenses at scale on large datasets, we show that some phenomena can more easily be studied in simpler settings (chapter 3). We use different robustness metrics from simple accuracy under attack (chapter 4). We explore the ripple effects of adversarial attacks on other security aspects like privacy attacks (chapter 5).

By *in-context* we mean that aspects of adversarial attacks and defenses cannot solely rely on standardized benchmarks and general-purpose algorithms. *Speech recognition* robustness is inherently different from image classification robustness. In addition, different ASR architectures are very different from each other under adversarial attacks (chapter 5). Even proxy-based attacks, which have proven extremely challenging on past ASR models, are possible in some specific circumstances (chapter 6). On the other hand, defenses against ASR adversarial attacks benefit from leveraging domain-specific knowledge (chapter 7). Finally, training robust ASR models is easier under some learning paradigms than others (chapter 8).

1.1 Outline

A detailed outline of our contributions is presented below.

- **Chapter 2** establishes the formal setting for adversarial attacks and defenses and discusses relevant previous work.
- **Part I: Towards fine metrics of adversarial robustness** A fundamental limit to our knowledge of adversarial examples is the set of tools and metrics we use to evaluate robustness. In this first part, we illustrate some of these limits and propose different, finer metrics. To point out these limits we focus on the most studied field for adversarial robustness: image classification. The validity of those lessons for speech is the focus of later chapters.
  - In **chapter 3** we introduce the intriguing phenomenon of *accidental robustness*. We show that a certain amount of adversarial robustness can happen with random probability given a fixed training setting. This phenomenon is hard to notice using traditional metrics on traditional datasets. We use it to design a probabilistic approach to robustness-optimal hyperparameter search. Accidental robustness partially rehabilitates small datasets, which in adversarial robustness research are often considered unreliable. It also emphasizes some limits of adversarial accuracy in measuring robustness.
  - This leads us to propose different robustness metrics in **chapter 4**. Rather than measuring whether an input is vulnerable to one or more adversarial perturbations (adversarial

accuracy), we define proxy metrics of *how many* perturbations this input is vulnerable to on a hypersphere around the input. Such metrics include the angle between random directions and the closest adversarial perturbations in L_2 norm, and the minimal number of pixels to change from a random configuration in L_∞ norm. Armed with these tools we revisit multiple previous works, such as state-of-the-art defenses and others that are considered "broken". We show that sparsity can emphasize finer differences in robustness than adversarial accuracy. We also revisit accidental robustness and show that angular metrics help observe it on larger sets.

- **Part II: Evaluating threats against speech recognition** We have tools to explore adversarial robustness more finely, but still lack insight into the ASR-specific aspects of adversarial attacks, which we now investigate. We consider not only the nature of the task but the different threat models, as well as specific threats to major *applications* of ASR in practical settings.
  - First, we evaluate ASR under direct optimization attacks in **chapter 5**. We run a comprehensive study of how different architectures and training paradigms are vulnerable to white-box audio attacks. We notably show that the recent improvements in model accuracy have not been accompanied by an increase in robustness. We dedicate a deeper analysis to the recent Whisper model in particular and show a discrepancy between its impressive robustness to random noise and domain change and its unremarkable lack of adversarial robustness. We discuss how this widening gap between performance and safety increases the overall vulnerability of ASR systems. In addition, we show that white-box adversarial attacks can also have indirect impacts on ASR security, notably regarding privacy attacks. Applications of ASR like personal assistants are particularly vulnerable to this family of attacks due to their customer data acquisition pipeline.
  - In **chapter 6**, we switch to the more realistic threat model of proxy-based or *transferable* attacks. In contrast with negative results established in previous works, we show that such attacks are possible on many recent ASR models. Through an ablation study, we link this new vulnerability to the rising popularity of *self-supervised* ASR models. While such training paradigms encourage distributionally-robust representations, we show that this benefit paradoxically makes models less robust to transferable attacks. Consequently, the scenarios in which ASR models can be fooled tend to become more realistic over time.
- **Part III: Adversarially robust speech recognition** Elements discussed in the previous part create a gap between ASR and general knowledge of robustness, like our angular metrics or robust adversarial defenses. In this part, we discuss approaches to bridge that gap and propose frameworks for applying general robustness principles while taking into account the particular aspects of speech recognition and audio inputs.
  - **Chapter 7** shows how combining speech processing tools and general-purpose adversarial defenses can lead to robust models. The *randomized smoothing* defense for robustness, which adds Gaussian noise to inputs, is in theory task and model-agnostic, but in practice leads to an important drop in model performance. We demonstrate that tools proposed by the speech community help mitigate that drop. For instance, we propose

speech enhancement pre-processing and ROVER post-processing on ASR to improve randomized smoothing, resulting in **sequential smoothing**, the current state-of-the-art defense on the LibriSpeech dataset.

  – In **chapter 8** we evaluate the applicability of adversarial training to build robust ASR models. While adversarial ASR training is challenging with most fully-supervised architectures, we show that it is applicable during the pretraining of self-supervised, Transformer-based models. We propose a framework called **adversarial masked prediction** to pretrain robust models without drastic increases in computational cost. Using this method we train **AdvHuBERT**, a speech encoder with improved robustness when finetuned for ASR. We quantify that model's tradeoff in clean performance.

- Finally, **chapter 9** sums up our contributions and their implications, and suggests particularly interesting research directions for future work to build upon this thesis.

1.2 Supporting publications

This work is supported by several publications, articles accepted at future conferences, and articles currently under review. Some chapters overlap with several articles, and some articles overlap with several chapters. The detailed list of publications is provided below, in chronological order of submission.

1.2.1 Main publications

- Olivier and Raj [2021]: R. Olivier and B. Raj. Sequential randomized smoothing for adversarially robust speech recognition. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, Punta Cana, Dominican Republic, Nov. 2021. Association for Computational Linguistics (**Chapter 7**)

- Olivier and Raj [2022]: R. Olivier and B. Raj. Recent improvements of asr models in the face of adversarial attacks. *Interspeech*, 2022 (**Chapter 5**, **Chapter 6**)

- Shah et al. [2022]: M. Shah, R. Olivier, and B. Raj. Uncovering the robustness potential of neural architectures by measuring the probability of high adversarial accuracy. *Under review*, 2022 (**Chapter 3**)

- Olivier and Raj [2023a]: R. Olivier and B. Raj. How many perturbations break this model? evaluating robustness beyond adversarial accuracy. In *Proceedings of the 40th International Conference on Machine Learning*, Proceedings of Machine Learning Research, Honolulu, Hawaii, USA, 23–29 Jul 2023a (**Chapter 4**)

- Olivier and Raj [2023c]: R. Olivier and B. Raj. There is more than one kind of robustness: Fooling whisper with adversarial examples. *Interspeech*, 2023c. URL `https://arxiv.org/abs/2210.17316` (**Chapter 5**, **Chapter 7**)

- Olivier et al. [2023a]: R. Olivier, H. Abdullah, and B. Raj. The surprising vulnerability of self-supervised speech recognition models to transferable adversarial perturbations. *Under review*, 2023a (**Chapter 6**, **Chapter 7**)
- Olivier and Raj [2023b]: R. Olivier and B. Raj. Adversarial masked prediction for robust self-supervised speech models. *Under review*, 2023b (**Chapter 8**)

1.2.2 Other related publications

- Shah et al. [2021a]: M. A. Shah, R. Olivier, and B. Raj. Towards adversarial robustness via compact feature representations. In *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3845–3849, 2021a. doi: 10.1109/ICASSP39728.2021.9414696 (**Chapter 3**)
- Shah et al. [2021b]: M. A. Shah, R. Olivier, and B. Raj. Exploiting non-linear redundancy for neural model compression. In *2020 25th International Conference on Pattern Recognition (ICPR)*, pages 9928–9935, 2021b. doi: 10.1109/ICPR48806.2021.9413178 (**Chapter 3**)
- Olivier et al. [2021]: R. Olivier, B. Raj, and M. Shah. High-frequency adversarial defense for speech and audio. In *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2995–2999, 2021. doi: 10.1109/ICASSP39728.2021.9414525 (**Chapter 7**)
- Olivier et al. [2023b]: R. Olivier, F. Teixeira, K. Pizzi, A. Abad, I. Trancoso, and B. Raj. Exploring loss-based features for membership inference on asr. *Under review*, 2023b (**Chapter 5**)

Chapter 2

Background

2.1 Adversarial attacks

2.1.1 General considerations

While Szegedy et al. [2014] is famously a pioneering article on adversarial attacks against deep learning models, as Biggio and Roli [2018] point out the roots of the field are older by several years. There are multiple works focused for instance on ways to fool spam filters with minimal changes that can be computed with *adversarial machine learning* [Dalvi et al., 2004, Wittel and Wu, 2004, Lowd and Meek, 2005, Zhou et al., 2007]. Known as evasion attacks, these threats have given rise to the search for effective *defenses* against such attacks and *robust machine learning* models [Globerson and Roweis, 2006, Biggio et al., 2008, Kolcz and Teo, 2009].

However, Szegedy et al. [2014] shifted the focus of adversarial machine learning onto deep neural networks and showed that these models can be fooled with a single gradient step on the input that the human eye could not notice. This work gained considerable traction, and in the following years, several refinements on that original attack were proposed, up to the point where undefended models typically achieve 0% accuracy under attack. These attacks have also expanded the definition of what is considered an adversarial example.

Goodfellow et al. [2015] introduce the Fast Gradient-Sign Method (FGSM), in which the perturbation step follows the sign of the gradient pixel-wise. The DeepFool attack [Moosavi-Dezfooli et al., 2016] runs multiple gradient steps until the input is misclassified. Other attacks modify a minimal number of pixels instead [Papernot et al., 2016a]. The Projected Gradient Descent [Madry et al., 2018] and the Carlini&Wagner [Carlini and Wagner, 2016], the bases behind current adversarial robustness evaluation standards, also differ considerably. Moreover, while the aforementioned attacks affect numerical inputs in academic contexts, multiple works have instead focused on attacks that can operate "in the real world", or *practical attacks*. This can include patches in the shape of glasses that fool face recognition systems [Sharif et al., 2016] or stickers on stop signs to break traffic sign classifiers [Eykholt et al., 2018]. Other attacks have targeted different tasks such as Speech recognition [Carlini and Wagner, 2018]

In this rapidly expanding field, it would be extremely challenging to propose a formalism flexible enough to account for all attacks and practical enough to allow tractable evaluation. The remainder of this section introduces elements of formalism that are sufficient for our needs and will be consistently referred to in this thesis.

2.1.2 Threat models

The notion of adversarial threat model [Chakraborty et al., 2018] refers to the underlying assumptions in an adversarial attack. It covers multiple notions: the *objective* of the attack (*targeted* vs untargeted), its *constraints* (the set of *admissible perturbations*), and the *information* at its disposal (*white-box* vs *black-box* attacks).

We consider in the following a model f that performs a supervised task. Given an input-label pair (x, y), the objective of f is to minimize the error metric $\mathbf{e}(f(x), y)$. If the task is classification then $\mathbf{e}$ is the prediction error: $\mathbf{e}(y, y')$ equals to 1 if $y \neq y$ and 0 otherwise. If the task is Speech Recognition it could be the Word-Error Rate (see Section 2.3).

f is trained over a supervised task by minimizing a loss $L(f, x, y)$ over its training set. (x, y) are an arbitrary input-label pair in the test set. Adversarial attacks take f, x and (optionally) y as input and return a perturbation δ.

Targeted vs untargeted A targeted attack tries to force a model to predict a specific output called *target*, i.e. to minimize $\mathbf{e}(f(x+\delta), y_T)$ with $y_T \neq y$ the target. On the other hand, an untargeted attack tries to force misprediction, i.e. minimize $\mathbf{e}(f(x+\delta), y)$. In practical contexts, this objective can be called *denial of service*.

Targeted and untargeted objectives may differ more or less crucially. If the task is binary classification they are equivalent. With Speech Recognition they are radically different, and targeted adversarial examples are much more difficult to compute than untargeted ones.

If the true label y is not available to an untargeted attack, then it can be replaced with the predicted label $f(x)$. For high-accuracy models, this has a limited but non-negligible effect on the outputs of attacks.

Admissible perturbations The goal of adversarial attacks is to return *minor* perturbations, that would be considered insignificant by a human agent. The exact definition of minor perturbations may differ from one context to another: some attacks try to return imperceptible perturbations, but this cannot be said of attacks that put glasses on faces [Sharif et al., 2016]. However many attacks define a fixed set of *admissible perturbations* Δ and their goal is to return a perturbation within that set:

$$\arg\max_{\delta \in \Delta} \mathbf{e}(f(x+\delta), y) \quad \text{or} \quad \arg\min_{\delta \in \Delta} \mathbf{e}(f(x+\delta), y_T) \tag{2.1}$$

for untargeted and targeted attacks respectively.

Commonly Δ would be a ball of radius ϵ in a norm L_p, with typically $p = \infty$ or $p = 2$. ϵ is then called the *noise budget*. Attacks like JSMA [Papernot et al., 2016a] use $p = 0$. Other attacks have tried to propose different sets Δ more aligned with human perception, that take into account the input x, such as Wasserstein distance [Wong et al., 2019]. However standard L_p norms constitute good benchmarks for adversarial robustness. In this thesis, we most often consider L_2 and L_∞-bounded attacks,

Equation 2.1 can be seen as the objective achieved by a perfect attack or *worst-case adversary*. Adversarial attack algorithms are surrogates for this worst-case attack.

In the case of classification tasks, $\mathbf{e}$ can only take two values. This allows us to identify in Δ, for a given input x, the subset of *successful* adversarial perturbations. On the other hand, some attacks such as Carlini&Wagner are *unbounded*. In that case, $\Delta = \mathbb{R}^n$ but the attack aims at returning a perturbation as small as possible. The notion of admissible set then loses most of its meaning. We can alternatively consider the attack to be always successful, but evaluate it with the amount of noise it uses: the smaller the perturbation, the better the attack is.

White-box vs black-box White-box and black-box attacks differ in the information they have access to. White-box attacks assume complete knowledge of the model and specifically can access its gradients with backpropagation. These gradients can be used to compute directions of greatest change on the inputs and optimize the adversarial perturbation δ. FGSM, PGD, and Carlini&Wagner are all white-box attacks. This setting also assumes knowledge of any defense mechanism the model uses: this is further detailed in section 2.2.1.

Black-box attacks only assume access to the model as a decision rule: they can feed it inputs and access outputs. These attacks use black-box optimization mechanisms, based on random search and rejection sampling with varying heuristics and amounts of supervision [Narodytska and Kasiviswanathan, 2017, Brendel et al., 2018, Su et al., 2019, Croce and Hein, 2020a, Andriushchenko et al., 2020]. Although white-box attacks are in principle strictly stronger than black-box attacks, they can be affected by *gradient obfuscation* effects that black-box attacks are immune to [Athalye et al., 2018a]. Therefore black-box attacks can be useful when evaluating adversarial robustness. This is further discussed in section 2.2.1.

Multiple intermediate scenarios can be considered between white-box and black-box settings and can be called *grey-box* threat models. A common situation is that the model weights are kept secret, but all of its training mechanisms and hyperparameters, randomness excluded, are made public. This scenario is especially relevant because of the *transferability* effect of adversarial examples, identified by Papernot et al. [2016b]. This article showed that perturbations computed on a given example and model often break *different* models on the same example. Transferability becomes more effective when models are trained similarly. This effect enables a family of attacks in which the adversary trains a *surrogate* model - using grey-box information - then runs white-box attacks on that model and transfers the obtained perturbations onto the original model. Transferred attacks are typically weaker than black-box attacks, but are easier to compute - provided that the surrogate model has already been trained.

We now provide more details about two standard white-box attacks that we will repeatedly use in the following chapters.

2.1.3 Projected Gradient Descent

The Projected Gradient Descent (PGD) attack was proposed by Madry et al. [2018] and is a generalization of the FGSM attack [Goodfellow et al., 2015]. Its authors argue that it is the strongest first-order adversarial attack - i.e. attack that only uses direct gradient information from the model loss. It is also a versatile attack, that can be targeted or untargeted, and declined in any norm L_p Given an input x and its label y, this attack optimizes the following objective (which is trivially adapted for targeted attacks):

$$\arg\max_{\delta\in\Delta} L(f(x+\delta), y) \tag{2.2}$$

using projected gradient descent for a number n of gradient update steps. At each step $k+1$ the updates are:

$$\delta'_k \leftarrow \delta_k + \eta * \text{sign}(\nabla_{\delta_k} L(f(x+\delta_k), y)) \quad \text{(gradient ascent)} \tag{2.3}$$

$$\delta_{k+1} \leftarrow \mathcal{P}_{\Delta}(\delta'_k) \text{ (projection step)} \tag{2.4}$$

where η is the step size, and $\mathcal{P}$ is the projection operator onto Δ. In the L_∞-PGD attack this projection step would be:

$$\delta_{k+1} \leftarrow \text{clip}_\epsilon(\delta'_k) \text{ (element-wise clipping)} \tag{2.5}$$

While for L_2-PGD it is:

$$\delta_{k+1} \leftarrow \min(1, \frac{\epsilon}{\|\delta'_k\|_2}) * \delta'_k \tag{2.6}$$

L_∞-PGD with $n = 0$ and $\eta = \epsilon$ is equivalent to FGSM. PGD attacks are typically evaluated with the score $\mathbf{e}(f(x + \delta), y)$ over the evaluation set - which for classification tasks is nothing but the model error rate, i.e. the attack success rate.

δ_0 can be initiated at 0 or a random admissible point - which improves the success rate. Other optimization tricks can include momentum [Dong et al., 2018b] or multiple restarts [Uesato et al., 2018b]. Croce and Hein [2020b] proposed AutoPGD, a parameter-free PGD attack in which the step size is automatically updated. The PGD attack has been extensively studied in the literature and is a staple of robustness evaluation. Combined with some other strong attacks, AutoPGD forms AutoAttack, which is used in the RobustBench benchmark [Croce et al., 2021].

2.1.4 Carlini&Wagner

The Carlini&Wagner (CW) attack [Carlini and Wagner, 2016] optimizes the following objective:

$$\arg\max_{\delta \in \mathbb{R}^n} L(f(x + \delta), y) + \lambda \|\delta\|_2^2 \tag{2.7}$$

This L_2 attack is unbounded and only enforces small perturbations through regularization term $\lambda \|\delta\|_2^2$. Its intended behavior is to find the smallest successful perturbation, by optimizing the objective in 2.7 (typically with the Adam optimizer), and using binary search on the regularization coefficient λ. There are versions of this attack for other norms, but it is more effective in the L_2 setting.

The CW attack often requires more steps than the PGD attack to obtain successful examples. It is however resilient to difficult settings: it has broken numerous weak defenses (section 2.2.1) and can be adapted for targeted attacks on Speech Recognition, where PGD optimization fails (section 2.4).

2.2 Adversarial defenses

Research on adversarial robustness at its beginnings had taken the form of an arms race. In reaction to the first attacks, a great amount of work has focused on mitigation mechanisms against adversarial perturbations, referred to as *adversarial defenses*. These methods were often based on "common sense" intuitions and heuristics. Defensive distillation [Papernot et al., 2015] for instance showed great success against early, simple attacks, but was broken by stronger algorithms [Carlini and Wagner, 2016]. Some works focused on detecting adversarial examples as outliers rather than predicting them correctly, using for example trained classifiers [Metzen et al., 2017] or statistical methods [Feinman et al., 2017]. These methods were ultimately defeated as well [Carlini and Wagner, 2017].

2.2.1 Obfuscation and adaptive evaluation

A few years of back and forth have eventually underlined some structural flaws in most heuristic defenses. After the standardization of the PGD and CW attacks as strong robustness benchmarks, many defense works [Guo et al., 2018, Buckman et al., 2018, Song et al., 2018, Xu et al., 2018, Samangouei et al., 2018, He et al., 2017] were broken at the same time and with limited effort by single papers [Carlini and Wagner, 2017, Athalye et al., 2018a, Uesato et al., 2018a]. Athalye et al. [2018a] in particular have emphasized the phenomenon of *gradient obfuscation*, in which the addition of the "defense" makes gradients either harder to backpropagate (with non-differentiable components) or less informative (with randomness or vanishing gradient effects). Obfuscation leads to artificially increased performance on adversarial performance by confusing the attacker, often without doing much to predict correctly the adversarial examples themselves. In other words, these defenses rely on the gap between actual (weak) attacks and the ideal worst-case adversary but are ineffective against that worst-case attack.

Multiple works have since then proposed guidelines to properly evaluate adversarial defenses [Athalye et al., 2018a, Carlini et al., 2019, Tramer et al., 2020]. Such guidelines involve using black-box attacks or transferred examples, which are typically immune to obfuscation effects. But the best evaluation method is to design adaptive attacks against every defense, to backpropagate gradients correctly. For instance, any defense involving random transformations of the inputs should use *Expectation over Transformation* [Athalye et al., 2018b] i.e. optimize the expectation of the transformation, approximated with samples. Designing adaptive attacks against one's own work is very difficult, however, and many recent works published at top conferences keep running misleading evaluations [Tramer et al., 2020].

Another way to circumvent obfuscation problems is to design defenses whose robustness is certified by mathematical guarantees. This has been an active field in recent years, with several families of robust defenses. Some rely on model *verification*, i.e. proof that the function a neural network computes is stable in the neighborhood of a specific input, using methods such as Satisfiability Modulo Theories solvers [Ehlers, 2017], Mixed Integer Programming [Tjeng et al., 2019] or Abstract Interpretation [Gehr et al., 2018]. These methods tend to suffer from very high complexity when applied to large networks. Others optimize a relaxed, loose bound on the adversarial objective, that enables to run convex optimization algorithms [Kolter and Wong, 2017, Gowal et al., 2018, Salman et al., 2019, Mirman et al., 2021].

2.2.2 Randomized smoothing

A final family of methods generates probabilistic guarantees instead. Noise addition certified defenses were introduced in Pinot et al. [2019] and Lécuyer et al. [2019], respectively based on the frameworks of probabilistic mapping and differential privacy. Later Cohen et al. [2019] improved on the formalism and robustness guarantees with Randomized Smoothing, showing in fact that their L_2 bound is tight. The principle of randomized smoothing is to replace a deterministic classifier $f : \mathbb{R}^d \to \{1, 2, ..., m\}$ with the smooth classifier:

$$g(x) = \arg \max_{k \in \{1,2,...,m\}} \mathbb{P}[f(x + \epsilon) = k] \tag{2.8}$$

with $\epsilon \sim \mathcal{N}(0, \sigma^2 I)$. More precisely, since classifier g cannot be evaluated exactly, it is estimated with a form of Monte Carlo algorithm: many noisy forward passes are run and a majority vote determines the output label.

The underlying reasoning behind this method is that given a small perturbation δ, and a standard deviation $\sigma >> ||\delta||_2$, probability distributions for $x + \epsilon$ and $x + \delta + \epsilon$ are very "close" by standard divergence metrics. Therefore discrete estimators built around these distributions have equal value with high probability. So if an attacker crafts an adversarial perturbation δ, it will have a very small chance of changing the output of g.

When using randomized smoothing, the biggest challenge is to retain good performance on very noisy data. One can see this defense as a way to shift the problem from adversarial robustness to white noise robustness. This is typically done with data augmentation during training. Salman et al. [2020b] propose to use a denoiser network instead, which makes smoothing applicable off-the-shelf on pretrained networks. Olivier and Raj [2021] use Speech Enhancement for the same purpose, as well as custom voting strategies, to use smoothing on Speech Recognition.

2.2.3 Adversarial training

While provable defenses come with many advantages, they suffer from overly conservative bounds or scaling issues. As a consequence, state-of-the-art results for most threat models are held by *adversarial training* defenses, a rare example of a heuristic *and* obfuscation-free approach [Athalye et al., 2018a] that comes with no formal guarantee but very strong empirical performance.

Essentially, these defenses train the model to be robust to attacks by encountering adversarial examples during training. This approach is very intuitive and almost as old as adversarial examples themselves [Goodfellow et al., 2015]. In its earliest version, FGSM attacks were computed on training inputs and fed to the model instead of the original data points. Borrowing the formalism in Madry et al. [2018], given the adversarial optimization objective on data distribution $\mathcal{D}$:

$$\min_{(x,y) \sim \mathcal{D}} \mathbb{E}[\arg \max_{\delta \in \Delta} L(f(x + \delta), y)] \tag{2.9}$$

adversarial training optimizes a lower bound of this objective, by replacing the worst-case loss $\arg \max_{\delta \in \Delta} L(f(x + \delta), y)$ with an adversarial example. Since it is a lower bound rather than an upper bound, no guarantee in this min-max problem can be achieved.

Despite its simple approach, turning adversarial training into an effective defense is difficult. Its earliest version was not robust to strong attacks. It becomes effective if a strong attack algorithm is used: the first strong adversarial training defense uses the PGD attack [Madry et al., 2018] and is still a solid defense baseline to this day. In fact, it has for several years mostly seen minor improvements compared to its original version: improvements to the PGD attack like Dong et al. [2018b] lead to slightly improved performance, and TRADES [Zhang et al., 2019] trades off standard and adversarial objectives, which led to a long-held state of the art for L_∞ robustness. However Rice et al. [2020] attribute this improvement to a form of overfitting, and an improved parameter-free attack [Croce and Hein, 2020b] brought TRADES performance down to the PGD training.

Several improvements to adversarial training have been proposed in recent years. It has been shown to benefit from larger datasets and data augmentation [Gowal et al., 2020, Rebuffi et al., 2021], and mixes well with self-supervised learning [Schmidt et al., 2018, Alayrac et al., 2019, Carmon et al., 2019, Zhai et al., 2019, Kim et al., 2020]. It also requires larger model capacity than standard training to fully exploit its potential [Xie and Yuille, 2020]. Another line of work has improved the computation time required by PGD attacks, managing to achieve comparable performance with fewer iterations or even FGSM training, thanks to random initializations or optimization tricks [Wang and Zhang, 2019, Wong et al., 2020].

Adversarial training has interesting uses on top of being an effective defense. Xie et al. [2020] show that it can improve performance on image recognition in some cases. It also improves domain generalization [Li et al., 2018].

2.3 Speech recognition

By Automatic Speech Recognition (ASR) we refer to the task of associating a speech utterance with its natural language transcription. This is a complex, temporal classification task with continuous inputs and a semantic structure in both inputs and outputs. A wide variety of models have been used over the years, involving for instance Hidden Markov Models [Rabiner, 1989] or hybrid Neural/HMM models [Levin, 1990, Iwamida et al., 1991, Johansen, 1996, Wang and Sim, 2011]. In the past decade, end-to-end neural models for speech recognition have seen important breakthroughs [Graves et al., 2013, Amodei et al., 2016, Chan et al., 2016, Dong et al., 2018a, He et al., 2019, Baevski et al., 2020, 2021, Hsu et al., 2021, Radford et al., 2022], and are the focus of this thesis.

ASR is evaluated by comparing the differences between the predicted and the target transcription, typically using Word-Error-Rate (WER) or Character-Error Rate (CER), which are edit distances in the word and character spaces.

2.3.1 Neural ASR architectures and training methods

Most neural ASR models rely on Long Short-Term Memory (LSTM) networks [Hochreiter and Schmidhuber, 1997], or more recently on Transformers [Vaswani et al., 2017], or Transformer-inspired architectures. An example is the Conformer network [Gulati et al., 2020], which interleaves self-attention layers with convolutional layers. Both architectures are still used in different settings: scaled Transformers usually reach higher performance, but LSTMs may sometimes be beneficial for streaming applications.

ASR is a sequence-to-sequence task, and as such a common method is to train Attention Encoder-Decoders (AED) [Chan et al., 2016]. For those models, the major difference with NLP tasks like Machine Translation is that the encoder takes continuous speech inputs rather than discrete units. Therefore, the tokenizer is replaced with an audio front-end downsampling the audio to a shorter sequence of local features. That front-end may compute Spectrogram-based encodings like Mel features or MFCC, or it may be a small CNN directly operating on raw audio.

Contrary to most NLP sequence-to-sequence tasks however, ASR is monotonic: given an (audio) input sequence $x_{[1..n]}$ and a (text) output sequence $y_{[1..m]}$, if y_i is aligned with x_j then y_{i+1} can only be aligned with x_k where $k > j$. Attention does not enforce such a constraint, but other training objectives that do have proven competitive. One such example is the RNN-Transducer, first proposed in Graves et al. [2013]. It consists of an LSTM encoder network that takes the current audio sample, an LSTM decoder network that takes the previous RNN-T output, a joint network that combines the outputs of the encoder and decoder, and a final softmax layer that predicts the current output token. The RNN-T is an efficient architecture with low latency, suited for streaming ASR on mobile devices [He et al., 2019].

It is even possible to remove the decoder altogether and train encoder-only models for ASR. This is achieved with Connectionist Temporal Classification (CTC) loss [Graves et al., 2006], which uses a Viterbi-like algorithm to average the loss over all possible speech-to-text alignments. Without a decoder, CTC models enforce strong independence assumptions on their output tokens, which can limit their performance in some settings. Those limitations can be addressed by combining them with a language model [Amodei et al., 2016, Higuchi et al., 2022]. CTC can also be advantageously combined with AEDs as an auxiliary objective for the encoder [Hori et al., 2017].

For each of those objectives, LSTMs and Transformers are largely interchangeable.

2.3.2 Self-supervised pretraining for ASR

The ASR training objectives above all require speech data labeled with transcription. This requirement limits the amount of data available: standard datasets like LibriSpeech contain up to 1khrs of labeled speech. To overcome this issue, a vast area of research has studied the use of unlabeled speech data to train ASR. Some methods rely on pseudo-labeling: models trained with supervised ASR objectives are used to label more data and continue training [Synnaeve et al., 2020, Likhomanenko et al., 2020, Lugosch et al., 2022]. Another popular approach is to use Self-Supervised Learning (SSL) to pretrain encoders.

SSL consists of training models to solve a "pseudo-task", with labels extracted from the audio itself [Mohamed et al., 2022]. Major families of SSL objectives are:

- *generative training*: learn to generate audio with autoregressive or denoising objectives [van den Oord et al., 2017]
- *contrastive learning*: learn to match outputs with the right inputs between a batch of distractors [van den Oord et al., 2018, Baevski et al., 2020, 2021]
- *masked prediction*: mask part of the inputs, and learn to predict a discrete or continuous label extracted from those inputs [Hsu et al., 2021, Baevski et al., 2022b]

The vast majority of SSL models introduced in the past three years rely on Transformers. After pretraining, those models can be further adapted to ASR or other tasks with small amounts of labeled data. Training methods include fine-tuning with CTC training, freezing the SSL encoder as an embedding to train an AED on, or using adapters or prompting. On most ASR benchmarks, the state-of-the-art approaches exploit SSL pretraining on a very large unlabeled corpus, followed by CTC finetuning Baevski et al. [2021], Hsu et al. [2021], Baevski et al. [2022a], Chen et al. [2021].

2.3.3 Our models

We detail the specific models we use throughout this work:

- **DeepSpeech2** [Amodei et al., 2016] (chapters 5,7) is a CTC-trained LSTM model trained on the LibriSpeech dataset.
- We use multiple AEDs:
  - the **CRDNN** model from the **SpeechBrain** library [Ravanelli et al., 2021] using LSTMs (chapters 5 and 6)
  - the **SpeechBrain Transformer** model (chapters 5, 6 and 7)
  - The **Fairseq Speech2Text** Transformer [Wang et al., 2020] (chapter 6)
  - The **Whisper** models [Radford et al., 2022], a family of English and multilingual AEDs trained not on LibriSpeech but on a large, unpublished labeled dataset of 680khrs.
- SSL models are the focus of chapter 6. We directly evaluate several models:
  - **Wav2Vec2** [Baevski et al., 2021], pretrained with contrastive learning on masked audio frames
  - **HuBERT** [Hsu et al., 2021], **WavLM** [Chen et al., 2021] and **UniSpeech** [Wang et al., 2021, Chen et al., 2022], trained on masked prediction of discrete labels extracted from the audio with clustering.
  - **Data2Vec** [Baevski et al., 2022b], where a student model is trained on masked prediction of continuous representations generated by a teacher. The teacher is an Exponential Moving Average of the student.

  All those SSL models are Transformers fine-tuned with CTC. We also use SSL models in chapters 7 (Wav2Vec2 and HuBERT) and 8 (HuBERT).

2.4 Adversarial examples on speech recognition

2.4.1 White-box attacks

Untargeted attacks on ASR are relatively straightforward. When applying the PGD attack to any ASR model with a differentiable loss we can easily induce denial-of-service with very small perturbations. With more effort, an attacker can compute such perturbations that transfer across models and inputs [Neekhara et al., 2019] or do not require white-box access [Abdullah et al., 2021a].

Targeted attacks are much more complex, and require specific algorithms. The earliest attacks proposed on ASR were the ultrasonic-based DolphinAttack [Zhang et al., 2017] and the Houdini loss for structured models [Cisse et al., 2017]. In Carlini and Wagner [2018] the authors propose an extension of their CW attack for audio inputs, computed against the DeepSpeech2 model. This attack replaces the loss function in Equation 2.7 by the CTC loss. It also proposes some tricks to improve the SNR, such as computing an initial adversarial example with the vanilla CW attack, then using its alignment with the target sentence to refine the perturbation with a per-character loss.

However, available implementations of that attack do not include this second step and we focus on the vanilla CW attack.

Other works have extended the state-of-the-art with over-the-air attacks [Yuan et al., 2018, Yakura and Sakuma, 2019, Li et al., 2019]. A recent line of work has improved the imperceptibility of adversarial noise by using psychoacoustic models to constrain the noise rather than standard L_2 or L_∞ bounds [Szurley and Kolter, 2019, Schönherr et al., 2019, Qin et al., 2019]. Such attacks reduce the perception of the noise at an equal budget but do not reduce the SNR.

2.4.2 Black-box and transferable attacks

Black-box attacks on ASR remain challenging. Some methods rely on black-box optimization but only have shown good results at a small scale [Alzantot et al., 2018, Taori et al., 2019]. Others do not rely on optimization at all, like Abdullah et al. [2021a] and Abdullah et al. [2023] which apply perturbations by focusing on inaudible frequencies. Those methods have the advantage of being automatically transferable, since they are model-agnostic. However, they are always untargeted.

The transferability of targeted, optimization-based ASR attacks is a highly challenging problem. Some works like Yuan et al. [2018] claim a certain amount of transferability for targeted attacks; however they embed their adversarial noise in songs, allowing for much more important amounts of noise while being technically stealthy. In fact, some negative results have been established. Abdullah et al. [2021b] shows that on DeepSpeech2-type models, targeted attacks show no transferability even when models are trained using the exact same hyperparameters apart from the random seed. In Abdullah et al. [2022] the same authors analyze more in detail the factors that limit transferability and show that recurrence and spectrogram-based features are important obstacles. However, even removing those obstacles did not enable them to transfer attacks between large-scale ASR models. Our work Olivier et al. [2023a] (chapter 6) is the first occurrence of targeted transferability between strong ASR models.

2.4.3 Defending ASR against adversarial perturbations

Only a few works have proposed defenses against adversarial attacks specifically for speech recognition models. Some, like Yang et al. [2019], have not stood the test of adaptive attacks [Tramer et al., 2020]. Others focused on methods that have a track record of robustness in other modalities, like randomized smoothing [Żelasko et al., 2021]. We improve on this work in Olivier and Raj [2021] (chapter 7) and investigate adversarial training in Olivier and Raj [2023b] (chapter 8)

Part I

Towards fine metrics of adversarial robustness

Chapter 3

Variance in adversarial robustness

Ultimately, one of the most desirable objectives in research on adversarial robustness is to create defense mechanisms that are versatile, robust, and scalable. In that regard, a sensible approach is to set strong attacks as benchmarks and evaluate defended models with their results under attack on large and complex datasets. However, when it comes to gathering knowledge and insight on adversarial examples, this approach has important flaws. A metric like adversarial accuracy only reflects a limited perspective on the model's behavior in the vicinity of its inputs; focusing on the most challenging datasets makes the analysis of those phenomena even more difficult. Our objective in this part is to illustrate and alleviate some of those limits.

In this chapter, we introduce an interesting aspect of adversarial robustness. We observe that in several contexts partial robustness to adversarial attacks can occur during training without any explicit effort to make models robust. This "spontaneous" robustness often happens with very high variance: a learning algorithm that does not seem to particularly favor robustness may occasionally output a partially robust model, with a non-zero probability. We call this phenomenon *accidental robustness*. While we show that these are general phenomena that suggest profound changes in our approach to robustness, accidental robustness is much more easily observable on simple or low-dimensional datasets. Yet its observation brings insights that can help design robust architectures at larger scales. As such, it perfectly illustrates how small datasets offer interesting perspectives for adversarial research, in addition to more complex ones. The fact that adversarial accuracy can vary so much in identical training settings also illustrates the limits of that metric as a measure of robustness - a topic that will be followed up in the next chapter.

3.1 Introduction

Fixing a neural architecture and random initial conditions (weight initialization, data shuffling, etc), standard gradient descent-based optimization will usually yield a high-accuracy model on the training dataset, and with proper generalization on the test dataset as well. It may also, with a certain probability, yield a model robust to adversarial attacks on that test set. In other words, certain achievable minima of the standard training objectives may correspond to minima of the adversarial training objectives. Those are the models we dub *accidentally robust*. This statement might seem banal when devoid of context; however, we experimentally show the following, surprising results:

- In several settings, the variance in accidental robustness can be very high

- Choices of hyperparameters affect the probability of accidental robustness
- Hyperparameters that increase the odds of accidental robustness tend to be the same across multiple datasets
- Hyperparameters that increase the odds of accidental robustness tend to align with the ones improving the results of adversarial training according to previous works [Madry et al., 2018, Rebuffi et al., 2021]

One interpretation of those results This is that accidental robustness is a method to find "robustness-friendly" hyperparameters. We argue that neural architecture choices give some robustness *potential* to classification models: they do not necessarily make them robust to adversarial attacks everything else equal, but they increase the capacity of the model to show robustness in favorable conditions. Training models adversarially is simply one way to unveil this robustness potential.

This chapter demonstrates the above results empirically, and to an extent theoretically. It is organized as follows. In section 3.2, we describe RXOR, a synthetic toy problem that most of our experiments use. We mathematically establish that choices in the number of neurons, layers, and activation functions condition the existence of a robust and accurate solution to the optimization problem. In section 3.3 we propose a methodology for measuring accidental robustness. In section 3.4 we describe our experimental setup. In section 3.5 we evaluate accidental robustness on RXOR when varying several hyperparameters, and show that hyperparameter choices condition the probability of accidental robustness. In section 3.5.4 we run similar experiments on the MNIST dataset and with adversarial training and show extremely similar results between these different settings. Finally in section 3.6 we show that activation pruning is an example of a post-training method that can also lead to increased robustness potential.

3.2 RXOR Problem Description

To illustrate the phenomenon of accidental robustness we consider deep neural networks trained to approximate a real-valued variant of the XOR function. The Boolean XOR function (BXOR) is a binary operator (usually denoted $x_1 \oplus x_2$) that takes Boolean inputs and outputs True if and only if one of the inputs is True and the other is False. The function can also be extended to an arbitrary number of inputs by using the base case BXOR$(x_1) = x_1$ and the following recursive rule for $n \geq 2$:

$$\text{BXOR}(x_1, .., x_n) = \text{BXOR}(x_1, .., x_i) \oplus \text{BXOR}(x_{i+1}, .., x_n)$$

Unlike the case of, for example, functions defined over natural images, the BXOR domain spans the entire input space. Therefore while on image classification there is for each input a region of input space in which the pixel values may change semantics of the input remain unchanged; in the case of the Boolean XOR, any perturbation changes the input in a semantically meaningful way. Due to this property, the Boolean XOR function does not permit adversarial perturbations as defined in this work. To get around this limitation, in this paper, we use a real-valued variant of the XOR function (RXOR), which is defined as RXOR$(x_1) = \text{sign}(x_1)$ and for $n \geq 2$:

$$\text{RXOR}(x_1, .., x_n) = \text{RXOR}(x_1, .., x_i) \neq \text{RXOR}(x_{i+1}, .., x_n)$$

where $x_i \in [-1, 1]$. It is easy to see that this formulation of the XOR function permits perturbations that do not change the input in semantically meaningful ways. For instance, if $x_{j \neq i}$ are fixed, changing x_i from 1 to $\epsilon > 0$ does not change the value of RXOR$(x_1, .., x_i, ..., x_n)$.

The domain of RXOR contains infinitely many elements, therefore it is impossible to train a model on the entire input domain. Moreover, since the training set of the model is a finite sample from an infinite set, a non-robust solution is necessarily permitted. This is because any finite sample from $[-1, 1]^n$ will necessarily produce a region around the decision boundary (the standard bases of $\mathbb{R}^n$) from which no points are sampled; due to which multiple solutions are made possible of which only one is the true one. This is shown in Figure 3.1 where the yellow region around the axes represents the unsampled region. For example, if in the training data $|x_i| \geq \epsilon > 0$ then a valid solution, given this training data, might consider $0 < x_i < \epsilon$ to have sign -1, that is the decision boundary is drawn at the edge of the yellow region. If this happens then an adversary might be able to perturb x_i just enough that $0 < x_i < \epsilon$ and cause a misclassification. One could indeed argue that the input with the perturbed x_i is not part of the training data distribution and is an example of an *off-distribution adversarial input*.

Based on the preceding discussion, we can define an adversarially robust model as a model that can make highly accurate predictions even when test data is not in the support of the training data distribution. Figure 3.1 illustrates an example of training and testing data distributions which differ as mentioned above. The yellow region is in the support of the test data but not of the training data. As a result the set of valid solutions i.e. those that achieve high accuracy on the training data but not necessarily on the test data, is a superset of the set of robust solutions, i.e. those that achieve high accuracy on the test data. If the prediction accuracy on the training data is the sole metric, the optimization procedure will not be able to identify the robust solution from amongst the valid solutions and hence will converge to the solution that is most accessible from the point of initialization. It follows that the probability that a trained model will be adversarially robust is equal to the probability that a randomly initialized model is close to a robust solution. In the subsequent sections, we first describe a Monte Carlo sampling-based method for estimating the probability that a randomly initialized model will converge to a robust solution, then we present and analyze the estimates obtained by applying this method to various models trained to compute RXOR. In our analysis, we aim to isolate trends that can be used to inform modeling choices that would facilitate the training of robust models.

3.2.1 Influence of hyperparameters on accurate and robust classifiers

Given certain choices of hyperparameters and attack radius ϵ, there may or may not be accurate and robust classifiers. For instance, in Figure 3.1 the orthogonal cross classifier is accurate and robust for some values of ϵ. We formalize that intuition into theoretical results, which we express below.

Formally, given a number m we define $F_{a_1, a_2, ..., a_m}$ as the set of n-hidden-layer binary neural classifiers, that have a_1 neurons on the first hidden layer, a_2 on the second, etc. The final layer always has one neuron for binary classification. When there is no ambiguity on f, we name $h_{i,j}$ the j^{th} neuron of the i^{th} layer. We name $h'_{i,j}$ the pre-activation neuron (which is an affine function). We can establish the existence or non-existence of valid solutions on $\mathcal{X}_\epsilon$ for certain values of ϵ, n, m, a_i, and certain activation functions.

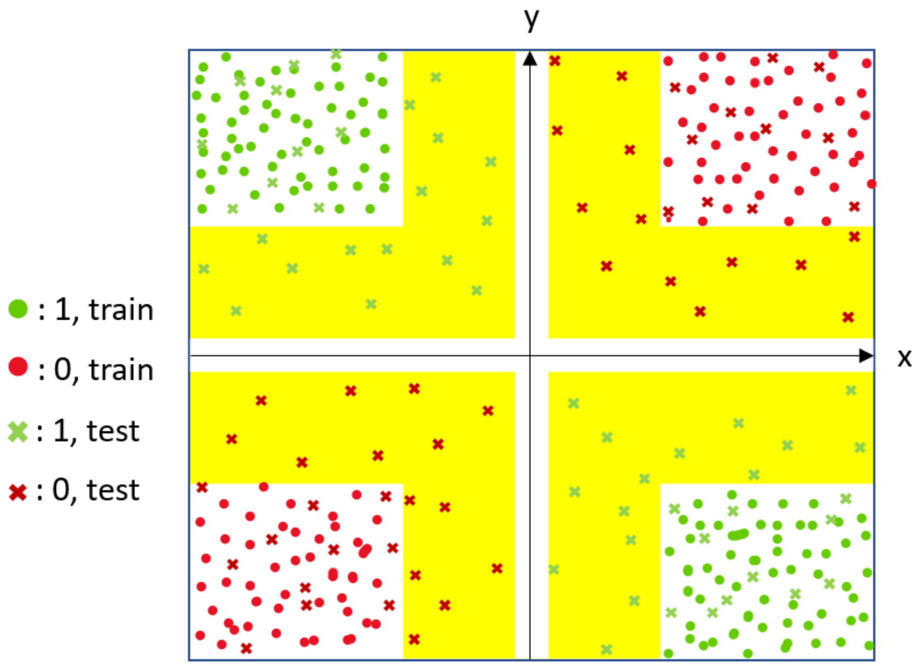


Figure 3.1: 2D RXOR dataset illustration. The yellow section is the adversarial region, i.e. in our framework set difference between the test data support and the training data support.

Proposition 3.1. *For* $n = 2$*, with threshold activations,* $F_{2,2}$ *contains a classifier that is valid on* $\mathcal{X}_\epsilon$ *for all* $0 < \epsilon < 1$.

Proposition 3.2. *For* $n = 2$*, with threshold activations, given* $0 < \epsilon_1 < \frac{2}{3}$ *and* $\frac{2}{3} < \epsilon_2 < 1$*,* F_2 *contains a valid classifier on* $\mathcal{X}_{\epsilon_1}$ *but not on* $\mathcal{X}_{\epsilon_2}$

Proposition 3.3. *For* $n = 2$*, with ReLU activations,* F_3 *contains a classifier that is valid on* $\mathcal{X}_\epsilon$ *for all* $0 < \epsilon < 1$.

We prove all of the above results in Appendix A.1. For the remainder of this chapter, we focus on empirically establishing which choices of parameters tend to favor robustness, using a methodology based on accidental robustness.

3.3 Methodology

To estimate the probability of arriving at a robust solution we use a Monte Carlo sampling-based approach. We begin by selecting an architecture for our neural network and initializing the parameters of the network by uniformly sampling a point from a p dimensional L_2 norm ball, where p is the number of parameters of the network. The radius of the norm ball is chosen such that valid and robust solutions are possible within it. Next, we use Stochastic Gradient Descent (SGD) to optimize the parameters to minimize the prediction error on a training dataset. In this step, we perform only a small number of SGD updates because we only want to search for a solution in the vicinity of the random initialization. This is to ensure that we are visiting a wide range of solutions, in the case that SGD tends to end on a specific minimum given enough time. The distribution of the training data resembles Figure 3.1 in that points are sampled uniformly from $([-1, -\epsilon] \cup [\epsilon, 1])^n$, where n is the number of inputs to RXOR and ϵ is the width of the unsampled region around the axes. Concretely, we sample data sets $\mathcal{X}_{\epsilon_1} ... \mathcal{X}_{\epsilon_k}$ for $\epsilon_0 < ... < \epsilon_k$, such that

$$\mathcal{X}_{\epsilon_j} = \{\mathbf{s} \odot \mathbf{x} | \mathbf{x} \in \mathbb{R}^n, x_i \sim U[\epsilon_j, 1], \mathbf{s} \in \{-1, 1\}^n, P(s_i = -1) = P(s_i = +1) = 0.5\} \tag{3.1}$$

where $\odot$ represents element-wise multiplication. We then train the model on one of the data sets, say $\mathcal{X}_{\epsilon_j}$, and evaluate its accuracy on all of them. Note that the support of the test sets $\mathcal{X}_{\epsilon_{i<j}}$ includes a region that is not in the support of the training set. Based on the definition of robustness presented in the previous section, if a model trained on the training set is able to make perfect predictions on the testing set, we will consider it to be robust. More specifically, we will refer to a model that achieves 100% accuracy on $\mathcal{X}_{\epsilon_i}$ as being ϵ_i-robust. Of course, all models that are trained on $\mathcal{X}_{\epsilon_i}$ will be ϵ-robust for $\epsilon \geq \epsilon_i$ but only some of them might be robust for $\epsilon \leq \epsilon_i$.

For each value of ϵ_j, we repeat the above procedure N times to obtain N models that had different random initialization and have been trained and evaluated on different samplings of the training set $\mathcal{X}_{\epsilon_j}$ and $\mathcal{X}_{\epsilon_1} ... \mathcal{X}_{\epsilon_k}$, respectively. Next, we create a set of valid solutions for each ϵ_j, denoted by $\mathcal{V}_{\epsilon_j}$, containing models that were trained on $\mathcal{X}_{\epsilon_j}$ achieved 100% accuracy. We also create, for each ϵ_j, sets of robust solutions, $\mathcal{R}_{\epsilon_j,\epsilon_1}, ..., \mathcal{R}_{\epsilon_j,\epsilon_k}$, corresponding to different levels of robustness and containing models that achieve 100% accuracy on $\mathcal{X}_{\epsilon_1}, ..., \mathcal{X}_{\epsilon_k}$, respectively. Using these sets we estimate the following probabilities:

- $P(\text{robust}|\epsilon_i, \epsilon_j) \approx \frac{|\mathcal{R}_{\epsilon_j,\epsilon_i}|}{|\mathcal{V}_{\epsilon_j}|}$ – the probability of (quasi) randomly arriving at an ϵ_i-robust solution
- $P(\text{valid}|j) \approx \frac{|\mathcal{V}_{\epsilon_j}|}{N}$ – the probability of arriving at a valid solution by training on $\mathcal{X}_{\epsilon_j}$
- $P^*(\text{robust}|\epsilon_i, \epsilon_j) \approx \frac{|\mathcal{V}_{\epsilon_i}|}{|\mathcal{V}_{\epsilon_j}|}$ – the oracle probability of robustness. This is the upper bound of $P(\text{robust}|\epsilon_i, \epsilon_j)$ that is achieved if $\mathcal{V}_{\epsilon_{i<j}} \subset \mathcal{V}_{\epsilon_j}$, i.e. all the solutions achievable by training on $\mathcal{X}_{\epsilon_i}$ are recovered by training on $\mathcal{X}_{\epsilon_j}$

3.4 Experiments

In our experiments, we use Multi-Layered Perceptrons (MLP) without biases. We removed the bias parameter because it is not required to compute RXOR and removing it reduces the parameter count, which in turn reduces the memory and computation requirements of our experiments. We ran experiments with various model architectures in order to observe the impact of width, depth, and activation functions on the probability of arriving at a robust solution. We set the radius of the L_2 norm ball from which the initial values of the parameters are sampled to 25. We optimize the initial parameters by performing 20 updates using SGD with a Nesterov momentum of 0.9.

To train and test the models we create datasets $\mathcal{X}_\epsilon$, as described in the previous section, for $\epsilon \in \mathcal{E} = \{0.4, 0.3, 0.2, 0.1, 0.05\}$ with each data set having 1000 points. We denote these 2D datasets as $\mathcal{X}^2_\epsilon$ for $\epsilon \in \mathcal{E}$. We train each model architecture on each of the datasets separately but we evaluate each trained model on all of the datasets. This allows us to determine what fraction of models trained on $\mathcal{X}_{0.4}^d$ achieve 100% accuracy on $\mathcal{X}_{0.1}^d$. We will refer to such models as being ϵ-robust for $\epsilon = 0.1$. Of course, all models that are trained on $\mathcal{X}_{\epsilon_i}^d$ will be ϵ-robust for $\epsilon \geq \epsilon_i$ but only some of them might be robust for $\epsilon \leq \epsilon_i$.

To estimate the $P(\text{robust}|i, j)$ we perform $N = 25000$ trainings. In each training and testing set, the initial parameters of the model are sampled randomly from their respective distributions.

3.5 Results

3.5.1 Model Architecture Notation

In this section, we adopt the following notation to refer to the model architecture. We use "MLP-w_1_..._w_L-f_a" to refer to a MLP with L layers having widths $w_1, ..., w_L$ and activation function f_a. In the case dropout Srivastava et al. [2014] (with probability p) or batch norm Ioffe and Szegedy [2015] is used, the notation is changed to "MLP-w_1_..._w_L-f_aDropout$p$" or "MLP-w_1_..._w_L-f_aBN", respectively. If skip connections Srivastava et al. [2015] are introduced that connect layer i_1 to i_2, i_2 to i_3 and so on, then the notation becomes "MLP-w_1_..._w_L-f_a-wSkip_$i_1 > ... > i_k$". The notation used for CNNs is "CNN-$n_1 \times h_1 \times w_1 \times f_1 - s_1$_..._$w_L n_1 \times h_1 \times w_1 \times f_1 - s_1$-$f_a$" where h_i and w_i are the height and width of the convolutional kernel, f_i is the number of filters, s_i is the stride of the kernel and n_i is the number of consecutive layers that have the configuration $h_1 \times w_1 \times f_1 - s_1$. For example, Conv-1x7x7x32-3_2x7x7x16-3-ReLU represents a model with 3 layers, the first of which has a kernel of size 7x7, stride 3 and 32 filters while the remaining layers have the same kernel size and stride, but 16 filters instead of 32.

3.5.2 Probability of Robustness of Minimal Models

Here we present results from experiments conducted on *minimal* models i.e. having the number of parameters that are necessary and sufficient for solving RXOR. The minimal model for 2D inputs consists of 1 hidden layer with 3 ReLU units as discussed in section 3.2.1. Figure 3.2 shows how $P(\text{robust}|\epsilon_i, \epsilon_j)$ and $P^*(\text{robust}|\epsilon_i, \epsilon_j)$ vary as the training and testing data margins, ϵ_j and ϵ_i respectively, are varied for 2D data. We see that for a fixed ϵ_j, both $P(\text{robust}|\epsilon_i, \epsilon_j)$ and $P^*(\text{robust}|\epsilon_i, \epsilon_j)$ decrease rapidly as ϵ_i is decreased to values less than ϵ_j. However, $P(\text{robust}|\epsilon_i, \epsilon_j)$ falls faster than $P^*(\text{robust}|\epsilon_i, \epsilon_j)$ and as a result as ϵ_i decreases, $\frac{\mathcal{V}_{\epsilon_i} \backslash \mathcal{V}_{\epsilon_j}}{\mathcal{V}_{\epsilon_i}}$ i.e. the proportion of valid solutions for $\mathcal{X}_{\epsilon_i}$ that are recovered by training on $\mathcal{X}_{\epsilon_j}$, also decreases. Stated another way, for a given ϵ_j, as ϵ_i is decreased not only does the number of *possible* ϵ_i-robust solution decreases but the ability of the training mechanism to actually find these solutions diminishes. This indicates that perhaps better training and sampling methods might improve the odds of arriving at ϵ_i-robust solution while keeping ϵ_j fixed. On the other hand, we note that for a given ϵ_i both $P(\text{robust}|\epsilon_i, \epsilon_j)$ and $P^*(\text{robust}|\epsilon_i, \epsilon_j)$ increase as ϵ_j is decreased, which implies that regardless of the value of ϵ_i, reducing ϵ_j can improve the odds of finding an ϵ_i-robust model.

From these experiments, we see that there are two factors that lower the probability of finding ϵ-robust models. Firstly, the number of valid solutions to $\mathcal{X}_\epsilon$ decreases rapidly as ϵ is decreased. Secondly, there is a substantial gap between $P(\text{robust}|\epsilon_i, \epsilon_j)$ and $P^*(\text{robust}|\epsilon_i, \epsilon_j)$, indicating that better sampling and optimization methods are required to close this gap. In the remainder of this

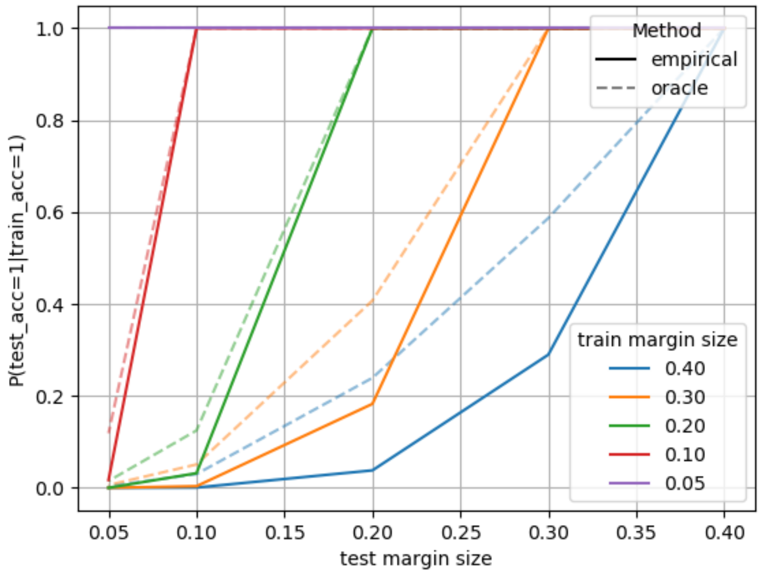


Figure 3.2: $P(\text{robust}|\epsilon_i, \epsilon_j)$ and $P^*(\text{robust}|\epsilon_i, \epsilon_j)$ curves for different values of ϵ_i (Test Data Margin) and ϵ_j(Train Data Margin) for the minimal model for 2D RXOR.

section, we will focus on the first factor. Specifically, we shall present experimental results that indicate the effect of modeling choices, such as network width, depth, and activation function, have on $P(\text{robust}|\epsilon_i, \epsilon_j)$ and $P^*(\text{robust}|\epsilon_i, \epsilon_j)$. We include the latter in our analysis to differentiate the following two cases: (1) a robust solution cannot be found if the training and testing distributions are very different, but can be found if the distributions are made similar, and (2) a robust solution can not be found even if the training and testing distributions are identical, i.e. the model architecture is insufficient for modeling the data.

3.5.3 Influence of Modeling Choices on Probability of Robustness

In this section, we present experimental results that show the influence different modeling choices have on $P(\text{robust}|i, j)$ and $P^*(\text{robust}|i, j)$. The modeling choices we consider here are the width of the network i.e. number of units in each layer, and the depth of the network i.e. the number of layers. In Appendix A.2.1 we extend our analysis to the choice of activation function, the dropout probability, the use of batch normalization, and the use of skip connections. To improve the clarity of presentation, in the remainder of this section we do not show the curves for $P(\text{robust}|\epsilon_i, \epsilon_j)$ and $P^*(\text{robust}|\epsilon_i, \epsilon_j)$ but, instead, we summarize the curve for each training margin (ϵ_i) by computing the area under it and presenting it as a bar chart in Figure 3.3. For a given ϵ_j, we denote the area under the $P(\text{robust}|\epsilon_i, \epsilon_j)$ and $P^*(\text{robust}|\epsilon_i, \epsilon_j)$ curves as AUC_{ϵ_j} and $\text{AUC}^*_{\epsilon_j}$ respectively.

Width

Figure 3.3a shows AUC_{ϵ_j} and $\text{AUC}^*_{\epsilon_j}$ for models consisting of one hidden layer containing 3, 6, and 9 ReLU units and trained on 2D data. We note that as the width of the network is increased

$P(\text{robust}|\epsilon_i, \epsilon_j)$ also increases, albeit at a decreasing rate. Nevertheless, the impact of increasing the number of units from 3 to 6 is substantial – AUC_{ϵ_j} almost double and $P(\text{robust}|0.05, 0.4)$ increases from almost 0% to 3% and $P(\text{robust}|0.1, 0.4)$ increases from 0.5% to 12%. These improvements take randomly achieving ϵ-robustness for small ϵ from being near impossible to being reasonably probable – one can expect to find a 0.9-robust model for every 9 models trained on $\mathcal{X}_{0.4}$.

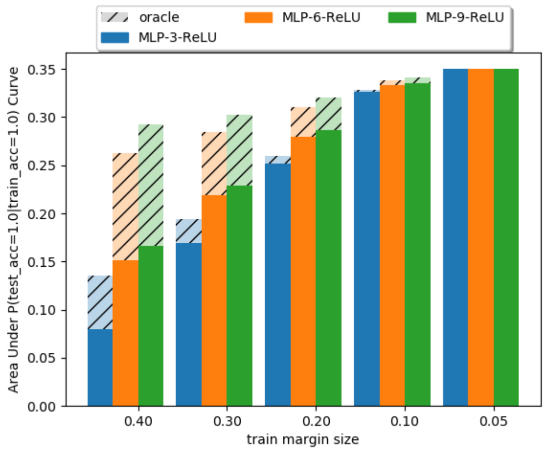
Turning our attention to $\text{AUC}^*_{\epsilon_j}$, we note that $\text{AUC}^*_{\epsilon_j}$, and consequently $P^*(\text{robust}|\epsilon_i, \epsilon_j)$, increases very rapidly as the width is increased. We also note that improvement in $P^*(\text{robust}|\epsilon_i, \epsilon_j)$ outstrips the improvement in $P(\text{robust}|\epsilon_i, \epsilon_j)$, and this gap only grows as the width of the network increases. This indicates that while wide models can find a large number of solutions when trained on $\mathcal{X}_{\epsilon_i}$ for small ϵ_i, they are unable to recover most of them when they are trained on $\mathcal{X}_{\epsilon_j > \epsilon_i}$.

Depth

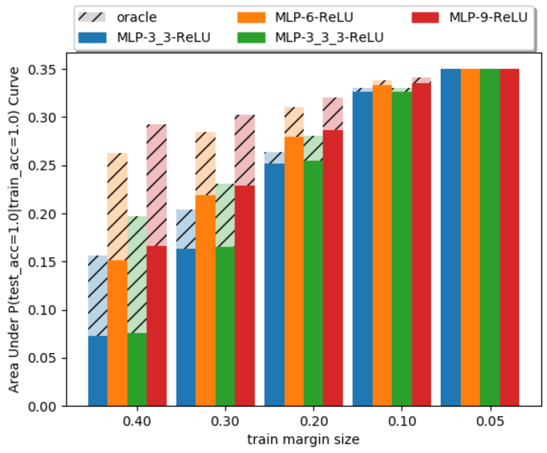
Figure 3.3b shows AUC_{ϵ_j} and $\text{AUC}^*_{\epsilon_j}$ for four models, namely, MLP-6-ReLU, MLP-3_3-ReLU, MLP-9-ReLU and MLP-3_3_3-ReLU (see Section 3.5.1). Note that the first two and the last two models have the same number of parameters, 18 and 27 respectively, so differences in $P(\text{robust}|\epsilon_i, \epsilon_j)$ are attributable to a change in the depth of the model. We observe that among models with the same number of parameters but different depths, the shallower models have higher AUC_{ϵ_j} and $\text{AUC}^*_{\epsilon_j}$ than the deeper models, indicating that both valid and robust solutions are more difficult to find for the latter models. Furthermore, we note that the impact of increasing the number of parameters from 18 to 27 depends on whether the additional parameters increased the width of the network or its depth. As noted in Figure 3.3a and again in Figure 3.3b, if the width is increased we see an appreciable increase in AUC_{ϵ_j}, however, if the depth is increased the change in AUC_{ϵ_j} is hardly noticeable. Finally, we observe that increasing the depth of the model does cause $\text{AUC}^*_{\epsilon_j}$ to increase. This indicates that the additional parameters indeed enhance the ability of the model to find solutions for small testing margins, ϵ_i, but only if the training margin, ϵ_j, is close to ϵ_i.

The above experiments indicate that to improve the odds of finding robust solutions the width of the model must be increased and the depth decreased, however, if for some reason the depth can not be decreased one may ask the question which layer should be widened? If resources are plentiful then one may simply widen all the layers, but in case of resource paucity then one may want to know which layers, if widened, lead to the most improvement in the odds of achieving robustness. To investigate this question we use three-layer models trained on 2D RXOR. The base model contains three ReLU units in each layer. We generate three wider models from the base model by increasing the width of each layer, one by one, by a factor of two. The AUC_{ϵ_j} and $\text{AUC}^*_{\epsilon_j}$ for these models data are shown in Figure 3.3c. We note that in most cases, increasing the width of the first layer seems to improve AUC_{ϵ_j} and $\text{AUC}^*_{\epsilon_j}$.

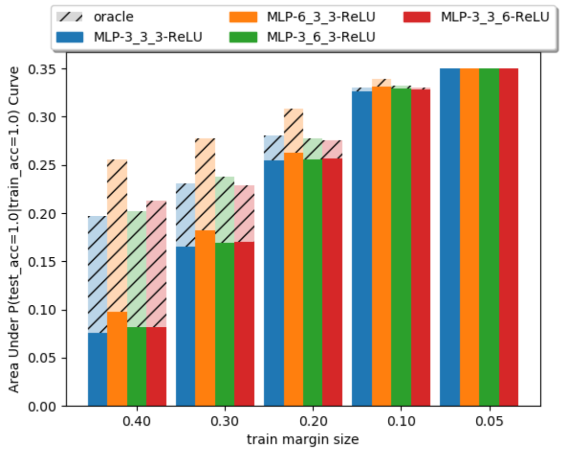
To summarize, the experimental results presented above have yielded the following insights: (1) wider models are likely to be more robust than narrower models, (2) deeper models are likely to be less robust than shallower models, and (3) increasing the width of the earlier layers improves the odds of robustness. These insights together suggest that the odds of robustness may be improved by first finding the minimal architecture for the task in terms of the number of layers and then widening this architecture as much as possible.



(a)

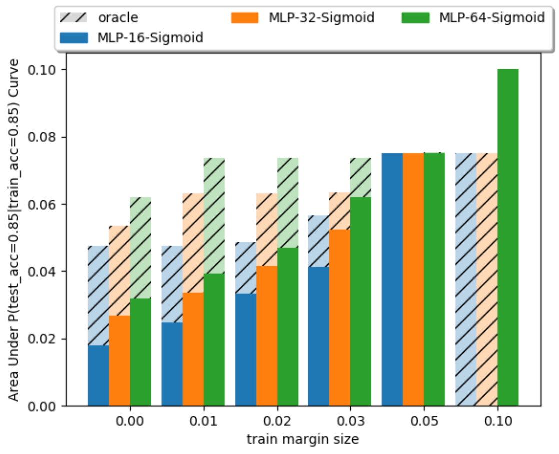


(b)

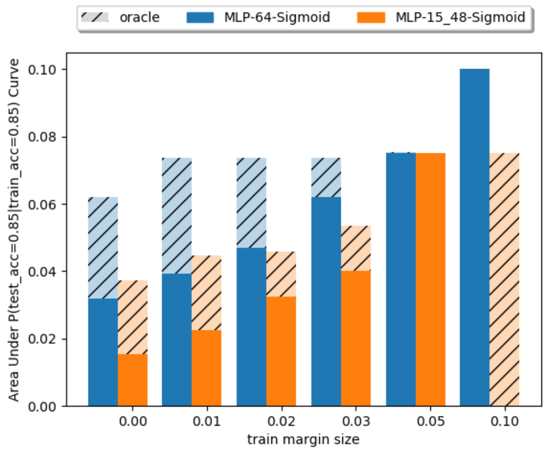


(c)

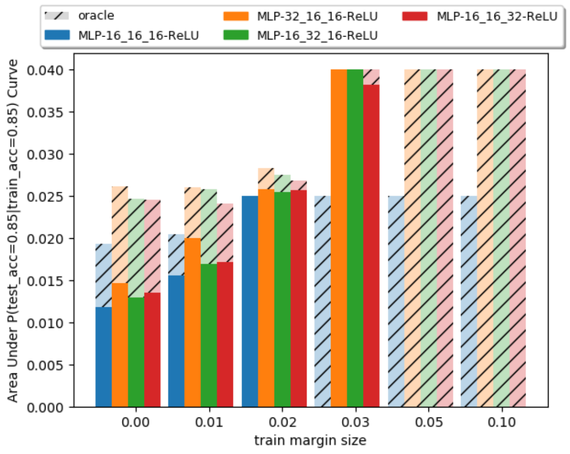
Figure 3.3: The influence of various modeling choices on $P(\text{robust}|\epsilon_i, \epsilon_j)$ as measured by AUC_{ϵ_j} and $\text{AUC}^*_{\epsilon_j}$ for MLPs trained on 2D RXOR data. Subfigure (a) shows the areas for single-layer MLPs with increasing width, (b) shows the areas for MLPs with increasing depth, and (c) shows the effect of widening different layers in a 3-layer MLP.



(a)



(b)



(c)

Figure 3.4: The influence of various modeling choices on $P(\text{robust}|\epsilon_i, \epsilon_j)$ as measured by AUC_{ϵ_j} and $\text{AUC}^*_{\epsilon_j}$ for MLPs trained on MNIST. Subfigure (a) shows the areas for single-layer MLPs with increasing width, (b) shows the areas for MLPs with increasing depth, and (c) shows the effect of widening different layers in a 3-layer MLP.

3.5.4 Generalization to More Complex Data and Models

In section 3.5 we considered models trained on a very simple dataset (2D RXOR). However, the datasets and models used in practice are much more complex. If the general trends relating different modeling choices to the robustness potential that are observed on the simple models and datasets generalize to more complex ones, then we can use them as rules-of-thumb when training models for practical tasks. In order to verify if the trends from 3.5 hold when more complex datasets and models are used, we repeat the experiments using larger MLPs trained on the MNIST dataset LeCun et al. [2010]. The methodology for the experiments is similar to the one presented in 3.3 with the following changes:

1. The notion of the margin used in 3.3 is not applicable to the natural image classification task and needs to be modified because, unlike the RXOR problem, the decision boundaries are unknown. Therefore, instead of considering a margin of width ϵ around the decision boundary beyond which all data lies, we consider a margin around the data point. This corresponds to a hypercube having sides of length 2ϵ to be specific, around each input in the dataset from which we sample training and testing data points. To sample hypercube efficiently we run Projected Gradient Descent (PGD) Madry et al. [2018] for a fixed number of steps to find data points within the 2ϵ hypercube that are not correctly classified by the model.

   This change necessitates redefining the data sets as $\mathcal{X}_\epsilon := \{x+\delta | \delta = \arg\max_{\delta:\|\delta\|_\infty \leq \epsilon} \mathcal{L}(f(x+\delta), y)\}$. where x is an image from the dataset, y is the true label of the image, f is the model and $\mathcal{L}$ is a loss function that indicates how close the prediction, $f(x+\delta)$ is to y. If no point is found within the hypercube for which the model makes an erroneous prediction we conclude that the model is ϵ-robust. Due to this change, in the remainder of this section, we use the training margin, ϵ_j, and testing margin, ϵ_i, to refer to *the margin around the data point and not around the decision boundary*.

2. In practice, however, the condition that every point in the hypercube must be classified correctly is unrealistic because currently, 100% accuracy is not achievable on these datasets even for $\epsilon = 0$. Therefore, We relax the definitions of valid and robust solutions such that a valid solution is one that achieves accuracy at least τ_v on the training data, and a robust solution is one that achieves accuracy at least τ_r on the testing data. All the experimental results that follow have been obtained with $\tau_v = \tau_r = 0.85$.

3. We increase the number of SGD updates performed on randomly sampled parameters. This is done because the natural image classification task is more complex, and training accurate models requires more iterations.

Figure 3.4 shows the AUC_{ϵ_j} and $\text{AUC}^*_{\epsilon_j}$ for different modelling choices. We see that the relationship between different modeling choices and, AUC_{ϵ_j} and $\text{AUC}^*_{\epsilon_j}$, that we observed in models trained on 2D RXOR has carried over to models trained on the much more complex MNIST data. Specifically, we see that the likelihood of finding a solution that generalizes to smaller ϵ_i, i.e. an ϵ_i-robust solution for $\epsilon_i > \epsilon_j$, is increased by increasing the width of the model, particularly of the earlier layers. On the other hand, increasing the depth of the network and applying batch normalization reduces the likelihood of finding an ϵ_i-robust solution.

In Appendix A.2.2 we report additional results using the same architectures with more hypaparameters, and with Convolutional Neural Networks as well.

3.6 A robustness prior: redundant features elimination

We have discussed the possibility for robustness to occur spontaneously during training, and how training hyperparameters have a significant influence on the chances that it occurs. This section illustrates with an additional set of experiments how model compression algorithms may increase robustness as a "side effect". It is based on results presented in Shah et al. [2021a].

3.6.1 Description

This section proposes a method to remove the influence of superfluous features from the model. The cornerstone of the approach is the observation that each neuron in the hidden layers of a neural network is a feature detector for the subsequent subnetwork. Based on this observation, we can draw an equivalence between neurons and features, which implies that to remove the influence of a superfluous feature we must remove the neuron that detects it. To identify such neurons we decompose the features learned by a neural network into two components: the redundant information that is already encoded by other features, and the novel information encodes by this feature. Based on this decomposition we select neurons that provide the maximum amount of novel information about the target output and discard neurons that encode redundant or irrelevant information. We modify a neuron pruning technique called LRE-AMC Shah et al. [2021b] to use our neuron selection criterion and remove superfluous neurons from several image recognition models.

We do not reproduce the entire feature elimination algorithm here, as model compression is not our central topic. The interested reader may read Shah et al. [2021b,a] to learn about the complete procedure. What we are interested in are the results of the compressed models on adversarial examples.

3.6.2 Experiments

We used CIFAR10 [Krizhevsky et al., 2009] and MNIST [LeCun et al., 2010] datasets to train and evaluate several well-known deep learning models. Specifically, we trained VGG-16 [Simonyan and Zisserman, 2014] and AlexNet [Krizhevsky et al., 2012] on CIFAR10 and LeNet [LeCun et al., 1998] on MNIST. We trained the models from random initialization to convergence on the clean dataset using stochastic gradient descent, with a learning rate of 0.1 and L_2 regularization weight of 0.005. During training 20% of the training data is used for validation, and if the validation accuracy does not improve for more than 5 epochs the learning rate is halved.

The adversary performs 100 steps of PGD over L_p balls. On CIFAR10 the adversary explored L_∞ balls of $\varepsilon \in \left\{\frac{4}{255}, \frac{8}{255}, \frac{16}{255}\right\}$ and L_2 balls of $\varepsilon \in \{0.5, 1.0, 2.0\}$ with steps size $\frac{\varepsilon}{4}$. We configured LRE-AMC hyperparameters to be $\tau = 4$ and $\lambda = 0.75$. During the fine-tuning steps the Adam optimizer with learning rate 0.0001 and L_2 regularization weight of 0.0001 are used. We apply the Top-Down LRE-AMC scheme: we move up the model, starting from the penultimate layer, shrinking each layer maximally, eliminating features as long as the accuracy remains above a threshold t.

3.6.3 Results

Method	$-\Delta_P$	Acc$_{cln}$	$\mathbb{E}$[Acc$_{rob}$]	Acc$_{rob}$ w/ $\|\varepsilon\|_\infty$			Acc$_{rob}$ w/ $\|\varepsilon\|_2$		
				4	8	16	0.5	1.0	2.0
VGG16									
None	0.0	90.3	1.3	1.4	0.0	0.0	4.0	1.8	0.6
Adversarial training	0.0	74.9	31.6	57.1	37.1	8.6	53.2	27.6	3.5
Randomized smoothing	0.0	82.9	20.6	43.5	13.8	0.8	47.6	16.6	1.0
Compressed model	87.7	85.6	17.7	20.0	17.4	13.3	20.6	19.3	15.8
AlexNet									
None	0.0	77.5	2.8	8.9	0.3	0.08	7.23	0.2	0.06
Compressed model	98.3	72.3	11.1	14.6	10.1	9.5	13.2	9.8	9.2

Table 3.1: The table presents the reduction in the number of parameters ($-\Delta_P$), the accuracy of the model on clean data (Acc$_{cln}$), the average accuracy of our CIFAR10 models on adversarially perturbed data (Acc$_{rob}$ w/ $\|\varepsilon\|_\infty$), and the accuracy of the model adversarially perturbed data for various perturbation sizes (Acc$_{rob}$ w/ $\|\varepsilon\|$).

We present our results in Table. 3.1. We see that removing spurious neurons from the network using our approach greatly improves the model's robustness to L_∞ and L_2 bounded attacks, especially in the case of VGG16-CIFAR10 where using the gradient-based selection criteria improved the average accuracy of the model, on perturbed data by more than 16% (absolute) compared to the standard model and by 8% compared to the model compressed with LRE-AMC. On AlexNet-CIFAR10, vanilla LRE-AMC outperforms our selection criteria. In fact, AlexNet with TD-V provides the best robust accuracy in all our experiments. Past studies Hendrycks and Dietterich [2019] have observed similar trends where AlexNet appears to be more robust to perturbations: we may be seeing the same phenomenon manifested here.

We applied two defenses to VGG16-CIFAR10 for comparison. First adversarial training with PGD [Madry et al., 2018], using 7 steps of size $\frac{2}{255}$ with $\|\varepsilon\|_\infty \leq \frac{8}{255}$. Then randomize smoothing [Cohen et al., 2019] with noise sampled from $\mathcal{N}(0, 0.25I)$. We note that for small values of $\|\varepsilon\|$ these defenses make VGG16 significantly more robust than models pruned with our technique, however for larger values the accuracy of these models precipitously decreases. At $\|\varepsilon\|_\infty = 16$ and $\|\varepsilon\|_2 = 2.0$ our model has an accuracy that is 4.7% and 12.3% greater than the adversarially trained model. In addition to being more robust at higher noise levels, the compressed model also achieves 9.7% greater accuracy on clean data than the adversarially trained model. These results are achieved while training our model on *clean images only*.

It is very important to note that, unlike the other methods, our models were trained on *clean images only*. In fact, our results seem to suggest that techniques that rely minimally on perturbed training data can generalize better to different attack methods and configurations than explicit adversarial defenses, which do not generalize very well to perturbations outside the L_p ball they are trained with. This once again illustrates the accidental robustness phenomenon, without even resorting to multiple experiments.

3.7 Related work

Early research on adversarial attacks and defenses was often conducted on MNIST [Szegedy et al., 2014, Goodfellow et al., 2015, Papernot et al., 2015]. Some of these works have been criticized for relying on artifacts [Carlini and Wagner, 2017]. Hence more recent works have focused on the larger CIFAR10 and ImageNet [Madry et al., 2018, Wong et al., 2020] to design general and scalable defenses.

Other works have studied how robustness to adversarial attacks may arise without adversarial-specific training methods in some contexts. Some have for instance shown how data pre-processing [Guo et al., 2018] or ODE networks [Carrara et al., 2019] may mitigate adversarial attacks. But to our knowledge, no previous work has proposed a general, systematic study of robustness in undefended models, or studied the variance of this robustness when repeating identical experiments.

3.8 Discussion

We have demonstrated the possibility of the spontaneous emergence of adversarial robustness on synthetic and real data. We also have shown how the intrinsic variance of this robustness - a phenomenon we call *accidental robustness* - can be observed and leveraged on small datasets, for example for architecture search. More generally they may be used to explore which phenomena, training methods, and paradigms affect robustness the most: model compression, optimization methods, etc. We will revisit accidental robustness in the next chapter, as we introduce robustness metrics that can help make this phenomenon, and others, more easily observable.

Chapter 4

Evaluating robustness beyond adversarial accuracy

In this chapter we more specifically address the shortcomings of the most common metric for adversarial robustness: *adversarial accuracy*, or accuracy under attack. As is frequently underlined, this metric is an imperfect surrogate for *worst-case adversarial accuracy*, especially in the presence of gradient obfuscation. Unfortunately, worst-case accuracy itself is not a perfect metric for robustness, and it leaves out a lot of relevant information. Worst-case accuracy only reflects whether, in the vicinity of an input point, there is *one* successful perturbation. This can be misleading in the quest for robustness.

For instance, consider a hypothetical defense that, around every point, eliminates 99% of all dangerous perturbations, leaving 1% to find. Surely this defense can be considered useful and has a role to play in designing robust models, but since its adversarial accuracy is 0% it would likely be considered "broken" and dismissed. In other words, the worst-case accuracy metric is biased towards defenses that are totally effective for *some* points, rather than those that are partially effective around *all* points. Whether a point has any or no adversarial perturbation is relevant knowledge, but it leaves out a lot of information. The "broken" defense above may for instance be useful in real-world contexts, where perturbations are harder to craft than on research datasets. Moreover, it may hypothetically be complementary to other "broken" defenses that eliminate a distinct subset of perturbations or improve a defense with non-zero adversarial accuracy.

Our goal in this work is to propose an alternative approach for measuring adversarial robustness. Specifically, we would like to measure not whether around an input x there is *at least one* adversarial perturbation, but rather *how many* such perturbations there are. In other words, we try to obtain some measure of the *size* of the adversarial space. Our candidate metric, *adversarial sparsity*, is proposed for L_2 and L_∞-bounded attacks and relies on the *geometry* of adversarial examples over the L_2 and L_∞ hypersphere. In this chapter we first motivate and detail this choice of metric, then we demonstrate its usefulness in multiple ways. For instance, by revisiting previous works we show that the usual criteria for qualifying defenses as "broken" are too restrictive. Some broken defenses, incapable of improving accuracy by themselves, can indeed contribute to strong ensembles of defenses. We also show that these metrics make the accidental robustness phenomenon more easily observable, notably in the presence of data augmentation methods which can improve robustness in ways imperceptible with accuracy.

4.1 Introduction

What are possible ways to measure the size of the adversarial set around an input? In high-dimensional input spaces, there are not many informative and computationally tractable metrics to estimate this quantity. Traditional measure theory hits considerable difficulties: for instance, within an L_2 ball of radius ϵ around x, the set of adversarial examples is way too small to be estimated with rejection sampling.

A naive approach would be to revert to evaluating with a set of more or less strong attacks and claim for instance that a model robust to FGSM but not PGD likely has fewer adversarial perturbations than a model robust to neither. However, there are good reasons why this approach was abandoned. Many defenses implicitly rely on gradient obfuscation effects [Athalye et al., 2018a] which makes attack convergence artificially difficult without actually defending against perturbations. Using only strong attacks, possibly enhanced with adaptive methods, protects evaluation against such effects.

In our approach, rather than weakening the attack we propose to constrain it to only look at a subset of the set of admissible perturbations Δ. The question we try to answer is: how large a typical subset must be to find an adversarial perturbation in it? We define metrics quantifying the expected size of the subset: the bigger it is, the fewer perturbations there are. We call this metric *adversarial sparsity*.

A major challenge is to define a distribution of subsets and a metric adapted to the adversarial threat model. In section 4.2, we define sparsity formally for L_2 and L_∞ perturbations. In the L_2 case we consider uniform directions u on the L_2 sphere, and adversarial sparsity is the expected angle $\widehat{\langle \delta, u \rangle}$ needed to find a successful perturbation. For L_∞ attacks we consider vertices of the L_∞ cube $u \in \{\pm 1\}^n$, and measure the number of dimensions k to modify to find a successful perturbation. We also discuss in section 4.2 how adversarial sparsity indeed reflects the size of the successful perturbations set.

Like worst-case accuracy, sparsity is not directly tractable, so we must estimate it using modified PGD attacks on a random sample of directions. Our algorithm is detailed in section 4.3. Armed with this tool we revisit multiple models and defenses proposed in prior works on the CIFAR10 dataset (section 4.4). We both include very strong defenses, taken among the strongest models on the RobustBench leaderboard [Croce et al., 2021]; much weaker defenses that are considered "broken"; and undefended models trained with various data augmentation schemes. Using sparsity we discover or strengthen several properties of these models (section 4.5), among which:

- Higher adversarial accuracy does *not* necessarily mean much fewer adversarial perturbations: many improvements on robustness benchmarks consist of removing a few residual perturbations around "almost robust" inputs.
- Most of the defenses "broken" by strong attacks do not reduce *at all* the number of perturbations, indicating that robustness claims on these defenses are spurious.
- Data augmentation methods, known to improve the results of adversarial training [Rebuffi et al., 2021], in fact increase robustness even *without* adversarial training - though not enough to be reflected in adversarial accuracy.

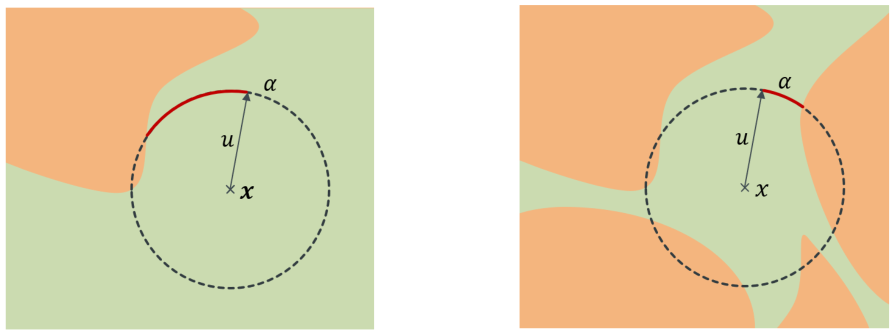


Figure 4.1: Representation of the decision boundary of some binary classification task (regions in green and orange) around point x in two cases. In both cases, the model is not robust around x with the represented L_2 radius ϵ. However, there are many more adversarial perturbations on the right than on the left. Adversarial accuracy evaluates both cases at 0, but α, the angle between fixed direction u and the closest adversarial perturbation, is smaller on the right. Adversarial sparsity is the expected value of α when sampling u uniformly.

These findings could lead to promising developments in future robust models, and confirm the interest in alternative evaluation metrics for adversarial robustness. Sparsity is however imperfect: it has high variance and is computationally expensive. We discuss those limits in section 4.6.

Finally, we also show that adversarial sparsity helps measure the variance in the accidental robustness phenomenon discussed in Chapter 3 (section 4.7).

4.2 Evaluating robustness

4.2.1 Definitions

Throughout the following sections f designates a machine learning model, and L its loss function. ϵ is the radius of the attack, and Δ is the set of admissible perturbations. Usually $\Delta = B_k^n$ with B_k^n the n-dimensional hyperball in norm L_k:

$$B_\infty^n = [0,1]^n \tag{4.1}$$

$$B_2^n = \{x \in \mathbb{R}^n \mid x_1^2 + ... + x_n^2 \leq 1\} \tag{4.2}$$

where x_j is the i^{th} coordinate of x.

To make sparsity computations simpler, we will instead only consider the adversarial examples that use their entire perturbation budget. This means that for $k < \infty$ $\Delta = S_k^n$, using the hypersphere ($\|x\| = 1$) rather than the hyperball ($\|x\| \leq 1$). For $k = \infty$ $\Delta = \{\pm 1\}^n$, i.e. we consider only the (finite) set of vertices of the hypercube. This is done with little cost of generality, as strong attacks like PGD nearly always use all of their perturbation budget (and the simpler FGSM attack is constrained to do so).

Given a point x and a radius ϵ, we define the *adversarial set* as the set of successful admissible perturbations:

$$\text{Adv}(f, x, \epsilon) := \{\delta \in \Delta \mid f(x + \epsilon.\delta) \neq f(x)\} \tag{4.3}$$

In our convention, adversarial perturbations are always unit vectors, to be scaled by factor ϵ when applied. Finally, we can redefine adversarial accuracy over a point x within this formalism:

$$\mathrm{AA}(f, x, \epsilon) := \mathbf{1}[\mathrm{Adv}(f, x, \epsilon) = \varnothing] \tag{4.4}$$

We will also sample points from probability distributions. $\mathcal{U}(X)$ designates the uniform distribution over measurable set X.

4.2.2 Adversarial sparsity

Adversarial sparsity quantifies "how big" a subset of Δ typically needs to be to contain an adversarial perturbation. Formally, we assume access to a sequence of increasing subsets of Δ: with $m_1 < m_2, \emptyset \subseteq \Delta^{m_1} \subseteq \Delta^{m_2} \subseteq \Delta$. We can define adversarial sparsity relative to Δ^m as:

$$\mathrm{AS}(f, x, \epsilon, (\Delta^m)) := \inf\{m \mid \Delta^m \cap \mathrm{Adv}(f, x, \epsilon) \neq \emptyset\} \tag{4.5}$$

If we have a distribution $\mathcal{D}$ of such sequences, we can define adversarial sparsity as the expected value of AS

$$\overline{\mathrm{AS}}(f, x, \epsilon) := \mathbb{E}_{(\Delta^m)\sim\mathcal{D}}[\mathrm{AS}(f, x, \epsilon, (\Delta^m))] \tag{4.6}$$

Intuitively, larger sparsity means more robust models, provided that some reasonable constraints are enforced on distribution $\mathcal{D}$. In particular, the distribution should be isotropic, i.e. not favor any particular direction.

We now give more concrete definitions relative to our two threat models. For L_2 perturbations we consider angular constraints. We sample $u \sim \mathcal{U}(S_2^n)$ and for $\alpha \in [0, \pi]$ we define Δ^α as the spherical cap of direction u and angle α, that is the set of admissible perturbations that form an angle at most α with u:

$$\Delta^\alpha = \mathrm{Sc}(u, \alpha) := \{\delta \in S_2^n \mid \delta \cdot u \geq \cos\alpha\} \tag{4.7}$$

For L_∞ perturbations, we sample both a vertex $u \in \mathcal{U}(\{\pm 1\}^n)$ and a permutation of dimensions $\sigma \in \mathfrak{S}_n$. For $m \in \{1, ..., n\}$ we define Δ^m as the set of perturbations differing of u only over dimensions $\sigma(1), ..., \sigma(m)$:

$$\Delta^m := \{\delta \in S_\infty^n \mid \forall k > m\ \delta_{\sigma(k)} = u_{\sigma(k)}\} \tag{4.8}$$

In other words Δ^m is the set of perturbations differing from u by at most m specific pixels $\sigma(1), ..., \sigma(m)$. The permutation σ is required to enforce that all dimensions are equally probable under the distribution.

4.2.3 Sparsity and number of perturbations

We propose sparsity as an approach to measure the size of the adversarial set. Is there indeed a relationship between the two? Intuitively this depends on how that set itself is distributed over Δ. If for instance $\mathrm{Adv}(f, x, \epsilon)$ is "one half" of Δ (e.g. all perturbations where the bottom left pixel perturbation is positive) then its sparsity will be 0 for half the directions but quite large for many others, leading to an expected sparsity that does not accurately reflect the immense size of this adversarial set. So which conditions should be met for sparsity to be a relevant metric?

Let us consider a simplified case where the set of perturbations is finite: $\text{Adv}(f, x, \epsilon) = \{\delta_1, ..., \delta_k\}$. If we assume that the δ_i are uniformly sampled over the admissible set, then there are no preferred directions of high adversarial concentration, as opposed to the example above. In this case, the link between sparsity and k can be mathematically established. For L_2 perturbations, the expected value of $\text{AS}(f, x, \epsilon)$ when sampling δ_j is:

$$\mathbb{E}[\text{AS}(f, x, \epsilon)] = \int_0^{\frac{\pi}{2}} ((1 - t_\alpha)^k + (t_\alpha)^k) d\alpha \tag{4.9}$$

with $t_\alpha = I_{sin^2(\alpha)}(\frac{n-1}{2}, \frac{1}{2})$ where I is the incomplete regularized beta function. Meanwhile for L_∞ perturbations:

$$\mathbb{E}[\text{AS}(f, x, \epsilon)] = \sum_{m=0}^{n} (1 - 2^{-m})^k \tag{4.10}$$

In this case, we can also show that :

$$\frac{n - \log_2 k}{4} \leq \mathbb{E}[\text{AS}(f, x, \epsilon)] \leq n - \log_2 k + \frac{e}{e-1} \tag{4.11}$$

i.e. sparsity tends to vary like $n - \log_2 k$. We defer all proofs to appendix B.1.

In the general case $\text{Adv}(f, x, \epsilon)$ is infinite[1] and we are interested in its volume relative to Δ, rather than its cardinal. Both situations can be linked if considering for instance

$$\text{Adv}(f, x, \epsilon) = \bigcup_{j=1}^{k} \text{Sc}(\delta_j, \beta) \tag{4.12}$$

ie $\text{Adv}(f, x, \epsilon)$ is the union of spherical caps of equal radius β. β can be called the "local adversarial radius", i.e. how much a typical adversarial perturbation should be changed to recover the original input. In that case, the volume of $\text{Adv}(f, x, \epsilon)$ is equal to $k.V(\text{Sc}(u, \beta))$ (assuming disjoint union) and $\text{AS}(f, x, \epsilon)$ would vary by at most β compared to the finite case. In other words, sparsity depends on both k and β. Therefore comparing the sparsity of two models is akin to comparing the size of their adversarial sets provided that these sets have similar local radii. We implicitly make that assumption in our experiments: estimating the local adversarial radius as well as sparsity would be an interesting extension of this work.

4.3 Algorithms

In this section, we detail the algorithms we use to empirically compute sparsity. We first describe the modifications we implement in the Projected Gradient Descent (PGD) attack to compute adversarial examples in constrained regions. Then we describe the full sparsity computation method, which applies PGD with multiple constraints. We sum up the full procedure in L_2 norm in Algorithm 1.

[1] With our definition of the admissible set, $\text{Adv}(f, x, \epsilon)$ is finite in the L_∞ case. But the same reasoning applies if we consider the more general set $\Delta = S^n_\infty$

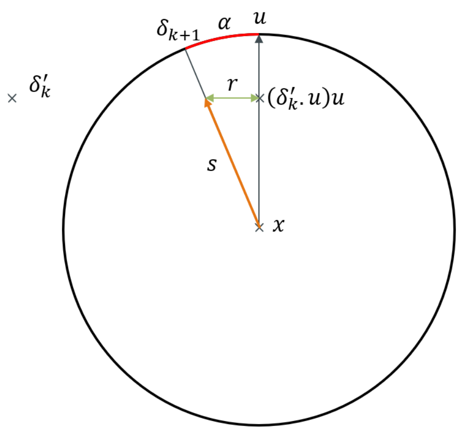


Figure 4.2: Illustration of the angular projection steps on a 2-dimensional sphere, with $\epsilon = 1$. δ'_k is the perturbation before projection, u is the direction of the cap, and δ_{k+1} is the final perturbation, projected both in norm and angle.

4.3.1 Constrained PGD

The PGD attack [Madry et al., 2018] is a strong first-order attack, i.e. it only uses model gradient information. It is fully defined in Chapter 2, in equations 2.2 to 2.6.

To practically compute angular sparsity we need to design a constrained attack, able to project not on a hyperball but on the subsets defined in section 4.2.2.

For L_∞ attacks this is straightforward: given a vertex u, a perturbation σ, and a number of dimensions m, we append after projection an additional step enforcing the constraints:

$$\forall k > m \;\; \delta_{\sigma(k)} \leftarrow u_{\sigma(k)} \tag{4.13}$$

For L_2 attacks, the steps are slightly more complex, as we need an angular projection on the spherical cap of angle α and direction u. This requires us to replace the projection step in equation 2.6 with a projection on a spherical cap (section 4.3.1). This projection consists of the following steps:

$$p \leftarrow \delta'_k - \delta'_k \cdot u.u \tag{4.14}$$

$$r \leftarrow \min(\frac{\|\delta'_k\|_2}{\|p\|_2} \sin\alpha, 1) \tag{4.15}$$

$$s \leftarrow \delta'_k \cdot u.u + r.p \tag{4.16}$$

$$\delta_{k+1} \leftarrow \frac{\min(\epsilon, \|\delta'_k\|_2)}{\|s\|_2} s \tag{4.17}$$

The output of Eq 4.16 is a linear combination of δ'_k and u (with positive coefficients) whose angle with u is $\min(\widehat{\delta'_k, u}, \alpha)$. We then rescale this vector as in standard PGD to obtain δ_{k+1}. In Figure 4.2 we visually illustrate how this procedure returns the closest perturbation to δ'_k of angle at most α with u.

We name PGD$_m$ and PGD$_\alpha$ these constrained attacks in the following.

4.3.2 Computing sparsity over a point

To estimate sparsity for L_2 (resp. L_∞) attacks, given a direction u (resp. u, σ) we explore possible values of α (resp. m) with binary search, and at each step run the constrained attack. We average over multiple sampled directions to estimate $\overline{\text{AS}}$ (in general 100).

The equivalent L_∞ algorithm is similar, replacing only direction u by (u, σ), angle α by a number of pixels m and the constrained L_2-PGD by the constrained L_∞ PGD. The time complexity of these algorithms is discussed in section 4.6.3.

Algorithm 1: L_2 sparsity computation algorithm

```
Require: model $f$, point $(x, y)$, $K \in \mathbb{N}, N \in \mathbb{N}$
  $k \leftarrow 0$
  $i \leftarrow 0$
  while $i \leq N$ do
      $\alpha_0^i \leftarrow 0, \alpha_1^i \leftarrow \pi, u_i \sim \mathcal{U}(S^n)$
      $i \leftarrow i + 1$
  end while
  while $k \leq K$ do
      $i \leftarrow 0$
      while $i \leq N$ do
          $\alpha^i \leftarrow \frac{\alpha_0^i + \alpha_1^i}{2}$
          $x_{\text{adv}} \leftarrow \text{PGD}_{\alpha^i}(f, x, y, u_i)$
          if $f(x_{\text{adv}}) \neq y$ then
              $\alpha_1^i \leftarrow \alpha^i$
          else
              $\alpha_0^i \leftarrow \alpha^i$
          end if
          $i \leftarrow i + 1$
      end while
      $k \leftarrow k + 1$
  end while
  return $\frac{1}{n} \sum_0^n \alpha^i$
```

4.3.3 Computing sparsity over a dataset

Adversarial sparsity only makes sense if there *are* adversarial perturbations in the vicinity of an input x. It is designed to capture the level of vulnerability in non-robust models. We choose to restrict sparsity computation to the residual subset of inputs where the model is vulnerable. For instance, if a model is robust over 50 inputs and has sparsity 0.3 over another 50, we will say it reaches 50% accuracy and *residual sparsity* 0.3. An alternative could be to default sparsity to a max value when evaluating a model that is robust around x ($\text{Adv}(f, x, \epsilon) = \varnothing$): π for L_2 attacks, and dimension n for L_∞ attacks.

1	Model	Defenses	Clean acc.	Adv. acc.	Sparsity	Clean acc.	Adv. acc.	Sparsity
2	R-18	None	88	0	0.180	88	0	56.8
3	R-18	Madry et al. [2018] (1 step)	90	62.1	0.559	90.73	40.08	229
4	R-18	Madry et al. [2018]	88.8	67.3	0.553	80.97	47.19	277
5	R-18	Madry et al. [2018] (w/o aug.)	82.8	60.2	0.532	73.96	43.02	273
6	WR-70-16	Rebuffi et al. [2021] (w/ data)	95.74	82.3	0.581	94.16	62.91	213
7	WR-70-16	Gowal et al. [2020] (w/ data)	94.74	80.5	0.590	91.17	62.5	215
8	WR-70-16	Rebuffi et al. [2021]	92.41	80.4	0.545	89.16	65.83	202
9	WR-28-10	Rebuffi et al. [2021]	91.79	78.8	0.529	89.58	61.66	209
10	R-18	Sehwag et al. [2021]	89.5	73.4	0.539	87.08	52.08	231
11	R-18	Rade and Moosavi [2021]	90.5	76.2	0.515	92.08	57.5	209
12	WR-34-10	Augustin et al. [2020]	92.23	76.3	0.525	-	-	-
13	R-50	Augustin et al. [2020]	91.08	72.9	0.572	-	-	-
14	WR-34-R	Huang et al. [2021]	-	-	-	89.58	57.5	227
15	WR-34-R	Huang et al. [2021] (EMA)	-	-	-	89.17	57.5	230

Table 4.1: Evaluation of state-of-the-art adversarially trained models under L_2 ($\epsilon = 0.5$) and L_∞ ($\epsilon = 0.03$) perturbations. We report Natural accuracy (without attack), adversarial accuracy (under AutoPGD), and adversarial (residual) sparsity. Model architectures are either ResNet (R) or WideResNet (WR) . (w/o aug.) means the model was trained without any data augmentation: most models are trained with at least Crop+Resize augmentation. (w/ data) means that external data was used to train the model. For some defenses there is only a L_∞ or only a L_2-trained model available; for many both exist, in which case we report the results of both.

Defense	Test Accuracy	L_2 Sparsity	L_∞ Sparsity
None	88	0.180	56.8
JPEG	77.1	0.161	80.9
FS	87.8	0.178	60.1
SPS	74	0.147	77.7
FS+SPS	74.5	0.174	86.6

Table 4.2: Evaluation of ResNet18 with various “broken” defenses under L_2 attack with 20 iterations and $\epsilon = 0.5$. We report Natural accuracy (without attack) and angular sparsity for L_2 ($\epsilon = 0.5$) and L_∞ ($\epsilon = 0.03$) perturbations.

4.4 Experiments

We run experiments in L_2 and L_∞ perturbations on the CIFAR10 dataset [Krizhevsky et al., 2009] and CNN-based models. Additional results on more datasets and architectures can be found in Appendix B.2. We use attack radii $\epsilon = 0.5$ and $\epsilon = 8/255$ respectively.

4.4.1 Models and defenses

We mainly evaluate defenses over a classic residual CNN architecture: ResNet-18 [He et al., 2016]. We evaluate larger ResNets and WideResNets as well. We consider multiple defense mechanisms:

Augmentation	Accuracy	L_2 Sparsity	L_∞ Sparsity
None	88.0	0.180	56.8
Crop+Resize	93.7	0.225	64.0
Cutmix	94.58	0.185	50.8
50% Cutmix	95.83	0.202	57.0
Mixup	93.75	0.157	71.1
50% Mixup	92.5	0.202	69.4
Cutout	94.16	0.225	62.2
50% Cutout	93.75	0.225	62.6
Ricap	92.2	0.272	105
50% Ricap	91.67	0.253	83.6

Table 4.3: Evaluation of ResNet18 with data augmentation schemes. The augmentation is applied either on all inputs or 50% of inputs

Adversarial training

Adversarial training consists in applying adversarially perturbed inputs during training rather than or along with natural inputs. It is one of the most robust defenses against PGD attacks. We use multiple pretrained models using state-of-the-art defenses based on adversarial training [Gowal et al., 2020, Augustin et al., 2020, Rebuffi et al., 2021, Rade and Moosavi, 2021, Huang et al., 2021]. These defenses are well ranked in the Robustbench leaderboard [Croce et al., 2021] for L_2 attacks on CIFAR10. Some use additional extra data for pretraining. We also train ResNet-18 models with PGD training, following Madry et al. [2018], with 1 or 10 attack steps.

Preprocessing

Early defenses often used input preprocessing during testing to "erase" adversarial noise. These defenses have mostly been beaten by stronger or adaptive attacks. We revisit some of them: JPEG compression and decompression [Guo et al., 2018], Feature Squeezing, and Spatial Smoothing [Xu et al., 2018]. The latter two were proposed as complementary defenses. This makes analysis with sparsity particularly interesting to determine whether each defense has a partial effect on robustness and would benefit from ensembling.

Data augmentation

Xie et al. [2018] suggest that random transformations can mitigate adversarial attacks. This defense is also considered broken when using strong attacks [Athalye et al., 2018a]. Additionally, Rebuffi et al. [2021] showed that data augmentation can significantly improve the performance of adversarial training.

We wonder if data augmentation affects robustness by itself. We train multiple models with standard training and data augmentation schemes used in Xie et al. [2018] and Rebuffi et al. [2021], and evaluate them with sparsity.

4.4.2 Attack

Preprocessing methods (JPEG, Feature Squeezing, Spatial Smoothing) may pose obfuscation problems when computing adversarial attacks during sparsity estimation. Hence, we follow Shin and Song [2017] and use Differentiable JPEG, which we backpropagate through. We use the Straight-Through estimator [Athalye et al., 2018a] against the other two.

In addition to sparsity, we evaluate adversarially trained models using the AutoAttack [Croce and Hein, 2020b] and $\epsilon = 0.5$ or $\epsilon = 8/255$. To reduce computation time we run a slightly cheaper AutoAttack, using only APGD-CE and APGD-DLR. All other models (weakly defended or undefended) achieve 0% adversarial accuracy, which we do not report in result tables.

4.5 Results

The results of adversarially defended models are reported in Table 4.1. We report averaged values of sparsity over the first 1000 vulnerable inputs in the CIFAR10 test set and, for each input, 100 random directions.

4.5.1 Comparing strong defenses by sparsity

At first sight, we can already observe in Table 4.1 that sparsity is overall consistent with adversarial accuracy. Adversarially defended models, whose accuracy is above 40%, have much higher sparsity than the undefended baseline. Models trained with strong adversarial training (lines 3-15) all reach L_2 (resp L_∞) sparsity greater than 0.5 (resp 200). In comparison, the baseline model achieves 0.180 (resp. 56.8).

However, among the robust models (lines 3-15), sparsity and accuracy variations are not well correlated. In fact, when it comes to L_∞ perturbations, the model with higher accuracy has often *lower* sparsity on the residual subset! We illustrate this phenomenon by plotting for robust models sparsity as a function of accuracy in Figure 4.3. An explanation would be that improvements in adversarial training from a baseline model A focus on robustly classifying the easiest points, i.e. those for which few perturbations fool A. Those points have higher sparsity - thus classifying them well drops the residual sparsity for the remaining points.

4.5.2 Comparing broken defenses by sparsity

In Table 4.2 we compare the sparsity of the vanilla model and models defended with one or more of the “broken” preprocessing-based defenses. Using the strong AutoAttack attack rather than the attacks these defenses’ authors had evaluated, we can break them on all inputs (0% adversarial accuracy) for both L_2 and L_∞ attacks. This weakness to strong attackers was, for all defenses, first pointed out in the articles that successfully broke them [Athalye et al., 2018a, He et al., 2017].

The fact that some defenses are effective against weak attacks like FGSM but not against stronger attacks has two possible explanations. One is that such defenses protect against a subset of all existing perturbations, but not all of them: a stronger attacker can find more subtle perturbations that evade the defense. Another is that defenses don’t actually protect against perturbations at all,

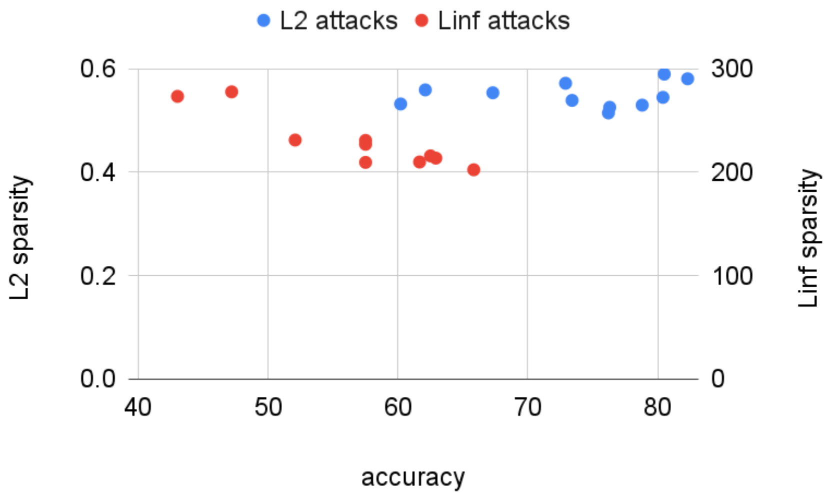


Figure 4.3: Sparsity as a function of adversarial accuracy for all robust models, in both L_2 norm (blue) and L_∞ norm (red). The values used are the same as in Table 4.1

but merely make them perturbations harder to find for attackers by *obfuscating gradients* [Athalye et al., 2018a].

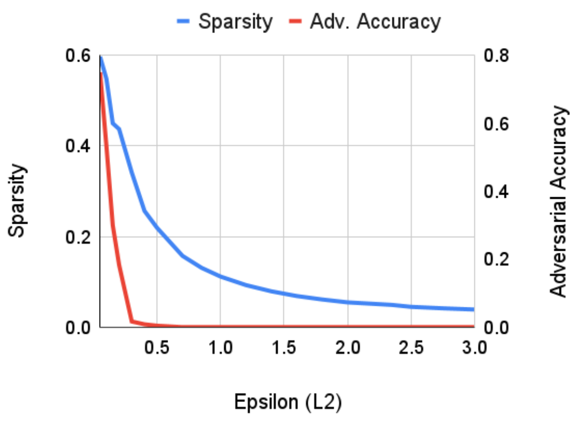
Adversarial sparsity offers a simple way to discriminate between these two situations. We expect a partially effective defense to increase sparsity, but not accuracy. An obfuscation-based defense on the other hand would not improve either of these metrics.

On L_2 attacks, in Table 4.2 we observe that none of these defenses lead to a significant improvement in sparsity. Any L_2-robustness claim regarding these methods is therefore likely to be relying on spurious obfuscation effects. Results are however different on L_∞ attack. While Feature Squeezing does not increase sparsity, JPEG compression and spatial smoothing do ($+20.9$ and $+18.7$ respectively). These defenses are therefore effective against some perturbations. Moreover, feature smoothing does boost robustness when combined with Spatial Smoothing, which was one of the claims of Xu et al. [2018]. We argue that the recent claims that most proposed defenses rely solely on subtle obfuscation effects should be revised. In a few examples, we have shown that these defenses offer non-negligible protection against some perturbations. Even if past works have identified obfuscation effects in these defenses using different methods, such as adaptive evaluation [Athalye et al., 2018a], they do not explain all of their effect on robustness. Sparsity is complementary to adaptive attacks when evaluating defenses.

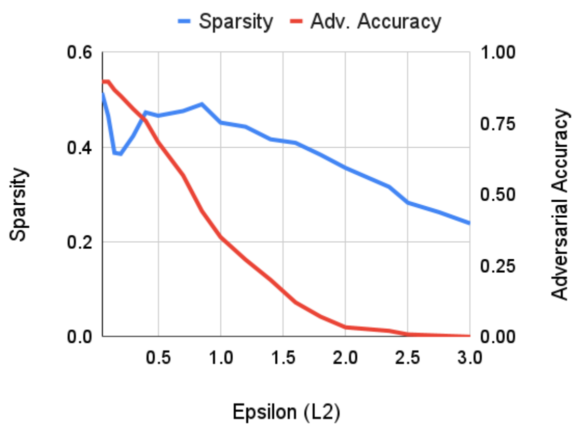
4.5.3 Testing data augmentation on standard models

In Table 4.3 we report the adversarial sparsity of various models trained with data augmentation. We observe that each of them increases sparsity on both L_2 and L_∞ perturbations. The only exception is CutMix [Yun et al., 2019].

Interestingly CutMix is also the best augmentation for robustness when combined with adversarial training, according to Rebuffi et al. [2021]. Moreover, RICAP, which largely outperforms all other augmentations for robustness with standard training in our experiments, is the worst-

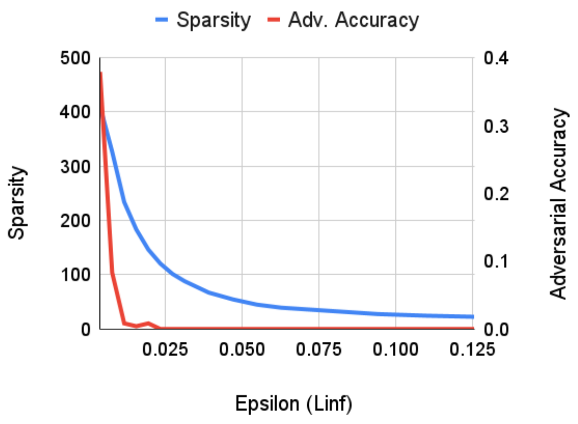


(a) Standard ResNet18 model

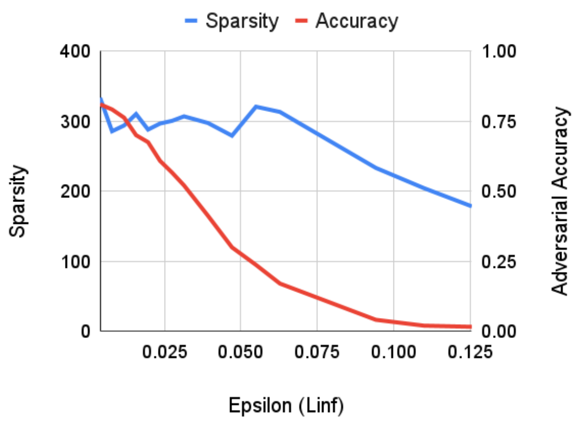


(b) Adversarially trained model (training $\epsilon = 0.5$)

Figure 4.4: Evolution of L_2 sparsity and adversarial accuracy of ResNet18 models as a function of the attack radius ϵ



(a) Standard ResNet18 model



(b) Adversarially trained model (training $\epsilon = 0.031$)

Figure 4.5: Evolution of L_∞ sparsity and adversarial accuracy of ResNet18 models as a function of the attack radius ϵ

performing one for adversarial training! A possible explanation is that despite improving robustness, data augmentation interferes with adversarial training: the best augmentations for robustness won't necessarily combine harmoniously with the best training schemes. This explanation is consistent with past works which had shown that augmentation leads to no robustness improvement [Rice et al., 2020]. One of the contributions of Rebuffi et al. [2021] was to overcome this issue with methods like weight averaging.

Our results suggest that the potential of RICAP augmentation for robustness is still underexplored. Given its considerable effect on the distribution of adversarial examples, it is likely that a version of RICAP+Adversarial training could outperform CutMix+Adversarial training if the aforementioned interference effects are overcome.

4.5.4 Influence of the attack radius

Another interesting application of sparsity is to provide a smoother robustness metric than accuracy. We illustrate this in Figures 4.4a and4.5a, where we plot both sparsity and accuracy for L_2 attacks for increasing values of the attack radius ϵ, and a standard ResNet18. We observe that while accuracy drops to zero after a small value of ϵ, sparsity decreases much more slowly, only converging to 0 asymptotically.

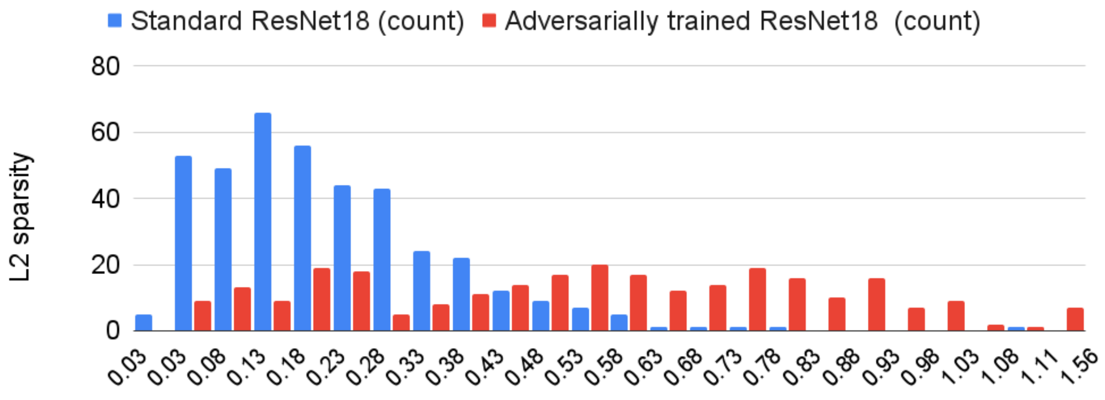
This property lets us compare models on a much larger range of perturbations. In Figures 4.4b and 4.5b we plot the equivalent plot for an adversarially trained ResNet18. On large ϵ values accuracy provides little information, being null or close to null for both standard and robust models. On the other hand, we easily observe that sparsity is over five times higher for the adversarially trained model. This shows that training models adversarially with a given attack radius is beneficial even for much larger radii.

In Figures 4.4b and 4.5b we also observe an interesting fluctuation in the sparsity curve. Although sparsity overall decreases when the attack radius increases overall, the training radius $\epsilon = 0.5$ (L_2) or $\epsilon = 8/255$ (L_∞) appears to be a local sparsity maximum. One interpretation is that for some points, the effect of adversarial training is to remove adversarial perturbations from the unit hypersphere, but not inside it. This leads to a strange effect where the model gets a bit more vulnerable when decreasing the attack radius. Because adversarial perturbations in sparsity are restricted to points on the hypersphere, it is particularly affected.

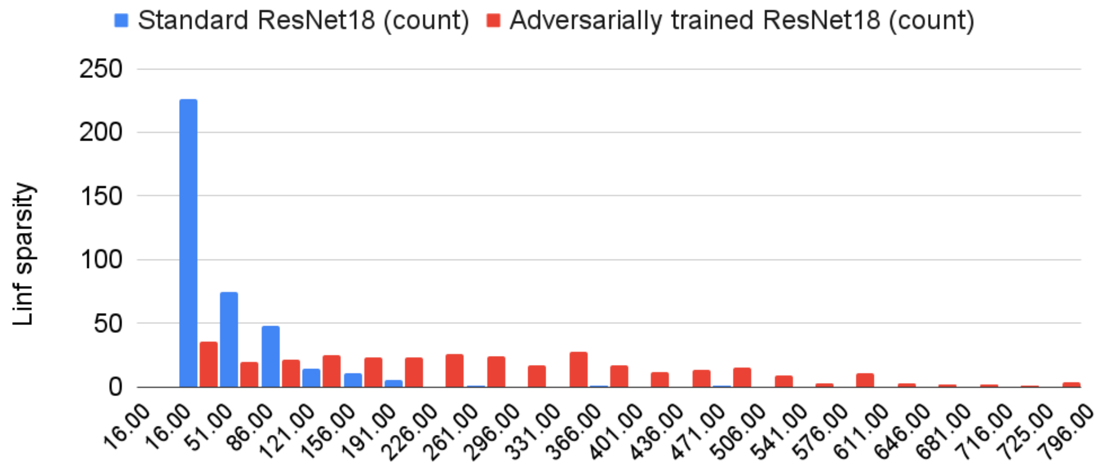
4.6 Analysis

4.6.1 Margin of error when estimating sparsity

Angular sparsity $\overline{\text{AS}}$ cannot be directly computed but must be approximated with the sample mean estimator over multiple directions. To assess the margin of error in this estimation we compute for multiple models, the standard deviation of L_2 sparsity over 100 directions, for 1000 input points. We find that these deviations are consistently lower than 0.022. Using the general formula for margins of errors $z * \frac{\sigma}{\sqrt{n}}$, we conclude that our estimates with 100 directions provide estimates within a ± 0.002 margin with 95% confidence. In section 4.6.2 we provide additional information on the sparsity variance, this time with respect to input points.



(a) L_2 sparsity



(b) L_∞ sparsity

Figure 4.6: Histogram of sparsity values for standard and adversarially trained ResNet18 models. It illustrates that sparsity variance with respect to input points is very large, especially for adversarially trained models.

4.6.2 Point-wise variance

When varying the input point there are significant variations in the value of the metrics. Taking the vanilla model evaluated over 100 inputs, for L_2 (resp. L_∞) sparsity we observe a standard deviation of 0.144 (resp. 52.8) with respect to data points. Keeping in mind the considerations in section 4.2.3, this hints that the adversarial set is considerably larger for some points than others. The deviation is even larger for adversarially trained models. This is clearly visible in Figure 4.6, where we plot the histogram of sparsity values of a standard and adversarially trained ResNet18 models.

Interestingly, there seems to be a correlation between sparsity on the vanilla model, and accurate prediction on a robust (adversarially trained) model. Using a threshold-based classifier on the vanilla model sparsity to predict whether the robust classifier predicts them correctly, we can reach a precision of 67% at Equal Error Rate. This demonstrates an additional property of adversarial sparsity: to discriminate points that are "easy" to learn with a robust decision boundary (the points with a sparse adversarial set) from harder ones.

4.6.3 Time complexity of sparsity computation

On CIFAR10, computing adversarial sparsity around an input point with 100 directions, 10 search steps, and 20 PGD iterations takes a few seconds for a ResNet-18 model on an Nvidia RTX 2080 Ti. For reference, we find it shorter than running the AutoPGD attack [Croce and Hein, 2020b] with default hyperparameters in its worst-case scenario (i.e. when there is no adversarial example).

Dependence on input dimension

Everything else equal, the duration of sparsity computation in a given direction is proportionate to the duration of one backward pass on the model. It is therefore not significantly longer on ImageNet than on CIFAR10 when the same model architecture is used, the only difference being the input size. Of course, reaching good performance on ImageNet requires (to this day) larger models than on CIFAR10: the current state-of-the-art models have about 2B parameters [Dosovitskiy et al., 2021], while our largest models in this work have less than 100M. This would impact the practical cost of sparsity computation.

One could also think that the number of samples required to confidently evaluate sparsity is much larger in higher dimensions. However, our algorithm does not require to sample in "every" direction, or enough to break the "curse of dimensionality". Sparsity in practice has low variance with respect to the direction (see Section 4.6.1). Therefore only a few samples are required to estimate with a reasonable margin of error.

Improving the complexity of sparsity

While attacks over multiple directions can be computed in batches, binary search constitutes the major bottleneck of this algorithm. Some heuristics could help speed up sparsity computation. For example, we could use batched n-Ary search with fewer directions to find a first approximation of sparsity, then confirm/refine it with more directions. As we have mostly worked with reasonably sized models and inputs we have not experimented with such heuristics.

4.7 Measuring variance in accidental robustness with sparsity

In the previous chapter, we discussed how adversarial robustness varies over identical trainings in several cases. We however only experimented with synthetic datasets as well as MNIST. The reason was that on larger datasets like CIFAR10, accidental robustness cannot be observed with standard training (the adversarial accuracy under any challenging radius is always 0). Sparsity helps us observe accidental robustness in such settings.

We train 100 ResNet18 models over CIFAR10 for 30 epochs, without any data augmentation. Each model achieves between 86% and 88% accuracy. We then compute sparsity over the first 100 points of the test set.

Over all models, and averaged over all data points, we report sparsity values between 0.165 and 0.196 with mean 0.176 and deviation 0.005. This average is consistent with the sparsity achieved in Table 4.1, line 2. Moreover, we can confirm that these differences are not due to an overfitting effect on these 100 points. We take the most robust (sparsity 0.123) and least robust (0.089) models based on these metrics, and recompute their sparsity on 100 different points. We achieve 0.088 and 0.126 respectively, indicating consistent differences in sparsity over the test set between these two models.

The difference in robustness induced by sparsity can also be observed with inter-model transferability, as perturbations computed on robust source models and/or applied to non-robust target models tend to transfer better. For each model, we compute adversarial examples over 10 input points, then transfer these examples onto all other models. For all 4950 source/target pairs of models, we measure *full transferability*, i.e. whether *all 10 examples* fool the target model. On average it is the case for 92% of all pairs. However, if we constraint that the source model is in the top 10 least sparse models, and the target in the top 10 most sparse, only 78% of pairs show full transferability. From the most to least sparse on the other hand, we have full transferability for 98% of pairs. This shows that the least sparse models are indeed more robust.

4.8 Related Work

4.8.1 Geometry of adversarial examples

The idea to explore the spherical geometry of adversarial examples has some precedents in the literature. Tramèr et al. [2017] take a linear algebra approach and estimate the dimension of a subspace of adversarial examples around a perturbation. Khoury and Hadfield-Menell [2018], and to an extent Samangouei et al. [2018] explore the data manifold and attempt to explain adversarial examples as out-of-manifold points that models have trouble classifying. Such approaches that consider adversarial perturbations as artifacts have been challenged by works such as Ilyas et al. [2019] which demonstrate that adversarial perturbations are features that models can learn, and as such are reasonable input points.

We do not make claims about the “nature” of adversarial examples, but simply propose metrics to quantify them. This makes this work closer to Tramèr et al. [2017] in its principle, although sparsity is different than dimension estimation. Parallels may also be drawn with Deep Hypersphere Embeddings [Liu et al., 2017], which have been shown to have some robustness properties [Pang et al., 2020].

4.8.2 Alternatives to adversarial accuracy

Minimal perturbation: it is common to compute the closest adversarial example to a given input as a metric for robustness around this point. Some attack algorithms are based on that approach Carlini and Wagner [2016]. This notion differs from sparsity in that it removes the notion of attack bound and perturbation constraint altogether: instead, all points are potential adversarial examples and the attacker finds the "best" one. A model could have a very small minimal perturbation but still have high sparsity for large radii.

Probabilistic robustness: Robey et al. [2022] introduce a relaxation on "worst-case" robustness, by training models to be robust to most perturbations sampled uniformly, rather than all of them.

Stemming from similar motivations to our work, probabilistic robustness differs from sparsity significantly in its approach, as it is "non-adversarial" robustness. Where we sample geometric regions to constrain the adversary, probabilistic robustness samples perturbations directly. Indeed, Robey et al. [2022] show that probabilistic robustness and adversarial robustness are largely uncorrelated, contrary to sparsity and accuracy.

Local Intrinsic Dimensionality (LID): Ma et al. [2018] analyze the local dimension of the submanifold of adversarial perturbations around an input. They use this metric to better understand the properties of adversarial regions. sparsity differs from LID as it does not assume that adversarial examples lie on a manifold. We probe regions and try to find adversarial examples in them, while LID starts from an adversarial example and probes the "adversarial region" around it. These approaches are intuitively complementary, in that sparsity estimates "how many" adversarial regions there are and LID "how big" those regions are. This duality is an interesting research topic for future work.

4.9 Discussion

In this chapter, we have proposed a novel metric, adversarial sparsity. We have shown that it is complementary to adversarial accuracy and offers more insight into “weak” defenses. We have used it to exhibit intriguing properties of adversarial perturbations and defenses. With sparsity, we have also shown that methods like data augmentation have an impact on accidental robustness. While accuracy under attack should likely remain the primary metric for benchmarking strong defenses, we believe that using finer metrics in the evaluation process can benefit the research field.

Sparsity can be seen as fixing the "spikiness" of accuracy with respect to each point. When it comes to Speech recognition, our main focus in all following chapters, even in non-adversarial settings accuracy is a poor metric to use. Getting all words correct in a transcription is highly challenging, and we benefit from measuring instead the proportion of correctly transcribed words. This information is captured by the **Word-Error-Rate**. We use that metric primarily in all of our work on speech recognition, thus using some of the insights that sparsity provides on images.

It would of course be possible to directly use sparsity on ASR. This however would induce a significant computational cost. It is worth exploring the motivations justifying that effort: how much of a danger do adversarial attacks represent for ASR? The next part focuses on answering that question.

Part II

Evaluating threats against speech recognition

Chapter 5

White-box attacks on speech recognition

After studying evaluation methods for robustness in general in Part I, we dive into attacks on Automatic Speech Recognition (ASR) in particular. The adversarial robustness of neural ASR has been an active research topic for several years and has brought about many interesting research directions (see section 2.4). However, ASR models are more versatile than image classification models: many different architectures, training objectives are still actively used (section 2.3). In addition, complex deployment methods and real-time transcription constraints make adversarial attacks tricky to run on ASR models. Robustness evaluation has often been local and limited to outdated model architectures, with threat models varying significantly between different studies. As a result it is hard to extract generalizable insights on ASR robustness from existing works. Our main objectives in this part are to systematize the evaluation process of adversarial attacks on ASR and to empirical robustness properties with well-defined application domains, that encompass recent state-of-the-art architectures.

Like Part I, this chapter focuses on white-box attacks (section 2.1.2), the upper bound of all attack algorithms, and the golden standard of adversarial evaluation. We answer questions such as: are some ASR architectures more robust than others? Does scaling training data and iterations increase adversarial robustness? And more generally, is the ASR field currently heading towards more or less robust models? We also evaluate the practicability of white-box attacks, which have the shortcoming of requiring model gradients. We show that this threat model remains of significant interest when applied to popular open-source models, and even has ripple effects on threats like privacy attacks. The extent to which the vulnerabilities we discuss here extend to contexts where model gradients are unavailable is the focus of the next chapter.

5.1 Introduction

Adversarial attacks against ASR models have been around for years and their results on baseline models like DeepSpeech2 [Amodei et al., 2016] or toy datasets like Speech Commands [Warden, 2018] are well known [Carlini and Wagner, 2018, Alzantot et al., 2018]. Research on Speech Recognition has since then hit multiple milestones by adopting Transformer models [Vaswani et al., 2017], looking into various architectures [Gulati et al., 2020] and even self-supervision [Baevski et al., 2020]. Yet most works on ASR attacks still often evaluate them on the same baseline models or a very limited number of models [Taori et al., 2019, Qin et al., 2019, Xie et al., 2021]. Rare exceptions typically focus on specific types of attacks, like universal perturbations [Lu et al., 2021]. This leaves a number of fundamental questions largely unanswered: how do LSTMs and Transformers compare in terms of robustness? What about encoder-decoders and encoder-only models? Does self-supervised pretraining, a staple of recent ASR systems, affect adversarial vulnerabilities? What about increasing the amount of training data, or changing the languages of that data? Works like Abdullah et al. [2021b] have shown that ASR adversarial examples are different from image ones in key aspects, and must therefore be the object of specific research. In that regard, setting the foundations for a standardized, systematic evaluation of attacks on speech models is an essential first step.

Our first contribution in this chapter consists in laying out some of these foundations (section 5.2). We evaluate targeted and untargeted white-box attacks on the LibriSpeech test-clean set against multiple standard models trained on the same data but varying in architecture, loss, training paradigm, and size. These results both set a standard to beat in future attack and defense works and let us draw some (cautious) conclusions on the relative robustness of ASR architectures.

Though they bring valuable insight, those experiments still focus on models trained in "simple" settings (one dataset of fairly clean audio) and may not be representative of the ASR models that real-world applications tend to use. We fix this limitation by focusing on the recent Whisper model [Radford et al., 2022] in sections 5.3 and 5.4. A key property of Whisper training is the scaling of its training data. Thanks to its massive training on diverse sources of audio, Whisper achieves very impressive robustness against noise and out-of-distribution data. We however show that in terms of adversarial robustness, this approach yields no benefits against untargeted attacks, and limited improvements against targeted attacks (section 5.3). Another novel feature of Whisper training is its massively multilingual training data. We show that this approach is not without drawbacks: multilingual Whisper models can easily be fooled to detect the wrong language (section 5.4). This very simple attack is sufficient to entirely mistranscribe sentences in low-resource languages. We even propose a *universal* version of this attack, with a single perturbation that can fool multiple speech utterances.

How much danger can those white-box attacks cause in practice? Whisper is open-source: if it is vulnerable to adversarial attacks, then its deployment in real-world applications could turn these potential dangers into real liabilities. Besides, causes for concern are unfortunately not limited to direct applications of white-box attacks. We also show that they can be used indirectly for privacy attacks (section 5.5). Specifically, we show on some models that a model's loss on adversarially perturbed inputs is a powerful feature to run membership inference (MI) attacks [Shokri et al., 2017] on those inputs, i.e. determine whether the utterance or its speaker is part of the model's training data. MI can be used by ill-intended parties to determine the presence of specific speakers in the training set of an Automatic Speech Recognition (ASR) model. This may pose a privacy breach if the mere presence of an individual in this dataset provides sensitive information about them. Thus privacy concerns are an additional reason why adversarial robustness in models may be desirable.

One of the limitations to a standard evaluation of ASR attacks until this point was the lack of a practical codebase to do so. Another of our contributions consists in releasing an ASR robustness framework, which we term `robust_speech`. Based on SpeechBrain [Ravanelli et al., 2021], this package can apply the same attack algorithms to most neural architectures and datasets, which is an important step towards setting the foundations of standardized adversarial attacks on ASR. More details about `robust_speech` are provided in Appendix C.1

5.2 Comparing the robustness of neural architectures

5.2.1 Experimental setup

Dataset

We use the LibriSpeech dataset [Panayotov et al., 2015] for evaluation. All the models we evaluate in this section are trained on that dataset. We evaluate our adversarial attacks on the clean test set. For the more computationally expensive CW attack, we restrict our evaluation to 100 utterances only.

Models

In order to minimize the time and the computational footprint of our experiments, we evaluate pretrained models directly provided by SpeechBrain [Ravanelli et al., 2021]. Some models output characters (31 output neurons), others output subwords with 5000 Byte Pair Encodings (BPE). All subwords (resp. character) models share their vocabulary and tokenizer. We do not use external language models for decoding. The detailed hyperparameters of all models are available in our source code.

We use two subword Seq2Seq encoder-decoder models (section 2.3.1). One uses Transformers and the other LSTMs. Each was trained with both the CTC Loss on the encoded states and Attention+Cross-Entropy Loss. When generating attacks we use either the Cross-Entropy Loss or the CTC loss, effectively treating each network as two different models sharing their latent encoded space.

We also evaluate DeepSpeech2 [Amodei et al., 2016], an LSTM CTC model with character outputs rather than subwords, in order to better analyze the impact of the network's output size on adversarial vulnerabilities.

Finally, we use multiple variations of Wav2Vec 2.0 [Baevski et al., 2020]: BASE models fine-tuned on 960h and 100h respectively, and the large model fine-tuned on 960h. For each of these models, in order to align the output space with our character vocabulary, the final projection was briefly fine-tuned on 100h for one epoch. This was sufficient to achieve the expected performance of these models on LibriSpeech.

Adversarial attacks

We use the untargeted Projected Gradient Descent (PGD) attack (section 2.1.3). We consider both the L_∞ and L_2 for the choice of a norm in Eq. 2.2. We control the perturbation size of the latter by fixing the Signal-Noise Ratio (SNR) of the attack as a hyperparameter. For each input, we set ϵ accordingly:

$$\epsilon = \frac{\|x\|_2}{10^{\frac{SNR}{20}}} \tag{5.1}$$

. For L_∞ PGD, we directly set ϵ as a hyperparameter.

We also apply the Carlini&Wagner (CW) attack (section 2.1.4) as a targeted attack. The attack objective (Equation 2.7) is optimized with Adam [Kingma and Ba, 2015]. The additional L_∞ constraint ϵ is first set large, then progressively decreased.

Attack methodology

We evaluate all attacks against all models whenever applicable. For PGD attacks we vary the hyperparameters controlling the amount of noise distortion; however, we keep all other hyperparameters constant. This includes the number of iterations in optimization attacks, which is respectively equal to 100 (PGD) and 5000 (CW). These values are greater or equal to those used by their original authors. Our targeted attacks use a set of three possible target sentences. For each input, we select the target whose length is the closest to the true label.

Metrics

To evaluate our attacks we rely on several metrics:

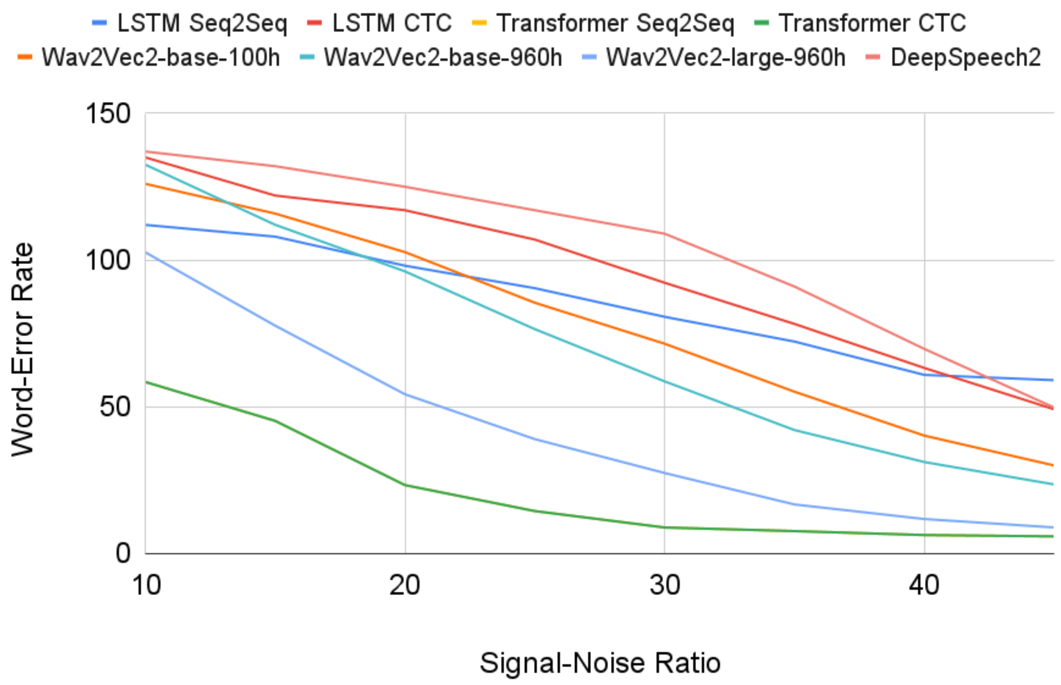
- For untargeted attacks, the **Word-Error Rate** (WER) of the model on the adversarial example with respect to the *true label.* The higher, the stronger the attack.
- For targeted attacks, the WER on the adversarial example with respect to the *target sentence*. The lower, the stronger the attack. We also report the **accuracy**, or success rate of these attacks, i.e. the fraction of inputs for which a successful adversarial example was found.
- The **Signal-Noise Ratio** (SNR) of the perturbation: the higher the SNR, the less perceptible the noise. 20dB can be considered a threshold for an "acceptable" noise level that does not disturb human comprehension; $30 - 40$dB is an estimate of "imperceptible" noise. Both these thresholds are approximate, as human perception does not align well with L_p norms.

5.2.2 Results and discussion

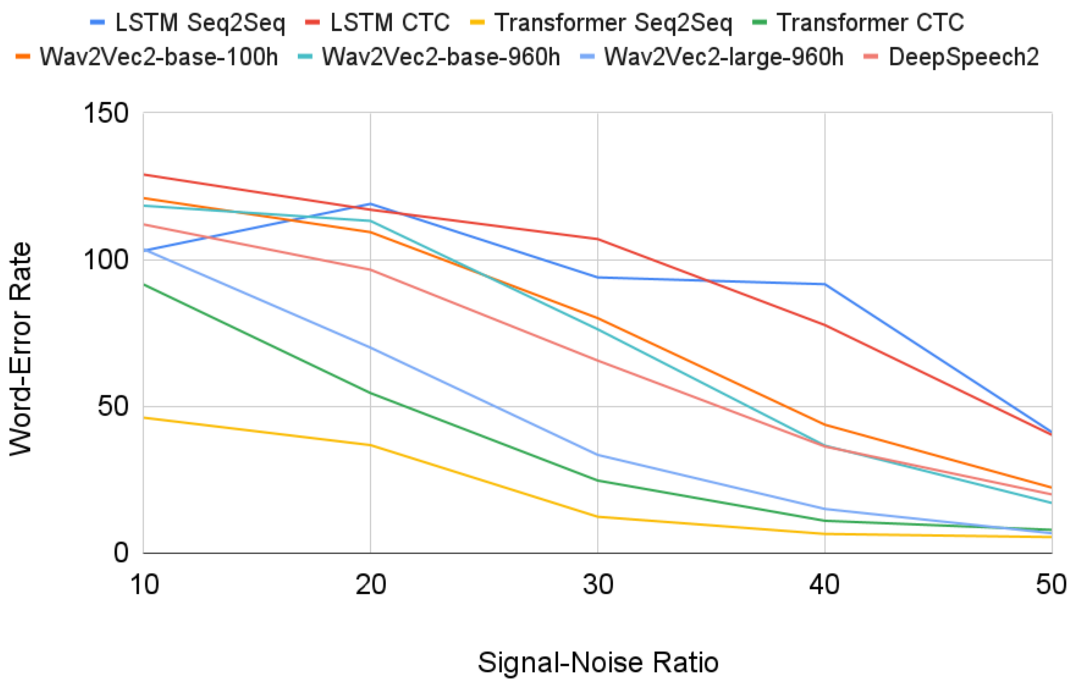
Model	Output type	Clean WER	Carlini&Wagner		
			WER↓	SNR (dB)	Accuracy
SpeechBrain LSTM Seq2Seq	subword	5.02	16.98	21	66%
SpeechBrain LSTM CTC	subword	6.15	4.63	27	91%
SpeechBrain Transformer Seq2Seq	subword	4.11	0	37	100%
SpeechBrain Transformer CTC	subword	5.88	0	38	100%
DeepSpeech2 (LSTM CTC)	character	9.44	0	40	100%
Wav2Vec2-BASE-100h (Transformer CTC)	character	6.27	0	35	100%
Wav2Vec2-BASE-960h (Transformer CTC)	character	3.53	0	35	100%
Wav2Vec2-LARGE-960h (Transformer CTC)	character	2.85	0	32	100%

Table 5.1: results of all models on clean data and under the (targeted) Carlini&Wagner attack. For CW We report the WER, SNR, and attack accuracy (i.e. success rate). Small WER values mean successful attacks.

We plot the results of the PGD attack in Figure 5.1, while the CW attack results are displayed in Table 5.1.



(a) WER as a function of SNR for the L_2-PGD attack



(b) WER as a function of SNR for the L_∞-PGD attack

Figure 5.1: Attacks results for the untargeted (a) L_2-PGD attack, and (b) L_∞-PGD attack, for different SNR values.

The analysis of the PGD plots shows large variations in the WER of different models under attack. The relative ranks of all models are consistent between L_∞ and L_2 attacks. The only exception is the Transformer CTC encoder, which compared to other models is more robust to L_∞ attacks than L_2 attacks. CW results on the other hand show a clear gap between the Transformer models and the LSTM models: the former always predicts the attack target on adversarial examples with low noise (SNR ≥ 30dB), while the latter require more noise for imperfect target transcriptions. The Imperceptible attack achieves almost identical results (not reported), with SNRs varying by up to 3dB.

Overall three trends emerge from these results:

- Encoders-decoder are *harder* to attack than encoder-only models. Seq2Seq models get lower WER than CTC models on PGD and lower SNR on CW attacks. The Transformer models model has a lower PGD WER than the Transformer encoder.
- Transformers are *harder* to attack than LSTMs with PGD, but *easier* with CW, as evidenced by the comparison of the SpeechBrain models, both Seq2Seq and CTC.
- Character LSTM models are *easier* to attack than subword models: the CW SNR and success rate are higher for DeepSpeech2 than the LSTM model. For transformers (SpeechBrain and Wav2Vec2) the difference is not perceptible

The higher robustness of encoder-decoders may simply be a result of their higher depth, which is a known cause of gradient obfuscation [Athalye et al., 2018a]. The fact that CW is effective against Transformer Seq2Seq models points towards that explanation: this attack's objective is especially powerful at circumventing optimization issues. The inconsistent results of PGD and CW against LSTMs and Transformers are harder to explain and certainly deserve more attention in future works. At the very least it shows that the performance boost of Transformers does not clearly carry over in terms of robustness. It also emphasizes the need to evaluate multiple models when crafting new attacks, and multiple attacks when building new models.

5.3 Effects of scaling on robustness

After evaluating models trained on the simple LibriSpeech dataset and with little real use, we move on to models that are actually deployed in numerous real-world applications. This section describes the adversarial evaluation of the Whisper model family [Radford et al., 2022], and our conclusions on the effect of data and model scaling on adversarial robustness.

5.3.1 Experimental setting

Models

Whisper exists in 5 model sizes. We run our attacks on all models from tiny (39M parameters) to large (1550M). All models are Transformer sequence-to-sequence models, with an encoder turning speech inputs into contextual representations, and a decoder mapping them to language tokens. There are four English-only models trained on 438kh of supervised English training data, and five multilingual models trained with an additional 243kh of multilingual data, for both transcription and translation. Initial tokens in the decoder can specify the task and the language. A language detection module generates the latter if it is not specified and if the model is multilingual. We do not change any of the default inference hyperparameters: for example, the beam size is 5 for all models.

For comparison, we also report attack results on the SpeechBrain Transformer model used in section 5.2, with the same beam size of 5 during inference. This beam size explains the (small) variation of results on that model between both sections.

Attacks

As in the previous section, we use the PGD and CW attacks. For PGD in L_2 norm, we set SNR objectives of 30 and 40dB, which correspond to very small noise. We use an attack learning rate of $0.1 * \epsilon$, and $n = 200$ attack iteration steps. This attack on Whisper medium runs in 2 minutes on one A100 for a typical utterance. For the L_∞ attack we fix $\epsilon = 0.005$ or $\epsilon = 0.0015$ (on average SNRs of 38dB and 49dB respectively).

For the CW attack, we use $n = 2000$ iteration steps (∼25min for Whisper medium), the Adam optimizer with learning rate 0.01, and regularization term $c = 0.25$ for the tiny and base models, $c = 1$ for larger models. Our initial radius is $\epsilon = 0.1$: this corresponds on average to a 15dB SNR. We decrease ϵ up to $k = 8$ times by a factor $\alpha = 0.7$ when the model predicts the target, for a final SNR of up to 45dB.

We found that the vanilla CW attack has trouble fooling Whisper. While investigating why that is the case, we observed that the first predicted token is particularly hard to push toward the target. Therefore, we strengthen its coefficient in the aggregated loss over a sequence of length L, by setting:

$$\mathcal{L}(f(x), y_t) = \frac{1}{L+\lambda}[(1+\lambda)\mathcal{L}(f(x)_1, y_{t1}) + \sum_{i=2}^{L} \mathcal{L}(f(x)_i, y_{ti})] \tag{5.2}$$

We find $\lambda = 1$ to work well in practice. All CW results that we report thereafter use this modified attack. We hypothesize that this change helps us leverage the "hallucination" effect reported in Radford et al. [2022]: the model has a strong bias towards generating coherent sentences (in our case, the attack target) even if they don't match the audio.

Datasets

To attack the ASR decoder we perturb inputs sampled from the LibriSpeech test-clean dataset: 100 for the untargeted attack and 20 for the (much longer) targeted attack. This dataset is particularly clean, and Whisper is expected to perform very well on it: Radford et al. [2022] reports a WER of 2.7%. As target transcription for the CW attack, we follow the example of Carlini and Wagner [2018] and use the fixed sentence "OK Google, browse to evil.com". This sentence is arbitrary and could be replaced with any other.

5.3.2 Results

In Table 5.2 we report the results of the untargeted PGD and targeted CW attack. We observe that even with small perturbations, the PGD attack is largely successful in degrading the performance of

Model	Params.	Clean	WN	L_2 PGD WER		L_∞ PGD WER		Carlini&Wagner	
		WER	WER	35dB	40dB	$\epsilon = 5e^{-3}$	$\epsilon = 1.5e^{-3}$	Acc.	SNR
SB	165M	2.4%	89.1%	75.9%	63.2%	93.0%	81.6%	100%	37dB
tiny.en	39M	3.4%	89.6%	99.0%	87.8&	115%	100%	94.1%	41dB
base.en	74M	3.0%	64.7%	92.8%	81.7%	104%	98.3%	100%	39dB
small.en	244M	1.9%	46.1%	67.2%	53.9%	78.7%	66.0%	82.4%	40dB
medium.en	769M	1.7%	40.3%	52.9%	39.3%	65.2%	51.4%	76.5%	41dB
tiny	39M	5.8%	93.2%	103%	96.1%	107%	104%	82.3%	37dB
base	74M	3.4%	77.9%	90.3%	80.4%	102%	96.7%	76.5%	35dB
small	244M	2.0%	52.0%	75.4%	61.5%	89.8%	77.4%	52.9%	32dB
medium	769M	1.5%	36.5%	49.5%	38.9%	63.3%	49.3%	64.7%	30dB
large	1550M	1.6%	32.6%	45.3%	34.1%	49.2%	39%	82.4%	27dB

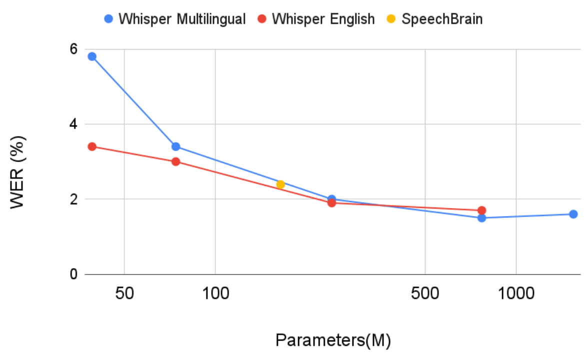
Table 5.2: Results on LibriSpeech test-clean of the white-box attacks. For untargeted PGD attacks, we report the achieved WER with a 35dB and 40dB respective SNR for L_2 attacks, and with $\epsilon = 0.005$ and $\epsilon = 0.0015$ for L_∞ attacks. We also report the Word-Error Rate on the same data under random white noise (WN) at 0dB. For the targeted CW attack, we report the proportion of successful adversarial examples (those that Whisper transcribes as the target) and their average SNR. We evaluate the Speechbrain transformer model (first line) and all Whisper models.

all Whisper models, by 35 to 89% in absolute WER for 40dB SNR and 48 to 99% for 35dB SNR. Such degradation can prevent Whisper from being useful in most practical applications (e.g. video captioning with a 45% WER is unsatisfying). L_∞ PGD attacks are equally or more successful, with very small radii (in practice similar to SNRs of 38-49dB).

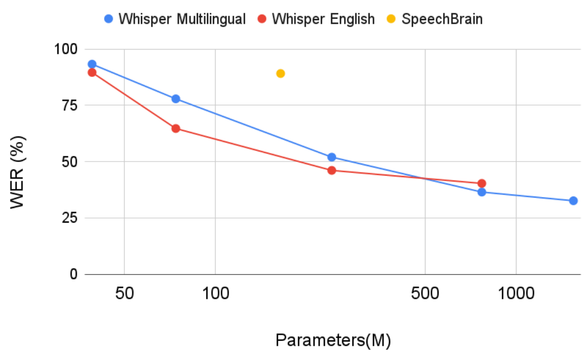
We observe that attack results on the SpeechBrain Transformer model are very close to those of the Whisper small model, of similar size and clean performance. We can conclude from this that the Whisper training method does not lead to any improvement in adversarial robustness on untargeted attacks, compared to models of similar architecture trained in-distribution on fewer data. In contrast, Whisper models perform considerably better than this baseline under white noise, which we report in the same Table. This illustrates the difference between robustness to *average* (random) and *worst-case* (adversarial) perturbations.

As for targeted attacks, in 50% to 90% of all sentences the CW attack succeeds in making the model predict the target with very little noise. This does show an increase in robustness compared to the baseline, but Whisper remains largely vulnerable to targeted attacks. Models of all sizes can be fooled, but multilingual models are harder to attack. Most models are fooled with an SNR above 35dB; for the bigger multilingual models the SNR drops but remains above 25dB. This could indicate that multilingual training helps make models robust to targeted attacks, though not robust enough to evade most attacks.

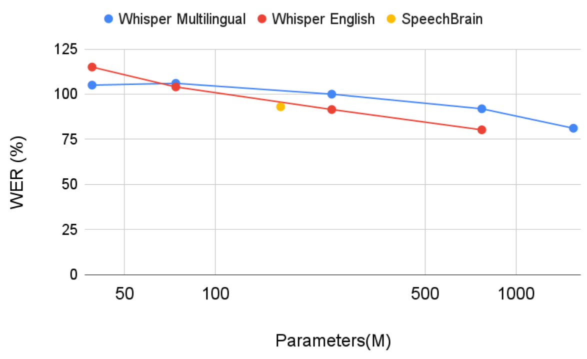
To help visualize the aforementioned results, we reproduce the results of Table 5.2 in Figure 5.2, by plotting the WER as a function of model size for all models, for clean data, noisy data, and L_2-PGD adversarial data. They emphasize that model size brings some robustness improvement, but at equal size, Whisper-style massive training is only beneficial against white noise, not adversarial noise.



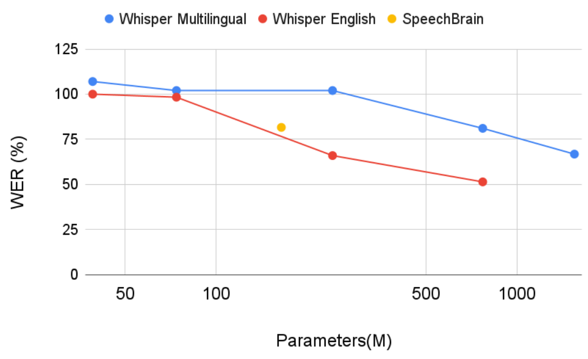
(a) Clean data



(b) Data corrupted with white noise (0dB)



(c) Data corrupted with L_2 adversarial noise (35dB)



(d) Data corrupted with L_2 adversarial noise (40dB)

Figure 5.2: WER of Whisper models and a baseline on inputs perturbed with different types of noise.

5.4 Effects of multilinguality on robustness

Another property of Whisper models is to handle sentences in multiple languages at once, by feeding the decoder a prompt containing a language identification token. This token is inferred if not provided. We show that an attacker can use this feature to their advantage, by fooling the model to detect the language.

5.4.1 Experimental setting

Language confusion attack

Whisper's language detector is essentially a classifier trained with cross-entropy loss. We use the PGD targeted attack to push the prediction from the original language to another target language. We evaluate how this affects subsequent ASR performance. We set an SNR of 45dB and $n = 30$ iteration steps.

Given that this is a simple attack objective, we also try applying a more restrictive threat model and run a *universal* attack. We optimize a single δ to fool not just one but all inputs x. Specifically, we train a 30-second long parameter δ to fool the following objective:

$$\max_{\|\delta\|_p < \epsilon} \mathbb{E}_{x \in \mathcal{D}} \mathcal{L}(f(x+\delta), y) \tag{5.3}$$

To optimize it we combine several utterances into a small "training set" $\mathcal{D}$, and train with PGD for several epochs. We then evaluate how that perturbation affects ASR performance on a test set disjoint from this training set. This universal attack uses an SNR of 40dB, and fits δ for 2000 epochs over the 70 training sentences using one iteration step per input and a learning rate of $0.001 * \epsilon$.

Dataset

To attack language detection, we use the multilingual CommonVoice dataset [Ardila et al., 2020]. We sample 100 sentences from the test set of each of the following seven languages: Armenian, Lithuanian, Czech, Danish, Indonesian, Italian, and English. Whisper was trained on varying amounts of these languages, making them representative of the distribution of its training data. We use three target languages: English, Tagalog, and Serbian, which are also present in very different amounts in the Whisper training set. For the universal language confusion attack, the attack training set consists of 70 sentences (10 per source language), disjoint from the seven test sets, and the target language is Serbian.

5.4.2 Results

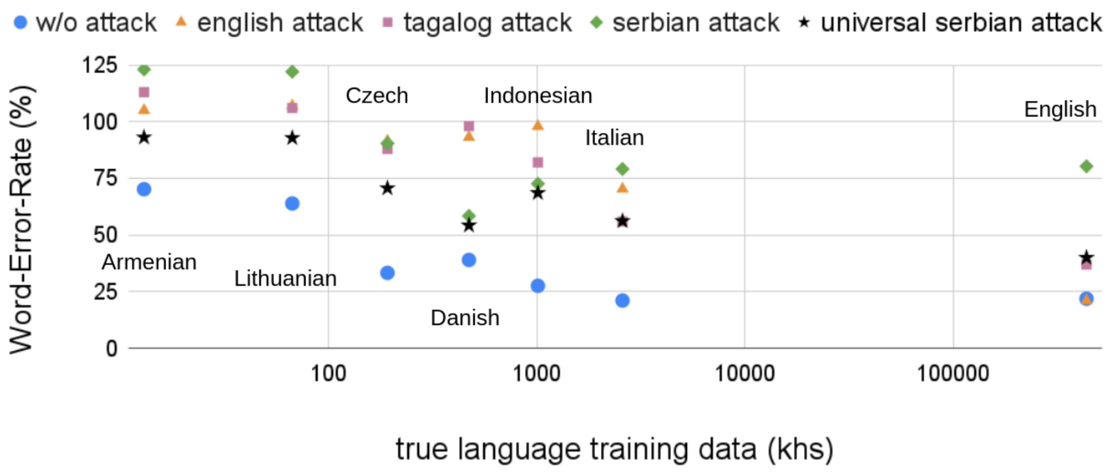


Figure 5.3: WER of the multilingual Whisper medium on subsets of the CommonVoice dataset in 7 languages, under language confusion attacks. We fool the model with white-box attacks into wrongfully detecting either English (yellow), Tagalog (pink), or Serbian (green). We also fool it with a universal attack to predict Serbian (black) The x-axis corresponds to the amount of training data Whisper was trained on for the *true* language of the inputs.

In Figure 5.3 we plot the WER of the Whisper medium model under the language confusion attack, as a function of the amount of Whisper training data in the true language. We report our results for three possible attack targets: English, Tagalog, and Serbian. We observe that in a large majority of cases, this very simple attack is sufficient to degrade significantly the WER of the ASR model. When the true language is poorly represented in the Whisper training data, as for Armenian and Lithuanian, the WER jumps over 100%. For higher-resource languages (Czech, Danish, Indonesian) the attack can degrade the WER to 75% or higher. For the best-represented languages, like English or Italian, the attack is a bit less effective but still degraded the performance by 16% to 60% absolute WER points. Intuitively, when the source and target languages are identical the attack has no effect, as observed with English.

We achieve this performance degradation with very low noise (45dB SNR) and an extremely simple attack with 30 iterations of PGD. This shows how brittle a multilingual model with language detection can be, especially over low-resource languages. While looking at the actual outputs on these low-resource languages under attack, we observed that the prediction is a mostly nonsensical mix of the true and target language.

Even the universal attack is largely successful in confusing Whisper, despite using a single perturbation for all inputs and all source languages. It achieves a Word-Error Rate degradation of 20 to 40% for all languages. This perturbation can be used off-the-shelf on any input, without additional computation. With its SNR of almost 40dB it remains almost imperceptible.

The influence of the choice of target language on attack success is hard to derive from our results. The Serbian attack target has a stronger effect on most source languages, but Danish and Indonesian are exceptions. Studying whether linguistic proximity between source and target languages would be an interesting follow-up to this work.

5.5 Applications of white-box attacks: the example of membership inference

5.5.1 Limitations and implications of adversarial vulnerabilities in ASR

The threat models we have evaluated in this work cannot be applied to 100% of ASR applications. Apart from the universal language confusion attack on Whisper, whose WER degradation does not match white-box attacks, our adversarial algorithms need time to generate noise tailored for each input. Moreover, they modify inputs in the digital space, not under real-world acoustic conditions. As a result, applying these attacks over the air as people speak rather than in the digital waveform space remains challenging. However, other works have extended adversarial attacks to be generated over the air [Qin et al., 2019, Schönherr et al., 2020] and in real-time [Lu et al., 2021, Xie et al., 2021]. Applications of those works to SpeechBrain, Wav2Vec2, and Whisper may be possible and would extend our results to many more threat models.

In addition, our simple threat models are sufficient to fool ASRs in several practical situations. For instance, if ASR is used to filter speech inputs, e.g. to remove hateful content from an online platform, new uploads can evade said detection with an untargeted attack. Other possible applications include *censorship triggering* (using targeted attacks to fool Whisper into perceiving hateful content where there is not) or even *data poisoning* if for instance Whisper is used to generate new text corpus from audio.

In this section, we propose an additional application of white-box ASR attacks in order to induce privacy leakage, by increasing the success rate of membership inference attacks. This section is based on results presented in Olivier et al. [2023b]. We do not provide all details of the methodology or results here - they are left to Appendix C.2 - and focus on the aspects most relevant to understanding how adversarial attacks can leak private information.

5.5.2 Membership inference pipeline

First presented by Shokri et al. [2017], membership inference (MI) is the problem of determining whether a given data record was used to train a machine learning model. MI is usually described as an *attack* on a model's training data, as it allows an external party (or *attacker*) to gain information about specific data records. Still, MI can be used benevolently, e.g., to check if a data record has been used in a model's training without the appropriate consent.

The spectrum of implementations and applications of MI is wide. Our MI pipeline is defined as follows: let M be an ASR model, trained with a dataset $\mathcal{D}_{\text{ASR}_{\text{train}}}$. Our goal is to determine which data records of a dataset $\mathcal{D}_{\text{MI}_{\text{test}}}$ are contained in $\mathcal{D}_{\text{ASR}_{\text{train}}}$ (i.e., were seen by the model during training). To this end, we first extract features. We apply a combination of feature extractors $f_1, ...f_J$, which are calculated on M's output for a given input x and a target transcription y. This results in the feature vector $F(x, y) = (f_1(x, y), f_2(x, y), ..., f_J(x, y))$.

Then, we build a MI binary classifier C, that takes $F(x, y)$ as input and outputs a binary label predicting whether x was in $\mathcal{D}_{\text{ASR}_{\text{train}}}$. By construction, in MI we typically do not assume *full knowledge* of the evaluated model's training setting, but *partial knowledge* such as its architecture but not its training data. Hence we train C using supervision from a shadow model $\bar{M}$, whose training dataset we assume is known to the attacker. We construct balanced train and test classification datasets consisting of utterance-transcription pairs either *positive*, i.e., in $\mathcal{D}_{\text{ASR}_{\text{train}}}$ or *negative*. Negative utterances are sampled from the same set of speakers as positive utterances to ensure that our attack is performed at the utterance level, and not at the speaker level. After training C on $\bar{M}$ with a training split $\mathcal{D}_{\text{MI}_{\text{train}}}$, we evaluate it on M with a test split $\mathcal{D}_{\text{MI}_{\text{test}}}$.

Many previous works like Shah et al. [2021c] have focused on *black-box* features for Membership Inference: they can be computed only with model outputs. In contrast, we mainly consider what we term *grey-box* features: they can be computed from the non-processed (i.e., non-decoded) output logits of the model. We consider that these features contain a higher amount of information on membership, than features computed from a post-processed output, as long as they are properly modeled [Carlini et al., 2022]. In this work we consider as loss-based features, the losses used to train a Seq2Seq ASR model: the attention loss (Att), which corresponds to the KL divergence between the output log-probabilities and the target transcription; and the CTC loss [Graves et al., 2006]. These features correspond to a gray-box model access setting.

Perturbed-input MI

In an effort to characterize the decision boundary around a given data point and potentially improve the MI decision, we also extend those features by perturbing the input signal, using Gaussian and Adversarial noise. We propose to explore the "worst-case" directions, in which the decision boundary is particularly close. We run a panel of PGD adversarial attacks in the L_∞ norm with different hyperparameters and compute features for all returned perturbations. We detail this procedure in Algorithm 2. To run PGD attacks, we require access not only to the loss but to the weights, so we consider adversarial features to assume *white-box* access to the target model.

We also perturb input data with random Gaussian noise (grey-box access). Gaussian perturbations are agnostic to the model and data, and let us evaluate the model's "average" behavior when getting further from the input in arbitrary directions. We use decreasing levels of Signal-to-Noise-Ratio (SNR), thus moving the perturbed signal away from the original input. For each SNR value, we select multiple random perturbations, whose MI results we summarize by their mean and standard deviation. This procedure is summarised in Algorithm 3.

Algorithm 2: Adversarial-based feature computation.

```
Require: Input $x$, set radii $\mathcal{E}$, #steps $N$, step size $\eta$, model $M(\cdot)$, target transcription $y$, feature extractor $F(\cdot)$
 1: feats ← [ ]
 2: for $\epsilon \in \mathcal{E}$ do
 3:     $\delta_\epsilon \sim \mathcal{U}(-\epsilon I, \epsilon I)$
 4:     for $n \leq N$ do
 5:         $g = \text{sign}\left(\frac{d}{d\delta_\epsilon} L(M(x+\delta_\epsilon), y)\right)$    ▷ Gradient
 6:         $\delta_\epsilon \leftarrow \text{clip}_\epsilon(\delta_\epsilon + \eta g)$    ▷ Optimization & projection
 7:     end for
 8:     feats[$\epsilon$] ← $F(M(x+\delta_\epsilon), y)$
 9: end for
10: return feats
```

Algorithm 3: Gaussian noise-based feature computation.

```
Require: Input $x$, set of SNRs $S$, #runs $N$, model $M(\cdot)$, target transcription $y$, feature extractor $F(\cdot)$
 1: feats ← [ ]
 2: for snr $\in S$ do
 3:     feats$_{\text{snr}}$ ← [ ]
 4:     for $n \leq N$ do
 5:         $\delta \sim \mathcal{N}(0, I)$    ▷ Sample Gaussian noise
 6:         $\delta_{\text{snr}} \leftarrow \sqrt{\frac{\|x\|_2^2}{\text{snr} \times \|\delta\|_2^2}} \times \delta$    ▷ Scale noise to SNR
 7:         feats$_{\text{snr}}[n] \leftarrow F(M(x+\delta_{\text{snr}}), y)$
 8:     end for
 9:     feats[snr] ← (mean(feats$_{\text{snr}}$), stddev(feats$_{\text{snr}}$))
10: end for
11: return feats
```

Type of Input	**Accuracy**	**AUC**
Natural	87.2 ± 0.39	92.7 ± 0.09
Gaussian	87.5 ± 0.23	93.8 ± 0.15
Adversarial	86.8 ± 0.26	93.1 ± 0.13
Nat. + Gauss.	89.4 ± 0.33	94.8 ± 0.14
Nat. + Adv.	89.1 ± 0.23	**95.1 ± 0.09**
Nat. + Gauss. + Adv.	**89.4 ± 0.36**	95.0 ± 0.16
Natural	87.4 ± 0.23	92.3 ± 0.17
Gaussian	85.9 ± 0.36	93.1 ± 0.19
Adversarial	85.9 ± 0.41	91.7 ± 0.14
Nat. + Gauss. + Adv.	**88.1 ± 0.17**	**94.2 ± 0.19**

Table 5.3: Results for experiments with full and partial knowledge of the training data. The features used are the attention and CTC loss, applied to natural and perturbed inputs. Above the line are full-knowledge results using the target model to train the MI classifier, and below are partial-knowledge results using a shadow model. For every experiment, we report the mean and standard deviation for different metrics, obtained by fitting the MI classifier 10 times with different random seeds.

5.5.3 Experiments

We run a number of MI experiments with and without shadow models, using the above features. All of our experiments are conducted on the LibriSpeech dataset, using subsets to train the evaluated and shadow model as well as the MI classifier. All of our models are Transformer AEDs. We report major results in Table 5.3. A fuller report of our experiments, as well as a complete experimental setting with all hyperparameters detailed, is provided in Appendix C.2.

The first 6 lines correspond to a full-knowledge setting where we train the MI classifier on the target model. The last 4 lines correspond to a partial knowledge scenario using a shadow model to train the MI classifier. In every case, both Attention and CTC loss are used as features.

In the full-knowledge setting, we can observe that Gaussian or adversarial inputs by themselves provide results that are comparable to the natural inputs. However, when combining natural inputs with one or the other we observe a significant increase in both accuracy and AUC (+2% in absolute value, or a 20% drop in error). Our best results are achieved when combining all features. This suggests that adversarial perturbations bring some additional features that the MI classifier can leverage. Partial knowledge results follow a similar trend, with the combination of natural and both perturbed inputs giving us the best overall results, allowing for an accuracy close to 88%, a 1% degradation over the full-knowledge attack. Our results confirm that adversarial perturbations are useful features that improve the performance of membership inference attacks.

5.6 Related work

5.6.1 Adversarial attacks on robust and multilingual ASR

The interactions between robustness to adversarial perturbations and other types of noise have not been explored for ASR so far. They however have been studied for image classification. Some works have unified adversarial and random noise under a *semi-random* regime and derived theoretical robustness bounds [Fawzi et al., 2016, Rice et al., 2021]. Moreover, a model robust to white noise can be indirectly useful in adversarially robust pipelines [Lecuyer et al., 2019, Cohen et al., 2019].

We also study specific attacks on multilingual ASR models. Other works have shown the fragility of multilingual language models under perturbations of text inputs [Rosenthal et al., 2021]. For ASR, some past work has compared the robustness of monolingual models trained in different languages [Markert et al., 2021].

Membership inference in speech recognition

Since it was first proposed, MI has been studied under different assumptions and domains, with a large predominance in the computer vision/natural language processing domains [Hu et al., 2022, Jayaraman et al., 2021, Choquette-Choo et al., 2021]. On the other hand, MI for ASR has been the focus of only a few studies. One of the first examples is the work of Miao et al. [2021], who extract speaker features and errors to train a shadow model to evaluate if a speaker's data was used during training. This was followed by the work of Shah et al. [2021c] who instead focus on the utterance level. While Miao et al. [2021] uses GloVE embeddings to compare the output and target transcriptions, and use these as MI features, Shah et al. [2021c] uses WER, and individual types of transcription errors, together with the ratio between the length of the output and target transcriptions. Differently, Tseng et al. [2022] focus on MI for self-supervised ASR models, where they leverage utterance- and speaker-level similarity scores. All these works only consider a strict black-box access to the model, potentially enriched by the model's confidence or the best k hypothesis. In this work, we go beyond the considered black-box settings to find an upper bound on what can be inferred if white-box access is possible.

Membership inference with perturbed inputs

An interesting subset of MI works, instead of only considering the model's output and loss for a data point x, also try to determine how the model's loss varies in its neighborhood. For instance, the authors in Jayaraman et al. [2021] suggest querying the target model multiple times on a sample x perturbed with different Gaussian noises. In a similar spirit, the algorithm presented in Choquette-Choo et al. [2021] queries for augmentations of x that likely have been used during training and the algorithm in Zhang et al. [2022] uses information on the record's adversarial robustness. In this work, we also leverage these ideas and propose two input perturbation-based approaches for MI attacks on ASR.

5.7 Discussion

Throughout this chapter, we have proposed a systematic evaluation of the adversarial robustness of ASR models and the effect on their robustness of a number of design choices. One common thread to our results is that the directions currently undertaken by the speech modeling community are not favorable to more secure ASR. Transformers are to a large extent more vulnerable than LSTMs. Massive training has little effect on adversarial robustness, and increasing model size has limited effect only. Multilingual training a la Whisper tends to raise additional vulnerabilities. Self-supervised models like Wav2Vec2 are not particularly more robust than fully supervised ones.

All of those results are conditioned on certain threat model assumptions, and in particular on the availability of model gradients. We have shown that those assumptions enable some significant threats already, with carryovers to data privacy on top of direct adversarial attacks. However, to get a complete picture of ASR robustness we must know which attacks remain possible when relaxing our threat model assumptions. Such *black box* attacks are the object of the next chapter.

Chapter 6

On the conditions of attack transferability on speech recognition

A recurring assumption of this thesis so far has been that a potential attacker has access to the parameters of their target model, and can therefore fool it by running gradient-based attacks. While we have pointed out in Chapter 5 that this assumption is often valid, and that this validity tends to increase over time, it remains common for ASR models to be released behind a commercial API (Speechmatics, Voci, Microsoft Azure, etc) or even embedded in a personal assistant device (Amazon Alexa, Google Home, etc). To fool those models, an attacker needs algorithms to run *black-box* attacks (section 2.4.2). Key methods to run such attacks include using black-box optimization, attacks without optimization at all, or using gradient optimization on a *proxy* model and leveraging the *transferability* of adversarial attacks (section 2.1.2), as illustrated in Figure 6.1a. Such attacks remain challenging when applied to most ASR models.

For instance, our experiments have shown that a number of black-box optimization and optimization-free attacks, effective on some outdated models, do not carry over well to recent model architectures and training methods. We report those experiments in Appendix D.1. Meanwhile, as we have mentioned in Chapter 2.4.2, previous works have shown that transferable attacks are extremely challenging on ASR models, at least on some architectures [Abdullah et al., 2021b, 2022]. Our objective in this chapter is to measure the extent of that difficulty on more recent architectures.

To our surprise, we find that a large number of recently proposed models are considerably more vulnerable to targeted transferable attacks than other models to transferable attacks. This chapter focuses on quantifying this vulnerability, explaining it, and discussing its implications.

6.1 Introduction

In this chapter, we study the robustness of modern transformer-based ASR architectures. We show that, in contrast with previously evaluated architectures [Abdullah et al., 2021b, 2022], many state-of-the-art ASR models are in fact vulnerable to the transferability property. Specifically, our core finding can be formulated as follows:

Pretraining transformer-based ASR models with Self-Supervised Learning (SSL) makes them vulnerable to transferable adversarial attacks.

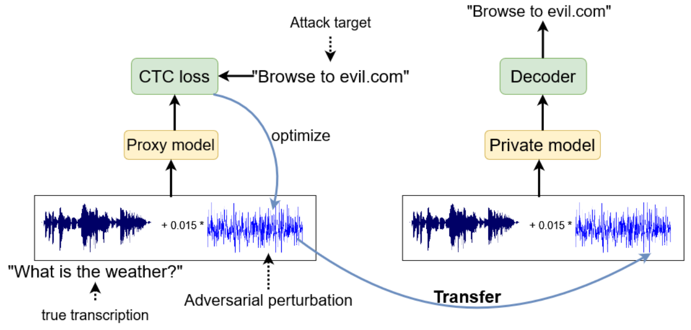
SSL is an increasingly popular learning paradigm in ASR (Figure 6.1b), used to boost model performance by leveraging large amounts of unlabeled data. We demonstrate that it hurdles robustness by making the following contributions:

- First, we motivate our study by conducting initial experiments on a variety of different architectures and transfer attacks between multiple pairs of models (section 6.2). We observe that SSL models are the only ones to display transferability, albeit limited.

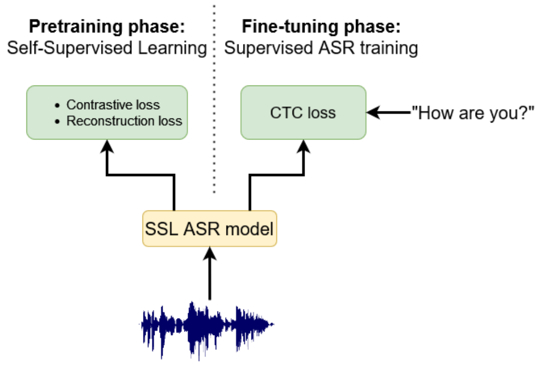
- Second, we push the limits of attack transferability by increasing the strength of our attack algorithm. We generate 85 adversarial samples by attacking two proxies at once (section 6.3). We show that these samples are effective against a wide panel of public transformer-based self-supervised ASRs (section 6.4). *This includes ASRs trained on different data than our proxies*. Our attacks can achieve a Signal-Noise Ratio of 30dB - not quite as imperceptible as the strongest white-box attacks [Qin et al., 2019], yet surprising for transferable attacks. On the other hand, models trained for ASR from scratch are much less affected. This suggests that SSL pretraining is the cause of the vulnerability to transferred attacks.

- Then, we provide more evidence that SSL-pretraining is the reason for this vulnerability to transferable attacks (section 6.5). We do so using an ablation study on Wav2Vec2-type models, either pretrained or trained from scratch and similar in all other aspects. We use each model as a proxy to generate adversarial examples, which we attack all other models with. We show that SSL pretraining in both proxy and private models indeed contributes to the transferability of adversarial attacks and that factors such as the amount of unlabeled data play an important role.

- None of the experiments above are sufficient to explain why SSL would make adversarial perturbations more transferable. In section 6.6, we propose an explanation for this curious phenomenon. We argue that targeted transferable ASR attacks require the proxy and private models to learn highly similar features; and that pretext tasks used in SSL training encourage such proximity between different model representations.



(a) Transferability



(b) SSL ASR

Figure 6.1: Diagrams illustrating (a) the transferability of an adversarial attack between a proxy and a private model, and (b) the training procedure of SSL ASR models

6.2 Motivation

In this section, we illustrate the challenges of transferable and targeted adversarial attacks on ASR.

6.2.1 Transferability between ASR models

Methodology

We use the models we analyzed in Chapter 5, and evaluate pairwise transferability. We select a *private* model (we wish to attack it but assume no access to its weights) and a *proxy* model (we have access to that model's weights). We run a CW attack on a proxy model and compute the WER of the target on the *private* model.

In practice, we use the adversarial examples we computed in (white-box results in Table 5.1). We use every possible model pair as proxy and private model.

Between the prediction and the attack target y_t, a low Word-Error Rate indicates a successful attack. Our metric to measure transferred attack success is therefore the word-level **targeted attack success rate**, defined as

$$\text{TASR} = \max(1 - \text{WE}R(f(x+\delta), y_t), 0) \tag{6.1}$$

Results

Model\Proxy	LSTM	LSTM CTC	Trans-former Seq2Seq	Trans-former CTC	Deep-Speech2	W2V2 BASE-100h	W2V2 BASE-960h	W2V2 LARGE-960h
LSTM S2S	*83.0*	*14.2*	0	0	0	0	0	0
LSTM CTC	*0*	*85.4*	0	0	0	0	0	0
Transformer S2S	0	0	*100*	*12.5*	0	0	0	0
Transformer CTC	0	0	*0*	*100*	0	0	0	0
DeepSpeech2	0	0	0	0	*100*	0	0	0
W2V2 BASE-100h	0	0	0	0	0	*100*	*38.9*	**26.6**
W2V2 BASE-960h	0	0	0	0	0	*51.9*	*100*	**37.4**
W2V2 LARGE-960h	0	0	0	0	0	7.1	8.6	*100*

Table 6.1: Word-level targeted success rate of the CW attack with multiple proxies and models of varying size and training data. Each row corresponds to a different model evaluated, and each column to a different proxy model. Italic results correspond to non-black-box attacks: the proxy and model are either the same model or share some weights or pretrained representation. Bold results correspond to successful attack transferability.

In Table 6.1 we report the TASR of each model pair. We observe that apart from the main diagonal (white-box attacks) almost all attacks have 0% success rate. This confirms the difficulty to achieve transferable attacks on ASR. Among the few exceptions are attacks between CTC and LAS models of identical architecture. However, in our experiments those models actually share weights: the CTC model is the encoder of the LAS model with an appended projection layer. Therefore those results also do not correspond to a full attack transferability scenario between different models. Besides, even in those cases, the success is very low ($< 15\%$).

However, results are different between Wav2Vec2 models. We first observe that perturbations transfer well between Wav2Vec2 BASE models using different finetuning data:, we achieve attack success rates of 38.9% and 51.9%. Those results also do not correspond to a full transferability scenario as the models were finetuned from the same base pretrained representation. However, the transferability is also important from the Wav2Vec2-LARGE proxy to the Wav2Vec2 BASE models (26.6% and 37.4%), while those models were trained entirely separately!

To our knowledge, this is the first case of targeted *and* transferable adversarial attack on ASR models. Intuitively, we could be tempted to attribute this peculiar attack transferability between Wav2Vec2 models to the fact that the proxy and target, have similar architectures, contrary to our other model pairs; or to the specificity of their architecture and training, such as their self-supervised pretraining method. Section 6.2.2 shows that the former factor is insufficient to explain transferability; the rest of this chapter investigates the latter.

6.2.2 Evaluating transferability under constant architecture

In order to achieve attack transferability, is it sufficient for the proxy and private model to share their architecture, training method, and training data? In this section, we show that it is not.

First, we refer to existing work reported in Abdullah et al. [2021b]. The authors evaluated transferability between models, using a similar attack method to ours in section 6.2. However, one key difference is that they only used versions of the DeepSpeech2 model that they trained themselves on LibriSpeech data, changing only the random seed at the beginning of training. Specifically, they trained 5 different DeepSpeech2 models, and run attacks between each pair of different models. The attack success rate was systematically 0 in each experiment. This shows that keeping the architecture constant is not sufficient to achieve attack transferability. The models in those experiments were closer in their hyperparameters than our Wav2Vec2 models: the BASE and LARGE models use different model widths, depths, and training iterations.

The fact that those results were achieved on DeepSpeech2 (an LSTM model) may suggest the confounding influence of model architecture. Maybe between two Transformer models of similar architecture and data we could observe transferability? To find that out we compute similar transferability experiments between Whisper models (section 5.3.1). All models have identical architectures apart from their size, and all English-only (resp. multilingual models) were trained on the same data.

In those experiments, we observed that on Whisper models as well, targeted attacks do not seem to transfer. More specifically, the attack TASR was consistently 0% between every distinct proxy/model pair. It seems that so far, only Wav2Vec2 models are subject to attack transferability. We hypothesize that this has to do with their specific training method and notably the use of self-supervised learning (SSL). In section 6.3 we explore the limits of that transferability by generating a strong adversarial dataset using SSL proxy models.

6.3 Experimental setting

In our core experiment, we fool multiple state-of-the-art SSL-pretrained ASR models with targeted and transferred adversarial attacks. We generate a small set of targeted audio adversarial examples using fixed proxy models. We then transfer those same examples to a large number of models available in the HuggingFace Transformers library. Table 6.2 specifies how much unlabeled and labeled data these models were trained on. We provide the full experimental details in Appendix D.2.

6.3.1 Generating adversarial examples on proxies

We describe our procedure to generate adversarial examples. To maximize the transferability success rate of our perturbations we improve the standard CW attack in section 2.1.4 in several key ways:

- To limit attack overfitting on our proxy, we combine the losses of *two* proxy models: Wav2Vec2 and HuBERT (LARGE). Both models were pretrained on the entire LV60k dataset and fine-tuned on 960h of LibriSpeech. As these models have respectively a contrastive and predictive objective, they are a representative sample of SSL-pretrained ASR models. The sum of their losses is used as the optimization objective in Equation 2.7.
- We use 10000 optimization steps, which is considerable (for comparison Carlini and Wagner [2018] use 4000) and can also lead to the adversarial noise overfitting the proxy models. To mitigate this effect we use a third model, the Data2Vec BASE network trained on LibriSpeech, as a stopping criterion for the attack. At each attack iteration, we feed our adversarial example to Data2Vec and keep track of the best-performing perturbation (in terms of WER). We return that best perturbation at the end of the attack.

  Because this procedure is computationally expensive, we only apply it to a subset A of 85 utterances of less than 7 seconds. We sample them randomly in the LibriSpeech test-clean set. We select attack targets at random: we sample a completely disjoint subset B of utterances in the LibriSpeech test-other set. To each utterance in A we assign as target the transcription of the sentence in B whose length is closest to its own. This ensures that a very long target isn't assigned to a very short utterance or vice versa.

6.3.2 Metrics

We keep using the word-level **targeted attack success rate** as our primary metric.

It is also interesting to look at the results of the attack in terms of *denial-of-service*, i.e. the attack's ability to stop the model from predicting the correct transcription y. Here a high WER indicates a successful attack. We define the word-level **untargeted attack success rate** as

$$\text{UASR} = \min(\text{WER}(f(x+\delta), y), 1) \tag{6.2}$$

We can also compute the attack success rate at the character level, i.e. using the **Character-Error-Rate** (CER) instead of the Word-Error-Rate. Character-level metrics are interesting when using weaker attacks that affect the model, but not enough to reduce the targeted WER significantly. We use them in our ablation study in section 6.5.

Finally, we control the amount of noise in our adversarial examples with the Signal-Noise Ratio (SNR). When generating adversarial examples we adjust the L_∞ bound ϵ (equation 2.7 to achieve a target SNR.

6.3.3 Models

Self-supervised learning

The principles of SSL-pretrained ASR models, whose robustness to attacks we evaluate in this work, are described in section 2.3.2. The models we focus on follow the neural architecture of Wav2Vec2 [Baevski et al., 2020]. Raw audio inputs are fed directly to a CNN. A Transformer encodes CNN outputs into contextualized representations, that a final feed-forward network projects in a character output space. The model is fine-tuned with CTC loss [Graves et al., 2006].

A number of different models follow this architecture, including Wav2Vec2, HuBERT [Hsu et al., 2021], Data2Vec [Baevski et al., 2022b], UniSpeech-SAT [Wang et al., 2021, Chen et al., 2022] or WavLM [Chen et al., 2021]. These networks only have very minor differences in their architectures. BASE models have 12 transformer hidden layers and 90M parameters. LARGE models have 24 layers and 300M parameters. Finally, XLARGE models have 48 layers for a total of 1B parameters.

While the networks are similar, the training pipelines of these models differ substantially. All models are pretrained on large amounts of unlabeled data, then fine-tuned for ASR on varying quantities of labeled data. The pretraining involves SSL objectives, such as Quantization and Contrastive Learning (Wav2Vec2), offline clustering and masked predictions (HuBERT), or masked prediction of contextualized labels (Data2Vec). Unispeech combines SSL and CTC pretraining with multitask learning. WavLM adds denoising objectives and scales to even greater amounts of unlabeled data.

Other baselines

We evaluate all SSL-pretrained models mentioned in section 6.3.3, along with several others for comparison: the massively multilingual speech recognizer or M-CTC [Lugosch et al., 2022] trained with pseudo-labeling, and models trained from scratch for ASR: the Speech-to-text model from Fairseq [Wang et al., 2020] and the CRDNN and Transformer from SpeechBrain [Ravanelli et al., 2021].

6.4 Results

We report the results of our adversarial examples in Table 6.2 for $\epsilon = 0.015$, corresponding to a Signal-Noise Ratio of $30dB$ on average. In Appendix D.5.1 we also report results for a larger ϵ value. Our adversarial examples have been released on the HuggingFace hub[1].

[1]https://huggingface.co/datasets/RaphaelOlivier/librispeech_asr_adversarial

Model	**Unlabeled data**	**Labeled data**	**Clean WER**	**Attack success rate (word level)**	
				targeted	untargeted
Wav2Vec2-LARGE	LV60k	LS960	2.0%	88.0%	100%
HuBERT-LARGE	LV60k	LS960	1.9%	87.2%	100%
Data2Vec-BASE	LS960	LS960	2.5%	63.4%	100%
Wav2Vec2-BASE	LS960	LS960	2.6%	**55.7%**	**100%**
Wav2Vec2-BASE	LS960	LS100	3.4%	**53.9%**	**100%**
Wav2Vec2-LARGE	LS960	LS960	2.3%	**50.7%**	**100%**
Data2Vec-LARGE	LS960	LS960	1.9%	**66%**	**100%**
HuBERT-XLARGE	LV60k	LS960	1.8%	**80.9%**	**100%**
UniSpeech-Sat-BASE	LS960	LS100	3.5%	**50.4%**	**100%**
WavLM-BASE	LV60k+VoxPopuli+GS	LS100	2.9%	**21.7%**	**100%**
Wav2Vec2-LARGE	LV60k+CV+SB+FSH	LS960	3.3%	**67.3%**	**100%**
Wav2Vec2-LARGE	LV60k+CV+SB+FSH	SB	6.3%	**41.5%**	**100%**
Wav2Vec2-LARGE	CV-multi	CV-multi	15.6%	**17.7%**	**100%**
Wav2Vec2-LARGE	CV-en	CV-en	7.69%	**19.7%**	**100%**
Wav2Vec2-LARGE	CV-fr	CV-fr	100%	0%	100%
M-CTC-Large	None	CV-en	21.7%	7.5%	76.4%
Speech2Text	None	LS960	3.5%	7.3%	63.3%
SB CRDNN	None	LS960	2.9%	5.9%	86.39%
SB Transformer	None	LS960	2.3%	6.49%	90.56%

Table 6.2: Results of the transferred attack on different ASR models ($SNR = 30dB$). The first three lines correspond to the proxies used to generate the adversarial examples. On all other models, the adversarial examples are transferred. We report for each model how much data was used for SSL pretraining and ASR finetuning. We also report its Word-Error-Rate on the LibriSpeech test-clean set, and the targeted and untargeted word-level attack success rate (see section 6.3.2)

On 12 out of 16 models, we observe that the attack achieves total denial-of-service: the untargeted success rate is 100%. Moreover, on the first 6 models (proxies aside), the targeted attack success rate ranges between 50% and 81%: the target is more than half correctly predicted! These results are in flagrant contradiction with past works on DeepSpeech2-like models, where even the slightest change in training leads to a total absence of targeted transferability between proxy and private model. Our private models vary from the proxies in depth, number of parameters, and even training methods, yet we observe important transferability. However, these 6 models have all been pretrained on LibriSpeech or Libri-Light with SSL pretraining, i.e. the same data distribution as our proxies.

The following five models were pretrained on different datasets. One was pretrained on a combination of Libri-Light, VoxPopuli, and GigaSpeech; two on Libri-Light, CommonVoice, SwitchBoard, and Fisher; and two on CommonVoice either multilingual or English-only. The transferability success rate on these five models ranges from 18% to 67%, which is significant. Even the CommonVoice models, whose training data has no intersection with Libri-Light, are partially affected.

Although our inputs and attack targets are in English, we apply them to a French-only CommonVoice Wav2vec2. This model, incapable of decoding clean LibriSpeech data, is also unaffected by our targeted perturbation. It therefore seems that, while multilingual models are not robust to our examples, a minimal performance on the original language is required to observe transferability.

The final 4 models for which the targeted transferability rate is null or close to null, are those that were not SSL-pretrained at all (including M-CTC which was pretrained with pseudo-labeling). These four models also partially resist the untargeted attack.

It emerges from these results that some recent ASR models, specifically those pretrained with SSL, can be vulnerable to transferred attacks. These results diverge significantly from previous works like Abdullah et al. [2021b, 2022] which showed no transferability between different models. Table 6.2 hints that SSL pretraining plays an important role in transferability, but does not prove it: to do so we would need to compare models of identical architecture and performance, pretrained and trained from scratch, both as proxy and target. This is what we do in the next section.

6.5 Ablation study

In this section, we conduct a thorough ablation study and establish rigorously that SSL pretraining makes ASR models vulnerable to transferred attacks. We also measure the influence of several other factors on transferability. This ablation study requires the generation of many sets of adversarial examples, using varying models as proxies, which would be computationally difficult with the improved attack introduced in section 6.3.1. Since we do not seek optimal performance, throughout this section we run the CW attack in section 2.1.4 with 1000 optimization steps.

6.5.1 Influence of self-supervised learning

In this section, we compare Wav2Vec2 models with varying amounts of pretraining data: 60k hours, 960h, or none at all. We use each model both as a proxy to generate adversarial noise and as a private model for evaluation with all other proxies.

As Wav2Vec2 models fine-tuned from scratch are not publicly available, we train our own models with no pretraining, using the Wav2Vec2 fine-tuning configurations on 960h of labeled data available in Fairseq [Ott et al., 2019]. These configurations are likely suboptimal and our models achieve test-clean WERs of 9.1% (LARGE) and 11.3% (BASE), much higher than the pretrained+fine-tuned Wav2Vec2 models. This performance discrepancy could affect the fairness of our comparison. Therefore, we add to our experiments Wav2Vec2 BASE models fine-tuned on 1h and 10h of labeled data only. These models achieve test-clean WERs of 24.5% and 11.1%. Therefore we can observe the influence of SSL pretraining by taking model architecture and performance out of the equation.

Our attacks are not as strong as in section 6.3.1, and only have limited effect on the targeted WER. Therefore we evaluate results at the character level, which offers much finer granularity. For reference, we observe that the CER between two random pairs of sentences in LibriSpeech is 80-85% on average. Therefore attack success rates higher than 20% (i.e. CER $< 80\%$ with the target) indicate a partially successful attack. We report those results in Table 6.3. Results in *italic* correspond to cases where attacked model is the proxy or was fine-tuned from the same pretrained representation and therefore do not correspond to a transferred attack.

Model\Proxy	**BASE LS960 960h**	**BASE LS960 10h**	**BASE LS960 1h**	**LARGE LS960 960h**	**LARGE LV60k 960h**	**BASE None 960h**	**LARGE None 960h**
BASE LS960 960h	*96.37%*	*64.11%*	*53.41%*	47.08%	44.7%	2.62%	2.53%
BASE LS960 10h	*42.64%*	*99.12%*	*72.91%*	43.14%	42.67%	2.65%	3.54%
BASE LS960 1h	*69.3%*	*81.12%*	*99.50%*	41.21%	36.68%	3.03%	3.04%
LARGE LS960 960h	44.61%	13.46%	8.32%	*67.03%*	37.34%	2.39%	2.54%
LARGE LV60k 960h	29.24%	5.68%	3.19%	25.19%	*97.13%*	2.59%	2.47%
BASE None 960h	7.84%	4.05%	3.83%	11.19%	7.16%	*99.57%*	19.05%
LARGE None 960h	8.12%	4.55%	3.46%	11.15%	7.52%	22.93%	*99.94%*

Table 6.3: Character-level targeted success rate of the attack with Wav2Vec2 proxies and models of varying size and training data. Each row corresponds to a different proxy and each column to a different private model. The format is [Model-size pretraining-data -finetuning-data]

These results show unambiguously that SSL pretraining plays a huge role in the transferability of adversarial attacks. Adversarial examples generated on the pretrained Wav2Vec2 models fine-tuned on 960h are partially successful on *all* pretrained models (success rate in the 25-46% range). They are however ineffective on the ASR models trained from scratch (4-8%). Similarly, models trained from scratch are bad proxies for pretrained models (2-3%) and even for each other (19-22%).

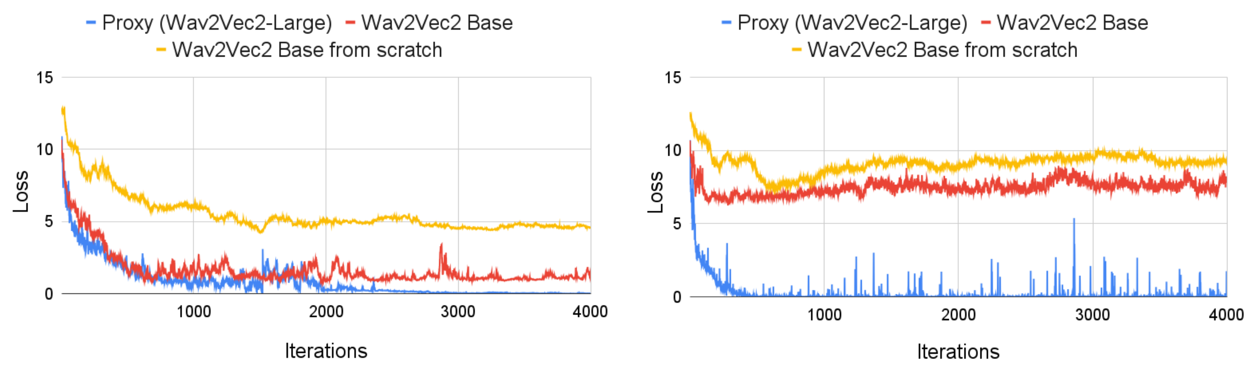
It follows that SSL pretraining is a necessary condition for transferable adversarial examples in both the proxy and the private model. We confirm it by plotting in Figure 6.2a the evolution of the target loss while generating one adversarial example. We display the loss for the proxy model (blue) and two private models. The loss of the pretrained private model (red) converges to a much lower value than the non-pretrained model (yellow).

SSL pretraining is however not a sufficient condition for attack transferability, and other factors play a role as well. For instance, the BASE model fine-tuned on just 10h and 1h are ineffective proxies: so strong ASR models are likely better proxies than weaker ones.

6.5.2 Influence of pretraining data

As observed in section 6.4 models that were (pre)trained on different data than the proxies can still be affected by transferred attacks. We analyze this effect in more detail in this section. We focus on five Wav2Vec2-LARGE models. One is pretrained and fine-tuned on LibriSpeech. One is pretrained on LibriLight and fine-tuned on LibriSpeech. Two are pretrained on LV60k, CommonVoice, SwitchBoard, and Fisher, and fine-tuned respectively on LibriSpeech and SwitchBoard. Finally, one is pretrained and finetuned on CommonVoice (English-only). As in the previous section, we evaluate every combination of proxy and target models.

We report the results in Table 6.4. We observe that most pairs of proxy and private models lead to important partial transferability. The major exception is the CommonVoice-only model, which does not succeed as a proxy for other models (0-8% success rate). In contrast, it is vulnerable



(a) With dropout (b) Without dropout

Figure 6.2: Evolution over attack steps of the loss on one adversarial input for three models: the Wav2Vec2 LARGE proxy and two targets, respectively with and without SSL pretraining. We run attacks (a) with dropout in the proxy model, and (b) without dropout in the proxy model.

to attacks transferred from other models, including those that do not have CommonVoice in their training data. We also note that models pretrained on Lbri-Light or more (60+khs) are better proxies, and more vulnerable to attacks, than the LibriSpeech-only and CommonVoice-only model. In other words, the vulnerability that we point out is worsened rather than mitigated by increasing amounts of available data.

Model \Proxy	**LS960 LS960**	**LV LS960**	**LV-CV-SB-FSH LS960**	**LV-CV-SB-FSH SB300**	**CV CV**
LS960 LS960	*64.3%*	41.9%	44.3%	39.8%	8.3%
LV LS960	26.3%	*96.1%*	79.8%	55.6%	2.9%
LV-CV-SB-FSH LS960	32.4%	83.2%	*93.6%*	*69.3%*	5.5%
LV-CV-SB-FSH SB300	19.8%	47.9%	*81.1%*	*88.5%*	0%
CV CV	26.11%	58.34%	48.0%	43.1%	*98.1%*

Table 6.4: Character-level success rate of the attack with different proxies and models. Each row corresponds to a different proxy and each column to a different private model. The format is [pretraining-data fine-tuning-data]. All models follow the Wav2Vec2-LARGE architecture.

6.5.3 Model size and training hyperparameters

We now extend our ablation study to models pretrained with different SSL paradigms. We report the results in Table 6.5. We observe that adversarial examples also transfer between models trained with different paradigms. Moreover, at equal pretraining data, all models are not equal proxies, and the HuBERT LARGE model (pretrained on 60kh) is the best proxy by a large margin.

Model \Proxy	**W2V2 BASE LS960**	**W2V2 LARGE LS960**	**W2V2 LARGE LV60**	**D2V BASE LS960**	**D2V LARGE LS960**	**HB LARGE LV60**	**HB XLARGE LV60**
W2V2 BASE LS960	*96.37%*	47.08%	44.7%	20.61%	24.9%	55.55%	47.46%
W2V2 LARGE LS960	44.61%	*67.07%*	37.34%	16.8%	20.89	56.87	42.21
W2V2 LARGE LV60	29.24	25.19	*97.13*	13.73	16.05%	71.78%	46.61%
D2V BASE LS960	37.9%	34.49%	47.15%	*98.44%*	24.2%	58.75%	46.71%
D2V LARGE LS960	28.72%	28.27%	47.75%	25.03%	*94.53%*	68.97%	51.02%
HB LARGE LV60	23.83%	27.19%	49.27%	14.92%	30.08%	*97%*	56.83%
HB XLARGE LV60	26.55%	33.31%	51.68%	17.53%	30.5%	83.92%	*87.66%*

Table 6.5: Character-level success rate of the attack with different proxies and models. Each row corresponds to a different proxy and each column to a different private model. The format is [Model-type Model-size pretraining-data] where model types are Wav2Vec2 (W2V2), Data2Vec (D2V) and HuBERT (HB). Each model was fine-tuned on 960h of LibriSpeech training data.

6.6 A hypothesis on the cause of and obstacles to transferability

We have established a link between adversarial transferability and the SSL pretraining of ASR models. In this section, we propose a hypothesis explaining that link. We first show in section 6.6.1, with empirical justification, that attacks with a very precise target are much harder to transfer everything else being equal, explaining why targeted ASR attacks are usually non-transferable. Then in section 6.6.2 we suggest ways in which SSL alleviates these difficulties, thus recovering some transferability.

6.6.1 At equal white-box success, very targeted attacks do not transfer

Targeted attacks on CIFAR10 force the model to predict one out of 10 different labels. Targeted attacks on ASR models force the model to transcribe one of all the possible transcriptions: With sequences of just five English words, the number of possibilities is equal to $170000^5 \sim 10^{26}$. We can call such an attack "very targeted", by contrast to more "mildly targeted" attacks on CIFAR10.

We hypothesize that the target precision, or "how targeted" the attack is, negatively affects its transferability success rate, explaining why targeted ASR attacks do not transfer easily. To demonstrate it empirically, we can imagine an experiment where an attacker tries to run a very targeted attack on CIFAR10. We hypothesize that in such a case, the transferred attack success rate would drop *even if the white box attack success rate remains high*. Inversely, if we designed a "mildly targeted" attack on ASR models, we would expect it to achieve a non-trivial transferability success rate. We designed experiments for both cases, which we summarize below. Complete experimental details and results are provided in Appendix D.3.

Very targeted attacks on CIFAR10

We run an attack on a ResNet CIFAR10 model. We do not just enforce the model's most probable output (top1 prediction) but the first k most probable outputs (top$k$ prediction). For example with $k = 3$, given an image of an airplane, the attack objective could be to modify the image such that the most probable model output is "car", the second most probable is "bird" and the third is "frog". Our attack algorithm sets a "target distribution" of classes, then minimizes the KL divergence of the model's probabilistic outputs and the target, using Projected Gradient Descent. The success rate is evaluated by matching the top k predictions and the top k targets.

We compute the L_∞ attack success rate ($\epsilon = 0.03$) for both white-box and transferred attacks as a function of the "target precision" k. For $k = 1$, we measure a transferability success rate above 30%. However, as k increases, the transferability success rate drops close to 10%, which is the success threshold that a random model would achieve. In other words, the transferability becomes null as k increases. Meanwhile, the white box attack success rate remains above 95%. Therefore very targeted attacks on images do not transfer.

Mildly targeted attacks on ASR

We train five small Conformer models on LibriSpeech. On each of them, we generate targeted adversarial examples. The target objective is simply to prepend the word "But" to the original transcription. This makes for a much less targeted attack as is traditionally done with ASR. The attack success rate is evaluated simply by checking the presence of the word "But" at the beginning of the prediction. We restrict evaluation to inputs whose transcription does not start with that word.

For each model, we generate 100 adversarial examples and evaluate them on all 4 other models. We thus obtain 20 different transferability success rates. The average of these scores is **18%** with a standard deviation of **4.7%**. Therefore mildly targeted attacks on ASR transfer substantially better than regular, very targeted attacks. Equivalent experiments with very targeted ASR attacks are reported in Abdullah et al. [2021b]: the word-level transferability success rate is 0%.

6.6.2 Very targeted transferability requires important feature overlap

Why would very targeted attacks transfer less? As Ilyas et al. [2019] show, statistically meaningful patterns in the training data may be "robust" (i.e. resilient to small perturbations) or non-robust. By leveraging non-robust features attackers can generate adversarial perturbations - and as these features can be learned by any models, these perturbations will transfer. The underlying assumption behind this framework is that all models learn the *same* features. In practice, two separate models do not learn identical features due to randomness in training. But if they are "close enough", i.e. if the *feature overlap* between both models is important, then transferability will be observed.

Therefore, it makes perfect sense that more targeted attacks would transfer less. The more precise and difficult the attack objective is, the more features the attacker will depend on to achieve it. This increases the amount of feature overlap needed between the proxy and private model for the attack to transfer. In the case of targeted ASR attacks, the required overlap is considerable. We hypothesize that SSL pretraining increases the feature overlap between ASR models. As empirically verifying it would pose important difficulties, we propose a high-level justification of that hypothesis.

ASR training aims at learning a representation that enables speech transcription. A subset of all features is sufficient to achieve this objective: for instance, there are lots of redundancies between low-frequency and high-frequency features, and a human listener can easily transcribe speech where most frequencies have been filtered out. The set of features learned by ASR models is therefore underspecified: two models even very similar identically may learn representations with little overlap.

Self-Supervised Learning on the other hand does not only learn useful features for transcription but features needed for predicting *the input itself*: parts of the input are masked, then they (or their quantized or clusterized form) are predicted using context. Arguably this much more ambitious objective requires the network to learn as many features as possible. In fact, the goal of such pretraining is to learn useful representations not just for ASR but any downstream task - i.e. "exhaustive" representations. Intuitively, different models trained in that way would share many more features than ASR models trained from scratch - leading to more transferable adversarial examples.

6.7 Related work

The transferability of adversarial attacks has been known for many years in Image Classification [Papernot et al., 2016b]. On ASR it has been limited to simple attack objectives, like preventing WakeWord detection in Alexa [Li et al., 2019] or signal processing-based attacks [Abdullah et al., 2021a, 2023]. When it comes to optimization-based attacks on large ASR models, transferability claims are usually limited and focus on untargeted attacks [Wu et al., 2022]. In very specific cases there have been limited claims of targeted, transferable attacks, such as Yuan et al. [2018]; however, this work does not focus on imperceptible attacks with small amounts of noise, but rather attacks embedded in music. When it comes to standard targeted optimization attacks, Abdullah et al. [2021b] have shown that they display no transferability on DeepSpeech2 models, even when the proxy and the attacked model are trained with identical hyperparameters apart from the initial random seed.

Past ASR adversarial attacks usually focus on a handful of neural architectures, typically DeepSpeech2 [Amodei et al., 2016], sometimes Listen Attend and Spell [Chan et al., 2016]. Only recently have attacks been extended to multiple recent architectures for a fair comparison between models [Lu et al., 2021, Olivier and Raj, 2022, Wu et al., 2022]. Most related to this work is Wu et al. [2022], which focuses on the vulnerability of SSL speech models. They however focus on attacking the base pretrained model with untargeted noise that remains effective on downstream tasks. We study targeted attacks, with a much deeper focus on transferability between different models. Olivier and Raj [2022] have hinted that Wav2Vec2 models are vulnerable to transferred attacks, but only report limited results on two models and do not investigate the cause of that phenomenon. We attribute it to SSL pretraining and back our claims empirically.

Abdullah et al. [2022] have identified factors that hinder transferability for ASR attacks, such as MFCC features, Recurrent Neural Networks, and large output sizes. Since Wav2Vec2 is a CNN-Transformer model with character outputs: this gives it a better prior than DeepSpeech2 to achieve transferable adversarial attacks. However, according to that paper, this should be far from sufficient to obtain transferable attacks: our results differ in the case of SSL-pretrained models.

6.8 Discussion

In this chapter, we have shown that SSL, a line of work gathering attention in the ASR community, is a source of vulnerability. Direct access to the weights of SSL-pretrained models is no longer required to fool them to predict outputs of the attacker's choice. To an extent, knowledge of their training data is not required either.

As it is likely that SSL will be used to train ASR systems in production, and given the existence of over-the-air attack algorithms, our results pave the way for practical, targeted attacks in the real world. By no means do these results imply that this line of work should be aborted, but they emphasize the pressing need to focus on robustness alongside performance. This is the focus of the next part.

Part III

Adversarially robust speech recognition

Chapter 7

Randomized smoothing for speech and audio

So far we have studied the question: *Do we need adversarially robust ASR models*. We have proposed analysis tools (like angular metrics in part I) and elements of answer (by identifying threats in part II). We however have ignored the question of the *possibility* for robust ASR models. Designing defense mechanisms for Speech tasks, and Speech Recognition in particular, is our focus in this part.

In this chapter, we study Randomized Smoothing (section 2.2.2), a very promising defense due to its versatility and simplicity. Where adversarial training, is time and resource-consuming even for image recognition, and Relaxed or exact provable defenses are heavily architecture-dependent, the Randomized Smoothing paradigm is at its core model-agnostic and time-efficient. By adding Gaussian noise to the inputs, we obtain a random but smooth and certifiably robust function. Of course, there is no free lunch, and a toll is paid in model performance: by applying the defense we shift the robustness challenge from small but worst-case perturbations to random but much larger input distortions.

Fortunately, the Speech Processing research community has been investigating robustness to noise for much longer than adversarial perturbations have been around. Simultaneously applying Randomized Smoothing for robustness and leveraging audio-specific methods to mitigate the performance tradeoff seems like a very promising approach. We explore such combinations in the current work.

7.1 Introduction

To use randomized smoothing on ASR while retaining good clean performance (section 7.2, we consider speech enhancement methods to make the defended model more accurate on Gaussian-augmented inputs. We also replace the majority vote with a strategy based on the "Recognizer Output Voting Error Reduction" (ROVER) [Haihua et al., 2009] method. Depending on whether we apply training data augmentation, we provide both an off-the-shelf defense and one that requires specific fine-tuning.

We first apply our defenses to a DeepSpeech2 and a Transformer model, trained and evaluated on the LibriSpeech dataset (section 7.3). We test them against strong attacks like the CW attack

[Carlini and Wagner, 2018] and the (untargeted) PGD attack [Madry et al., 2018]. We run adaptive versions of these attacks to avoid obfuscation effects. Our best model shows strong robustness against these attacks: to achieve partial transcription of the target sentence, attack algorithms require 10 times larger perturbations. Under equal noise distortions, the Word-Error-Rate (WER) on the ground truth under denial-of-service attacks improves by 30 to 50% for our model compared to the baseline. In section 7.4, we extend our analysis to more advanced ASR models like Whisper and Wav2Vec2, as we did in Chapter 5. We show that contrary to undefended ASR, under randomized smoothing clean performance improvements do carryover in terms of robustness.

Finally, in section 7.5 we use speech processing for a slightly different purpose. We investigate a potential limit of Randomized smoothing: its agnosticism to the particular vulnerabilities of the model it is applied to. Given information about the distribution of adversarial perturbations, we may wish to alter the noise distribution applied to the inputs accordingly, with digital filters. We investigate high-pass filtering of the Gaussian noise and find it to be very effective in specific contexts, such as speaker classification.

7.2 Randomized smoothing for speech inputs

The variable length of speech inputs is not an issue to use randomized smoothing. The main consequence is that the L_2 norm of a perturbation scales with the utterance length. Since Signal-Noise Ratio is normalized by utterance length, this does not significantly affect our experiments. Regardless of the nature of the output space, the reasoning that Gaussian distributions centered on an utterance x and an adversarially perturbed one $x + \epsilon$ are close remains valid. This tends to show that the overall noise-additive method still makes natural and adversarial points similar from the model's perspective.

7.2.1 Gaussian noise-robust speech models

As mentioned above, when using randomized smoothing it is critical to retain good performance on Gaussian-augmented inputs. This is not a trivial objective, especially with ASR models. We consider the following techniques, applicable for most tasks using speech inputs.

Augmented Training Cohen et al. [2019] train image classification models with Gaussian augmentation. This approach is applicable to speech tasks as well. However, when it comes to neural ASR models we find that training them entirely on Gaussian-augmented data leads to difficult convergence issues. Instead, we used a pretrained model on clean data that we fine-tune with Gaussian augmentation for 3-5 epochs. We find that it helps training converge and leads to similar or better results on noisy data in a much shorter time.

Speech enhancement Speech enhancement algorithms help improve speech quality. After adding Gaussian noise, we can use speech enhancement to restore the original audio quality. We tried multiple standard enhancement methods and found a priori SNR estimation (ASNR) [Scalart and Filho, 1996] to be the most effective.

Salman et al. [2020b] proposed *denoised smoothing* for image classification, using a custom-trained network as denoiser. The ASR equivalent would be to use neural enhancement methods such as SEGAN [Pascual et al., 2017]. In our experiments, it does not reach the same performance (in terms of WER in the end-to-end ASR pipeline), most likely because state-of-the-art speech enhancement models are tailored to real-world-like noise conditions and are not trained on Gaussian noise. It is possible, though not certain, that a generative model trained on Gaussian noise would improve the enhancement results: Pascual et al. [2017] argue that they outperform first-order filters like ASNR specifically for complex noise conditions.

7.2.2 Voting strategies on text outputs

Even with augmented training and/or enhancement, when feeding noisy inputs to ASR models the output distribution has high variance. Running multiple forward passes and "averaging" the outputs can help reduce that variance and improve accuracy. But this requires a good voting strategy on text outputs.

We first discuss some elementary strategies and their potential drawbacks, then describe the ROVER-based vote that we use. All of these strategies are empirically compared in section 7.3.2. We denote the sampled transcriptions as $t_1, ..., t_n$, and t is our final transcription.

One-sentence estimation A solution that has the merit of simplicity is to not vote at all. Using only one input, we can hope that the sentence we get is "close" to the most probable sentence (in terms of Word-Error Rate for instance) and just return it as our output. This is the strategy used by Żelasko et al. [2021].

Majority vote Following Cohen et al. [2019] for classification we can vote at sentence level: $t = argmax_{t' \in \{t_1,...,t_n\}} card(\{i/t_i = t'\})$

Designed for classification, majority vote is not adapted to probabilistic text outputs. The set T of all possible transcriptions is infinite, and even with our most stable models and a relative noise of -15 dB, 100 noise samples typically output 100 different transcriptions. Even without setting up rigorous statistical tests, it is clear that outputting the likeliest transcription, or just a "likelier than average" one, with high probability would require thousands of ASR iterations, which in practice is not feasible. In other words, majority vote is barely better than one-sentence estimation, for a high computation cost.

Statistics in the logits space For a given input utterance length, some ASR architectures, such as CTC-trained models, first generate fixed-length logits sequences $l_1, ..., l_n$, then apply a best-path decoder d to generate transcriptions $t_i = d(l_i)$. It is then possible to aggregate these logits over the random inputs, for example by averaging them, then to apply the decoder: $t = d(\frac{1}{n} \sum_i l_i)$.

One potential issue with this strategy as a defense is that it distances itself from the randomized smoothing framework, where the use of *discrete* outputs to vote on is critical. To get a concrete idea of how this could be a problem, one should remember that adversarial examples can be generated with *high confidence* (aka very large logits). Such a phenomenon could disrupt the statistic by over-weighting the fraction of inputs that are most affected by an adversarial perturbation.

ROVER The Recognizer Output Voting Error Reduction system was introduced by NIST in Haihua et al. [2009], as an ensembling method that mitigates the different errors produced by multiple ASR systems. Contrary to majority vote it works at the word level rather than the sentence level, by selecting at each position the word present in most sentences. ROVER should be fed the time duration of each word in the audio space, which we can extract using audio-text alignment information provided by the ASR models (see section 7.3.1). ROVER works in two steps. First, it aligns all k sentences word-by-word and aggregates them into one Word Transition Network (WTN), i.e. a graph where nodes represent timesteps, and edges between two timesteps are word (or silent) candidates. Alignment is done iteratively: the first sentence serves as a base WTN, then for $i = 2, ..., k$ ROVER merges sentence i with the base WTN using Dynamic Programming tool SCLITE, using a process close to Levenstein distance: it finds the minimal cost alignment using operations of substitution, insertion, and deletion. These alignment steps make use of audio alignment information as well as word and sentence scores, to output a final WTN. At this step, ROVER votes on the aligned words using (in our version) the frequency of each word. It also accepts metrics based on word confidence.

In our work, we introduce an alternative use of ROVER, as a voting system on the text outputs of the same probabilistic model rather than for ensembling multiple models. We mostly use it as a black box, by feeding to ROVER multiple output sequences. When evaluating its confidence-based variation (using DeepSpeech2's softmax outputs as confidence scores) we found not to bring any improvement in our use case, therefore we report results using the word frequency-guided algorithm. The main drawbacks of this method lie in the time penalty of the voting module when using a large number of inputs. We further discuss that limitation and our choices for the number of random inputs in appendix E.

7.3 Fully-supervised Speech Recognition Experiments

7.3.1 Setup

Dataset We run all our experiments on the 960 hours LibriSpeech dataset [Panayotov et al., 2015], and report our results on its test-clean split. As adversarial attacks can take a considerable amount of time to compute, we evaluate attacks on the first 100 utterances of this test set.

Models We test our methods on several model architectures (see section 2.3.3 for details)

- The DeepSpeech2 model [Amodei et al., 2016] pretrained on the clean LibriSpeech training set. We fine-tuned it on Gaussian-augmented data for one epoch, using always the same deviation used at inference for smoothing. For decoding, we use greedy search, as we find that increasing the beam size has very little impact on WER for this model. This is a relatively lightweight model that we use for ablation experiments. The CTC decoder provides frame alignments for each transcription character: we use them to infer word duration (needed for ROVER) with good precision.

- An end-to-end Transformer architecture. Training and hyperparameter search for transformer models can be computationally expensive, so we use the off-the-shelf SpeechBrain Transformer [Ravanelli et al., 2021] trained on LibriSpeech and do not fine-tune this model on Gaussian noise. This transformer implementation does not output character alignment.

It however provides word-level attention scores with the encoded audio. We can align each word with the highest-scoring audio vector, and obtain a word-level alignment.

Attacks We evaluate our defenses against white-box attacks:

- The untargeted PGD attack (section 2.1.3). Rather than fixing a value for its perturbation budget ϵ over all sentences, it is more interesting to bound the relative amount of noise compared to the input, that is the **signal-noise-ratio** (SNR) expressed in decibels:

$$\text{SNR}(\delta, x) = 20 * \log_{10}(\frac{\|x\|_2}{\|\delta\|_2}) \tag{7.1}$$

  When running PGD attacks, we set an SNR threshold, then derive for each utterance the L_2 bound $\epsilon = \frac{\|x\|_2}{10^{\frac{SNR}{20}}}$.

- The targeted and unbounded CW attack (2.4) against the DeepSpeech2 model. We fixed 3 target sentences of different lengths, constant in all our experiments. We try to perturb each input utterance until the model generates one of the targets (the one of closest length). For example, all utterances of less than 3-8 words are perturbed to predict the target *"Really short test string"*.

Our defenses use randomized (Gaussian smoothing) and non-differentiable (speech enhancement) pre-processing steps. We follow the recommendations of Athalye et al. [2018a] and adapt our attacks to alleviate these effects, using the *Straight-through estimator* for non-differentiable modules and *Expectation over Transformation* for randomness: we average the gradients returned by 16 backpropagation steps.

We illustrate the need for such attacks in additional results in appendix E.1.

Defenses Our models are defended with Gaussian noise, ASNR enhancement, voting strategies, or a combination of all of these. The noise deviation is set to $\sigma = 0.01$ or $\sigma = 0.02$ depending on the experiments, which corresponds for the vast majority of utterances to a signal-noise ratio in the 10-14dB and 7-11dB respectively.

Against adversarial examples, we compare our models with each other, as well as with the undefended DeepSpeech2 model and a baseline defense using MP3 Compression [Carlini and Wagner, 2018].

Evaluation metrics We evaluate our models under untargeted attacks with **Word Error Rate** (WER) on the ground truth: the lower the WER the better the defense. With targeted attacks, we also report WER both on the ground truth and the attack target. More importantly, since the CW attack is unbounded our primary evaluation metric is the **Signal-Noise-Ratio** (SNR).

7.3.2 Results

We first show that our defended models, which add Gaussian noise to all inputs, retain low WER on this noisy but non-adversarial data. Then we report their performance against adversarial attacks. We show that they successfully recover all attacks that use near-imperceptible noise.

	Model	$\sigma = 0$	$\sigma = 0.01$	$\sigma = 0.02$
Baseline	DeepSpeech2	9.7%	27%	63%
	Transformer	5.7%	12%	35%
	DeepSpeech2+ σ-AUG	-	11%	16%
Enhancement,	DeepSpeech2+ ASNR	-	17%	33%
augmentation	DeepSpeech2+ σ-AUG + ASNR	-	12%	18%
	Transformer+ ASNR	-	9.1%	19%
	DeepSpeech2+ Maj-100	-	27%	69%
	DeepSpeech2+ Avg-100	-	25%	62%
Voting	DeepSpeech2+ ROVER-8	-	21%	52%
	DeepSpeech2+ ROVER-16	-	20%	50%
	DeepSpeech2+ ROVER-32	-	20%	50%
Proposed models	DeepSpeech2+ ASNR + ROVER-16	-	14%	26%
	DeepSpeech2+ σ-AUG + ROVER-16	-	9%	**12%**
	Transformer+ ASNR + ROVER-16	-	**8.1%**	15%

Table 7.1: Word Error Rate (%) for DeepSpeech2 on the LibriSpeech clean test set under various defenses on *clean* utterances when adding Gaussian noise of deviation σ. σ-AUG stands for Gaussian augmentation of deviation σ in training - the same deviation used at inference. ASNR means A priori SNR filtering of inputs. Maj-N,Avg-N,ROVER-N refer to majority vote, logits averaging and ROVER voting strategies, using N forward passes.

Performance under Gaussian noise We report the performance of our model on noisy inputs (but no attack) in Table 7.1.

Augmentation and Enhancement We evaluate Gaussian augmentation on DeepSpeech2 and ASNR on both our architectures. Both techniques lower the word-error rate significantly under $\sigma = 0.01, 0.02$, with an advantage for the former. Interestingly, combining both techniques at once (on a DeepSpeech2 model trained on noisy *and* enhanced data) does not really improve results compared to using augmentation only. This suggests using ASNR enhancement as a "fallback option" in situations where retraining a model is not acceptable. This off-the-shelf method nonetheless provides competitive performance when using state-of-the-art architectures like Transformer.

Voting strategies We compare all our proposed voting strategies on DeepSpeech2 outputs. As expected, majority vote brings no significant improvement over one-sentence estimation (i.e. the baseline). Logit averaging is somewhat effective; however, it does not compare to ROVER, by far the best voting method even with fewer sentences.

Proposed models As a consequence, we propose two smoothing-based defenses:

- a **trained** defense using smoothing, augmentation, and ROVER. On DeepSpeech2, with $\sigma = 0.01$ (resp. 0.02) it reaches a WER of 9 (resp. 12)
- an **off-the-shelf** defense using smoothing, ASNR enhancement, and ROVER. Applied to DeepSpeech2 this defense suffers from a higher WER of 14 (resp. 26) but is still relatively

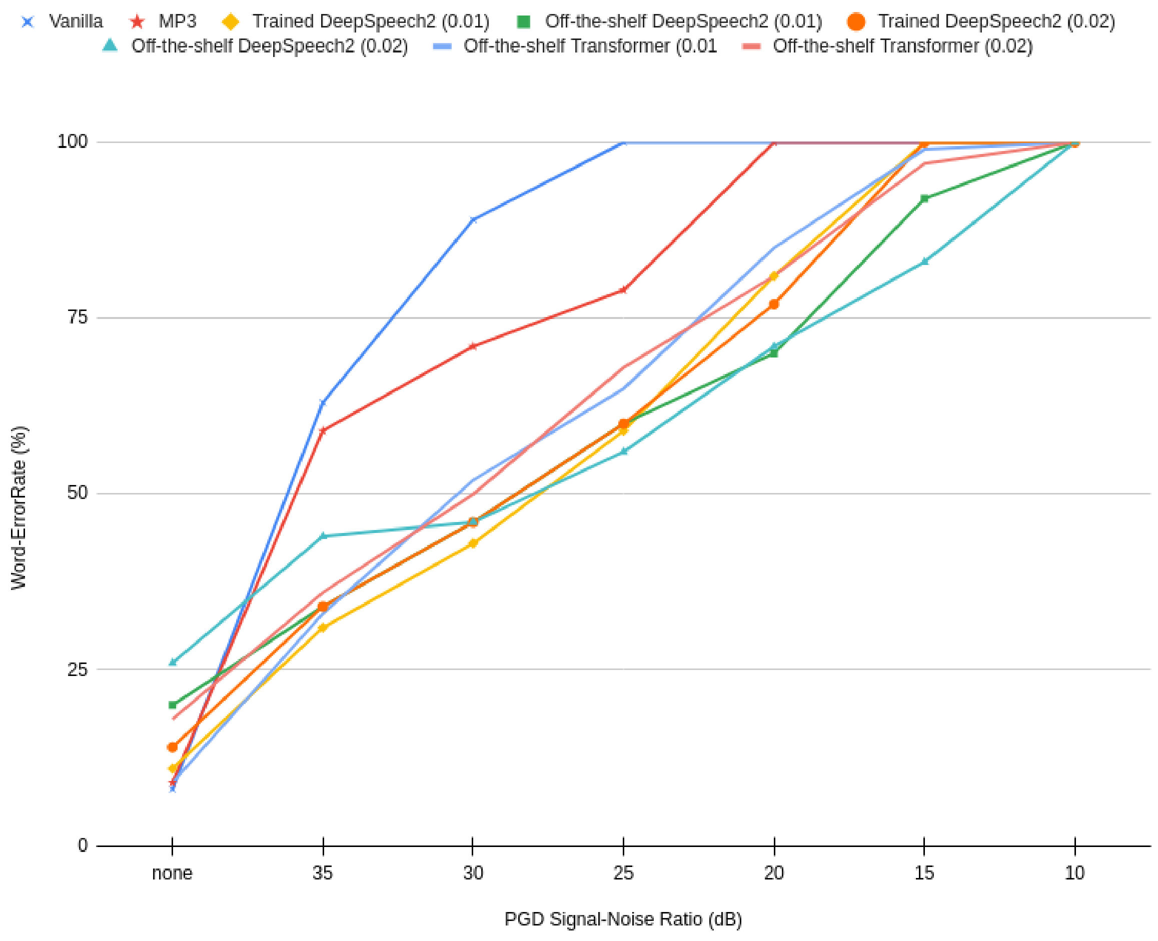


Figure 7.1: WER achieved against PGD attacks by the baselines (Vanilla model, MP3 compression) and the proposed trained (DeepSpeech2 with Smoothing, augmented training, and ROVER) and off-the-shelf (DeepSpeech2/Transformer with Smoothing, ASNR, and ROVER) defenses, with Gaussian deviations 0.01 and 0.02. We plot the results when varying the PGD SNR bound: a lower SNR means a larger perturbation.

effective. With Transformer it performs much better with a WER of 8.1 (resp 15).

Performance under attack In Figure 7.1 we plot the results of our defenses and baselines against the untargeted PGD attack, as a function of the SNR used to bound the attack. They demonstrate the effectiveness of ASR smoothing: compared to the vanilla DeepSpeech2 the Word-Error-Rate improves by 20 to 50 points for all PGD attacks bounded by SNR $\geq$ 20dB for our proposed models, with both $\sigma = 0.01$ and $\sigma = 0.02$. When SNR $= 25dB$ is sufficient to reach total denial-of-service (WER $= 100$) on DeepSpeech2, and 20dB for the MP3 compression baseline, the same feat requires SNRs of $10 - 15$dB to defeat our defenses.

Table 7.2 reports the results of the targeted CW attack. As expected for an unbounded attack, it is partially able to break our defenses (low WER on its target), but at a high cost. The SNR it requires to achieve these results is as low as $10 - 14$ under $\sigma = 0.01$ and $5 - 8$ with $\sigma = 0.02$!

Model	No attack	CW attack		
		GT	TGT	SNR
Vanilla	8	100	0	27dB
MP3 compression	9	100	3	16dB
Trained ($\sigma = 0.01$)	11	100	8	14dB
Off-the-shelf ($\sigma = 0.01$)	17	100	4	10dB
Trained ($\sigma = 0.02$)	14	100	6	8dB
Off-the-shelf ($\sigma = 0.02$)	31	100	6	5dB

Table 7.2: Word Error Rate (%) for DeepSpeech2 on the first 100 utterances of the LibriSpeech clean test set, for clean inputs and under the CW attack for the baselines and the proposed trained and off-the-shelf defenses, with Gaussian deviations 0.01 and 0.02. We report both the WER on the ground truth (GT) and the attack target (TGT), and the SNR required to achieve it.

This compares to 27dB for the undefended DeepSpeech2 and 16dB for the MP3 baseline. With the higher σ in particular, the adversarial noise becomes very much audible to the human ear. The tradeoff in clean (no attack) accuracy is fairly low for the trained defense even with $\sigma = 0.02$ ($+5\%$ WER). It is higher for the untrained, off-the-shelf defense, with which using a lower deviation may be required for practical applications.

7.4 Leveraging advances in Speech recognition modeling

Randomized smoothing presents the remarkable property of turning the adversarial robustness problem into a noise robustness problem. Under that framework, excellent model accuracy under Gaussian noise can be turned into excellent performance under L_2 adversarial attacks. This property is useful as performance under random noise is a highly active area, and progress in that area can be leveraged for security purposes. We have shown in this chapter how traditional speech processing is an effective way to boost adversarial robustness. On the image modality, Salman et al. [2020b], Carlini et al. [2023] have shown that improvements in scale and generative AI can also be used for robustness.

We wonder whether scale also helps with ASR robustness in the context of sequential smoothing. Whisper, for instance, shows excellent performance under Gaussian noise, as we have shown in section 5.3.1. Could it be a natural fit for randomized smoothing? SSL models pretrained on large unlabeled datasets are also known to be more noise-robust.

In Figure 7.2, we run on the Whisper base.en and small.en models, as well as Wav2Vec2, HuBERT, and Data2Vec models pretrained on LibriSpeech and LibriVox, the same PGD attack as in section 7.3.2, with and without smoothing at $\sigma = 0.02$. We compare them with our previously established off-the-shelf and trained models. We observed that those improved models indeed lower the WER under untargeted attacks when combined with randomized smoothing. This is a sharp contrast with section 5 in which we established that in the absence of any defense mechanism, the improvements of ASR have not carried over on robustness.

Overall, our results illustrate the high effectiveness of randomized smoothing as a defense that can leverage general performance improvements and turn them into robustness increases.

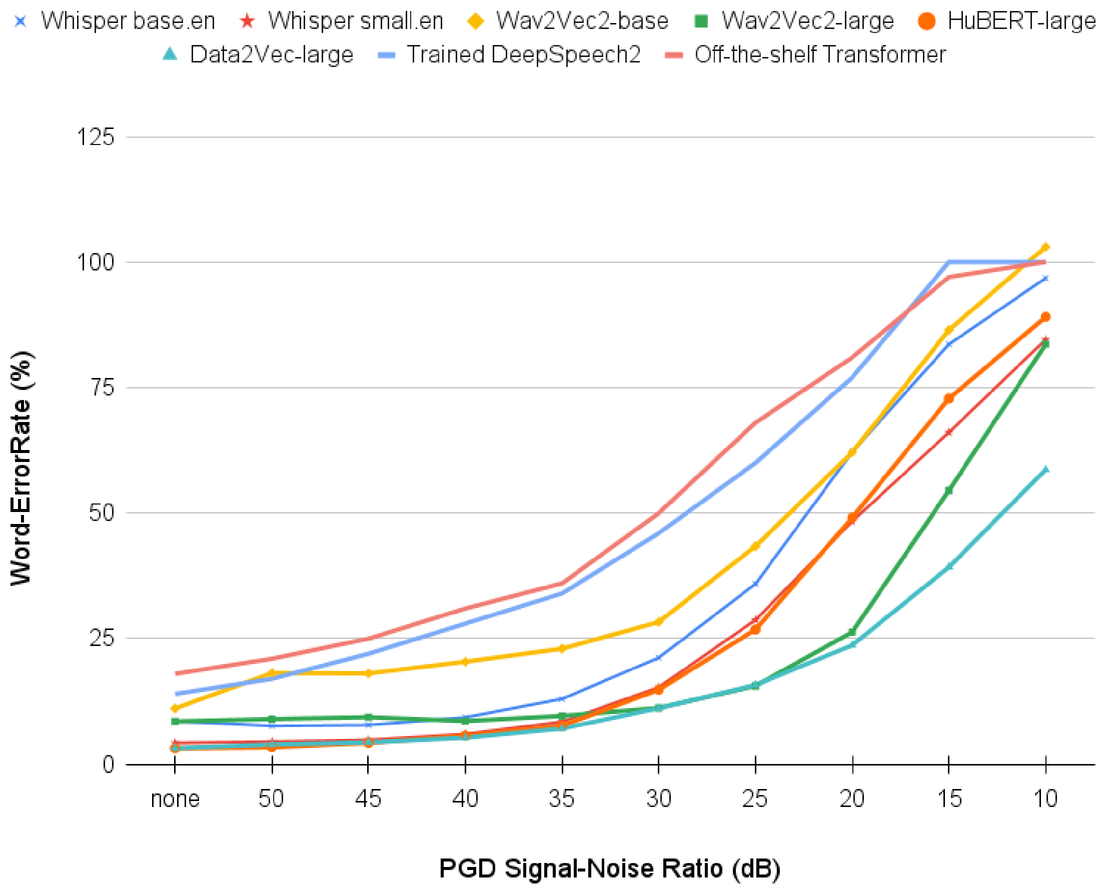


Figure 7.2: WER achieved against PGD attacks by applying sequential smoothing on Whisper English base and small, Wav2Vec2 base and large, and HuBERT and Data2Vec large. The latter two are pretrained on LibriVox and the Wav2Vec2 models on LibriSpeech. We use $\sigma = 0.02$, ASNR filtering, and ROVER-8 for all those models. We plot the WER of the models proposed in section 7.3 for comparison.

7.5 Smoothing high-frequency patterns and speaker classification experiments

The strength of randomized smoothing is that it makes almost no assumption on the characteristics of the input domain, or the nature of classifier f: linear, neural, or random models would be treated in the same way and fall under the same bounds. However, we argue that this strength is also a weakness, in the sense that no knowledge about classifiers and their adversarial vulnerabilities can be used to refine its predictions. We propose a general method to inject knowledge in smoothing, illustrated with a particular example of such knowledge.

Ilyas et al. [2019] showed that predictive patterns in data can be naturally divided between robust and non-robust. Wang et al. [2019] suggest that such non-robust patterns are enabled by high-frequency components, at least in the case of Convolutional Neural Networks (CNN). We

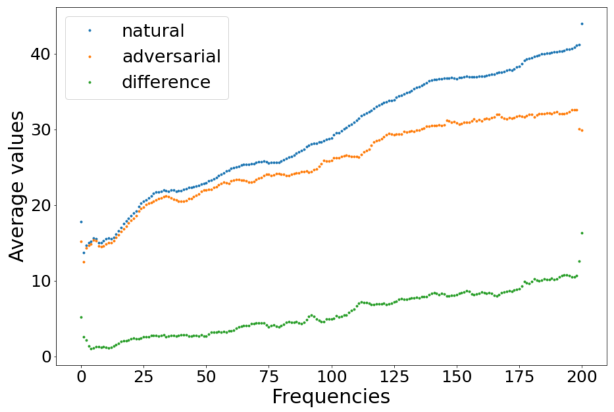


Figure 7.3: dB-spectrogram values of the natural and adversarial examples, as well as their difference, averaged on the small-LS test set (see 7.5.2). The examples were generated from an undefended model.

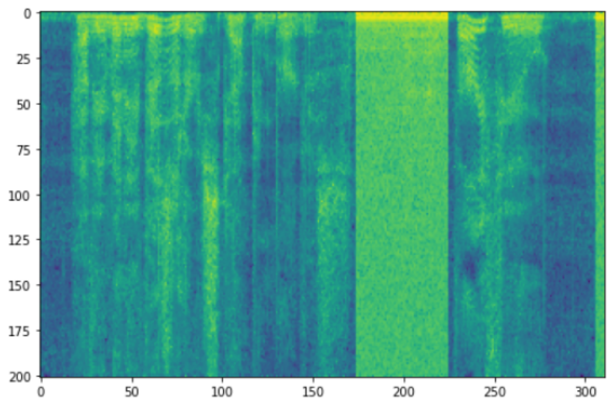


Figure 7.4: Spectrogram representation of the signal-to-noise ratio of an adversarial noise computed on a LibriSpeech utterance. The example was generated against an undefended classification model.

observe the same effect on the LibriSpeech speaker classification task (see section 7.5.2). To that end, we attack an undefended model, and we compute the spectrogram representations of natural and adversarial examples. We display those representations, averaged on the test set in Fig. 7.3, as well as the signal-noise ratio (SNR) per frequency on an example utterance in Fig. 7.4. The SNR in Fig. 7.4 makes it clear that low-frequency components are better preserved than their high-frequency counterparts. However, the range of highly affected components is not restricted to a small fraction; on the contrary, looking at Fig. 7.3, out of 200 frequency components only the 1-40 range is lightly affected (by less than a few percent).

7.5.1 High-frequency smoothing

The targeted effect of adversarial attacks on some frequency components naturally leads us to wonder how to use that phenomenon to our advantage when designing defenses. We propose to that end an approach called *high-frequency smoothing*. It consists in changing the nature of noise ν, from a frame-wise independent Gaussian distribution to one that specifically targets high frequencies. However, we want to keep the essential properties of the Gaussian distribution that make Gaussian smoothing so effective. To that end, we propose to apply a high-pass filter to the Gaussian noise. We use an elementary filter $\nu'_n = \frac{\nu_n - \nu_{n-1}}{2}$, i.e. we substract from noise ν_n its moving average of order 2. This is a pre-emphasis filter with z-transform $z(1-z)$. We refer to this defense as **MAD smoothing** ("Moving Average Difference").

Our intuition is that, at equal Gaussian deviation σ, applying ν or ν' is equally effective at smoothing potential perturbations, as they lie in the bandwidth of the high-pass filter. However, the latter then has the advantage of a *smaller noise amplitude*, which makes prediction easier for our learning algorithms. This gives MAD smoothing the edge in terms of robustness-performance tradeoff. We now wish to verify this intuition empirically.

7.5.2 Experimental Setup

MAD and Gaussian smoothing can be compared not only on ASR but on any task with audio inputs. We however observe varying effects across different tasks. We report the bulk of our experiments on Speaker classification, the task that we experimented on originally. Then we present additional results on Noise classification and Speech Recognition.

Model and Dataset We work on a subset of the LibriSpeech Speaker Identification task (referred to as small-LS below). From its development set, we retain 40 speakers, each with 8 minutes of audio. We divide that data into a 50%-25%-25% train-val-test split and consider the task as classification among the 40 speakers. This reasonably sized dataset allows us to perform multiple experiments. Our model is a 1D CNN with 4 convolutional layers and a final linear projection. Waveform amplitudes are normalized to $[0-1]$ and adversarial noises are always applied on this range ($x+\delta \in [0,1]$).

Attacks and defenses We use the l_∞-Projected Gradient Descent (PGD) attack (2.1.3) with $\epsilon = 0.003$ and 20 iterations. When a defense seems effective on Small-LS we additionally evaluate out against the L_2-PGD ($epsilon = 1$) attack Carlini and Wagner [2016].

In addition to Gaussian and MAD smoothing, we tried several other methods for comparison.

- *Adversarial training*. We use 50% of adversarial instances in the first half of training, and 80% in the second half.
- *Lowpass filters* on the inputs directly, as a "naive" way to remove high-frequency components from the prediction pipeline. Since our models use spectrogram representations as input features we can use "ideal" lowpass filters, i.e. truncate the high-frequency range from those representations. We retain the lowest 5% frequency components (i.e. the [0-10] band out of 201 frequencies). Alternatively, we try applying a biquad lowpass filter on the raw inputs. In each case, the filtering method is used in both testing and training.

7.5.3 Results

All models run are trained 200 epochs, each of which takes less than half a minute due to the reasonable size of the dataset, but also the great scalability of every proposed method other than adversarial training, which takes about 5 times longer.

We display in Table 7.3 the results of all proposed defenses on small-LS, and compare them to those of the baselines. From those results, it is rather clear that filtering methods are ineffective. Their adversarial accuracy is only marginally higher than the undefended model, with an important drop in natural accuracy. This is not very surprising when considering the failure of similar preprocessing schemes on image tasks [Athalye et al., 2018a]. On the other hand, using direct lowpass filters does protect against the high-frequency adversarial vulnerabilities of the vanilla model. We show that by performing transferability experiments between those two models, displayed in Table 7.4. As we can see, when we compute adversarial examples against one model and feed them to the other, the target model retains most of its natural accuracy.

This hints that the bad adversarial accuracy of lowpass models is due to different vulnerabilities, that were created by applying the lowpass filter. We theorize that restricting our models to

Defense	Nat. accuracy	L_∞-PGD accuracy	L_2-PGD accuracy
None	88%	7%	10%
Ideal lowpass (5%)	68%	10%	-
Biquad lowpass ($f = 200$)	71%	9%	-
Gaussian smoothing ($\sigma = 0.1$)	71%	69%	63%
Adversarial training	77%	31%	-
MAD smoothing ($\sigma = 0.15$)	80%	**74%**	**68%**

Table 7.3: Results of the baselines and the smoothing approaches on small-LS, on natural (column 2) and adversarial inputs, using L_∞-PGD (column 3), and in case of success the L_2-PGD (column 4) and CW attack (column 5).

Source	Target	Adv. accuracy
Undefended	Ideal lowpass (2%)	72%
Ideal lowpass (2%)	Undefended	74%
Ideal lowpass (2%)	Biquad lowpass	39%

Table 7.4: Transferability of attacks between vanilla models and models focusing on low frequencies, on the small-LS dataset. Column 3 refers to the accuracy of the target model when fed adversarial inputs generated against the source model.

the low-frequency domain weakens the input representations used by the model and creates extra vulnerabilities. Therefore completely eliminating a range of the input space where adversarial examples lie does not yield good adversarial robustness while downgrading significantly the natural accuracy of the models (from 88% to 60-70%).

On the other hand, MAD smoothing does improve on traditional randomized smoothing. For both MAD and Gaussian smoothing we reported in the table the noise deviation that maximized the results, leading to using a greater σ for the former- which makes sense as the high-pass filter diminishes the amplitude of the noise. Since the latter is more effective than adversarial training, that makes high-frequency smoothing our best defense mechanism. The additional results using L_2-PGD attacks confirm those results.

7.5.4 Additional experiments

We now compare Gaussian and MAD smoothing on additional tasks.

In Table 7.5 we report results on the UrbanSound8K noise classification task. Models are 2-D CNNs with 5 convolutional layers and a 2-layer final projection. The attack is the L_∞-PGD adver-

Defense	Nat. accuracy	Adv. accuracy
None	78%	13%
Gaussian smoothing ($\sigma = 0.1$)	65%	63%
MAD smoothing ($\sigma = 0.15$)	68%	64%

Table 7.5: Results of the baseline and the smoothing approaches on the UrbanSound8K dataset, on natural (column 2) and L_∞-PGD adversarial inputs with $\epsilon = 0.02$ (column 3).

Defense	Nat. WER	Adv. WER
None	9.7%	88%
Gaussian smoothing ($\sigma = 0.01$)	14%	46%
MAD smoothing ($\sigma = 0.01$)	13%	51%
MAD smoothing ($\sigma = 0.015$)	18%	46%

Table 7.6: Results of vanilla and smoothed DeepSpeech2 models on the LibriSpeech dataset, on natural (column 2) and L_∞-PGD adversarial inputs with SNR $= 30$dB (column 3). The Gaussian model is equivalent to the off-the-shelf model in section 7.3.2

sarial algorithm $\epsilon = 0.02$. We observe that MAD smoothing outperforms Gaussian smoothing in this different context, although by a shorter margin.

On the other hand, MAD smoothing applied to the DeepSpeech2 ASR model does not seem to bring any improvement, as evidenced in Table 7.6. We simply use the off-the-shelf model from section 7.3.2, and for MAD smoothing modify the ASNR filter before our high-frequency noise. Regardless of the deviation used for MAD smoothing, either its natural or adversarial WER is greater than that of Gaussian smoothing, indicating no progress in the robustness-performance tradeoff. Several important differences between ASR and Speaker classification experiments may factor in these contrasting behaviors: for example the use of ASNR enhancement, or the values of deviation σ which vary by an order of magnitude between both experiments. It could also be explained by properties specific to ASR adversarial examples - in that regard the work proposed in Chapter 6 may eventually shed new light on this phenomenon.

7.6 Related work

Efforts to adapt adversarial training or relaxation methods to ASR have been limited so far: Sun et al. [2018] have used training for speech based on the FGSM attack, which is simpler but not nearly as robust as PGD training. Most proposed ASR defenses such as MP3 compression [Das et al., 2018] or quantization [Yang et al., 2019] have shown the same weakness as above to adaptive attacks [Subramanian et al., 2019]. Exploiting temporal dependencies in speech to detect adversarial manipulations [Yang et al., 2019] is a promising line of work. However, at best it only enables the user to *detect* these modified inputs. *Reconstructing* the correct transcription is an entirely different challenge, and our objective in this work.

Noise-based defenses for *audio classification* have been proposed: for instance, Subramanian et al. [2019] use simple white noise as a defense mechanism. This is a straightforward extension of randomized smoothing to another classification setting. Regarding specifically ASR, the only existing randomized smoothing works we are aware of are Mendes and Hogan [2020], which propose an adaptation of the noise distribution to psychoacoustic attacks, and the recent Żelasko et al. [2021]. The latter in particular thoroughly explores the effects of Gaussian smoothing on DeepSpeech2 and the SpeechBrain Transformer. However, their work on making these models more robust to white noise is limited to Gaussian augmentation in training. Specifically, they do not explore the issue of voting on transcription and resort to one-sentence estimation (see section 7.2.2), which limits the amount of noise they can use, and therefore the radius of their defense. Besides, they do not use adaptive attacks which makes their evaluation incomplete. To our knowledge, we propose the first *complete* (randomization, training, and vote, evaluated on adaptive attacks) version of randomized smoothing for Speech Recognition.

7.7 Discussion

We have proposed a state-of-the-art adversarial defense for ASR models based on randomized smoothing. It is successful against all attacks using inaudible distortion while retaining a low error rate on natural data. To achieve strong performance under noise, we have leveraged speech enhancement methods and proposed a novel use for ASR output ensembling methods like ROVER. We successfully defend against state-of-the-art adaptive attacks and analyze the importance and limits of each component of our defense. In appendix E, we study the certification of this sequential randomized smoothing for ASR, which poses new challenges compared to its classification counterpart. We show that sequential smoothing is to an extent provably robust, although practically useful robustness bounds would be difficult to compute.

This adaptation of existing defenses to ASR is specific to Randomized Smoothing, for the reasons discussed above. Adversarial training, which takes a heavier computational toll but usually leads to better accuracy, is the focus of the next chapter.

Chapter 8

Robust pretraining with adversarial masked prediction

As we have seen in Chapter 7, randomized smoothing is a powerful defense, but has limits. Notably, its guarantees only apply to L_2-bounded threat models, and it suffers from a tradeoff that remains important. On classification tasks, randomized smoothing is a state-of-the-art method for L_2 certified robustness, but not for empirical robustness. Almost all top places on known benchmarks are held by versions of adversarial training. This motivates us to explore the application of adversarial training to ASR in this chapter.

That objective is particularly challenging. As we will show, adversarial ASR training has trouble converging to low loss values. We suspect that end-to-end transduction losses like CTC mix poorly with adversarial training when compared to classification losses like cross-entropy. This explains why adversarial training has to our knowledge never been applied to complex speech tasks like ASR in the past. On the other hand, many popular Self-Supervised Learning (SSL) methods rely on the *masked prediction* of a finite set of discrete classes, in a manner similar to frame-wise classification. This suggests that applying adversarial examples during SSL pretraining may be more effective than during supervised ASR training. This observation leads us to propose an adversarial training algorithm for the self-supervised learning of speech models.

8.1 Introduction

Our proposed defense is termed **AdvMP** which stands for **adversarial masked prediction** (section 8.2). AdvMP is a training method based on a general adversarial attack on masked reconstruction pretext tasks, using a clean model to extract targets. We apply AdvMP during the training of a HuBERT model. We evaluate the performance and robustness of that model, named **AdvHuBERT**, against several adversaries. Against PGD attacks [Madry et al., 2018] of radius 0.01, our best model achieves a Word-Error Rate of 50-60% where an undefended model gets $>100\%$. On the unbounded Carlini&Wagner attack [Carlini and Wagner, 2018], fooling AdvHuBERT requires an additional 23dB of distortion compared to undefended models. In addition, AdvHuBERT recovers a WER of 13.5% on the adversarial dataset proposed in chapter 6, which was explicitly designed to fool SSL-pre-trained models with transferable perturbations.

We study different finetuning strategies for AdvHuBERT, both with standard and adversarial training. We compare the robustness of that model to several baselines, including an adversarially finetuned HuBERT, a CTC model we adversarially train from scratch, and a HuBERT model defended with Sequential Smoothing (chapter 7). Our experiments lead to the following conclusions:

- Adversarial pretraining provides much stronger robustness than adversarial ASR training or finetuning
- Finetuning adversarially pretrained models with standard losses may lead to robustness forgetting and requires caution.
- AdvMP is a promising defense, but making it competitive with sequential smoothing would require a heavy computational cost

As is common for adversarially robust models in all tasks, AdvHuBERT does not match the performance of SSL or ASR models with standard training on clean data. Our proposed model achieves a Word-Error Rate of 6.7% on the LibriSpeech test-clean set. For reference, this is approximately the performance achieved by a model of similar architecture trained from scratch for ASR, without any pretraining and without adversarial objectives [Higuchi et al., 2021].

8.2 Adversarial masked prediction

Although well established in computer vision, adversarial training has thus far never been applied in speech tasks beyond classification. Adapting it for speech pretraining requires crafting an adversarial objective for SSL pretext tasks, taking into account the fact that targets are also functions of the input, rather than fixed as in Equation 2.9. In this section, we introduce *adversarial masked prediction* (AdvMP), which in theory can be applied to any model trained under Equation 8.1.

8.2.1 Background: Predictive SSL pretraining

A large family of SSL speech models relies on *predictive* pretext tasks: targets are computed outside of the computation graph, and the model is trained to minimize the loss against those targets given a noisy or masked input. Many such methods are inspired by Masked Language Models in NLP, like BERT, and generate discrete targets with clustering [Hsu et al., 2021] or quantization [Chiu et al., 2022]. Other methods include using a teacher model to generate continuous targets from clean inputs [Baevski et al., 2022b]. Formally, all methods optimize an objective of the form:

$$\min_{\theta} \mathcal{L}(f_\theta(x_{\text{mask}}), t(x)) \tag{8.1}$$

with input x, f_θ the model, t the target generating model and $\mathcal{L}$ the loss function (e.g. cross-entropy or L_p).

An important factor in predictive SSL's success is the quality of the target generator t. Different models handle this challenge in various ways. Data2Vec [Baevski et al., 2022b] uses a moving average of the model as a teacher and a carefully chosen optimization schedule to encourage high-quality targets. HuBERT generates targets with K-means clustering in several phases but needs an existing speech representation to cluster. Training initially uses MFCC targets, which are replaced and iteratively improved with intermediate HuBERT checkpoints

8.2.2 Definition

AdvMP optimizes the following objective:

$$\min_{\theta} \max_{\|\delta\|_\infty \le \epsilon} \mathcal{L}(f_\theta((x+\delta)_{\text{mask}}), t(x)) \tag{8.2}$$

The inner objective of this problem is approximated using a specific perturbation δ, computed for each x using PGD (Equation 2.2) k steps to maximize loss $\mathcal{L}(f(x_{\text{mask}}+\delta), t(x))$. Intuitively, the adversarial perturbations fool the model but leave the target generator unchanged. After pretraining, the model can be finetuned with standard training objectives, and rely on its pre-training initialization to retain robustness in downstream tasks. Another option is to finetune it with the adversarial training objective of Equation 2.9, computing untargeted perturbations against the downstream task's loss and targets.

8.2.3 Practical implementation

Despite its simple formulation, AdvMP must in practice be carefully implemented. As discussed in section 8.2.1, SSL pretraining requires high-quality targets. Accordingly, since adversarial training lowers performance on clean data, using our model to generate targets can lead to subpar supervision. For instance, during an early attempt to apply Equation 8.2 to an existing Data2Vec training recipe, we observed that adversarial training collapsed to a trivial solution after 10k steps, where the teacher generated constant targets.

Following the scheme of HuBERT in using our own intermediate model to generate targets would likely also be problematic. We apply an alternative solution by using the model trained on *clean* data to generate targets; then we train a new model adversarially on those targets. Specifically, we apply K-means clustering to representations extracted with the HuBERT BASE model. We then fix those targets and apply Equation 8.2 to adversarially train our model **AdvHuBERT**. Our targets are the same used to train the HuBERT LARGE models and of higher quality than those used to train the original HuBERT BASE model.

8.3 Experimental setting

8.3.1 Pretraining

Our architecture is the HuBERT BASE model [Hsu et al., 2021] with no changes. SSL is computationally expensive, and adversarial training multiplies the compute budget by the number of PGD steps that must be used before every training iteration. To keep our computation cost low, we apply the following tricks:

- We use only $N = 1$ PGD step with step size $\eta = \epsilon$, combined with random initialization of the perturbation δ. In computer vision, this was shown in some cases to be almost as effective for training as using stronger attacks [Wong et al., 2020].

- We reduce the number of steps in HuBERT pretraining compared to the original HuBERT BASE models. While it was trained for 650k steps in total, we only use 100k.

We run pretraining on 960h of LibriSpeech data, for a fair comparison with previous SSL speech models which are commonly evaluated on that dataset. We keep the original HuBERT BASE pretraining recipe, changing only the targets and scheduler. We decay the learning rate linearly for 68k steps after a 32k-step linear warmup to 0.0005. Our adversarial radius is $\epsilon = 0.01$

8.3.2 Finetuning

We finetune our models to evaluate ASR performance and robustness. As in previous works, we append three linear layers to our pretrained speech representation. The final layer projects representations into an output space of size 31 (English characters and special tokens). We restrict finetuning to 100k steps for 960h of LibriSpeech, 20k steps for 100h, and 4k for 10h. We warm-up for 10% of steps with the pretrained features kept frozen, and an initial learning rate of 0.0001 for the pretrained layers and 0.1 for the final layers. We decrease these learning rates by a factor 0.8 when the validation loss reaches a plateau.

After 100k pretraining steps, we finetune models and verify their robustness. We experiment with multiple forms of finetuning:

- Standard CTC finetuning, on 100h and 960h of LibriSpeech training data
- Adversarial CTC finetuning with 1-step PGD on 10h, 100h and 960h
- Adversarial CTC finetuning with 10-step PGD on 960h

8.3.3 Implementation and computation details

We use the Fairseq [Ott et al., 2019] HuBERT recipe for pretraining, modified with a custom implementation of adversarial training. For finetuning, we use the SpeechBrain [Ravanelli et al., 2021] binding for Fairseq models and customize the SpeechBrain Wav2Vec 2.0 finetuning recipe. All parameters are kept unchanged in these recipes apart from the number of iterations. To run and transfer adversarial attacks we use robust_speech (Chapter 5).

We pretrain advHuBERT with a batch size of 25min of audio and finetune all models with a batch size of 3min of audio. The entirety of our experiments required about 1000 GPU hours (600 for pretraining and 400 for all finetuning). This is lower than the computational cost of pretraining a single HuBERT BASE model [Hsu et al., 2021]. We trained on 4 Tesla V100 GPUs for a total of ten days.

8.3.4 Evaluation attacks

We run all white-box attacks on 100 sentences from the LibriSpeech test-clean set. For the PGD attack, we use a fixed radius $\epsilon = 0.01$, the same as adversarial training, which corresponds to an SNR of 33dB on average on the evaluated set. We use different numbers of steps: this gives a partial indication of whether the observed robustness is trustworthy. At fixed ϵ, if the model is robust to low-step attacks only then adversarial examples are harder to find but still present; while robustness to high-step attacks is a stronger sign. We run this attack with different numbers of iteration steps and evaluate it using the WER of the output computed against the true transcription: the higher, the more successful the attack.

We apply the Carlini&Wagner audio attack [Carlini and Wagner, 2018] with initial radius $\epsilon = 0.04$ and 4000 iterations, using the fixed target sentence "OK Google browse to evil dot com". We evaluate it by reporting the Word-Error Rate on the perturbed input against this target sentence (a successful attack gets WER $= 0$), as well as the Signal-to-Noise Ratio expressed in decibels. An SNR above 30dB corresponds to a discrete background noise, and above 40dB to a barely perceptible noise.

Finally, we apply the adversarial dataset which we proposed in Chapter 6 to evaluate the robustness of pretrained ASR models. Results on that dataset indicate whether the evaluated model is robust to transferred attacks from undefended proxies. Generated against Wav2Vec2 [Baevski et al., 2020] and HuBERT LARGE proxies, this dataset fools a large number of SSL-pretrained ASR models. We report the WER both on the attack target and on the true transcription of those inputs.

8.4 Results

In Table 8.1 we report the results of our attacks on all models, and compare them with finetuned HuBERT baselines. Multiple other SSL-pretrained models (Wav2Vec2, Data2Vec, ...) have been studied under attack in the past on some threat models [Wu et al., 2022], including Chapters 5 and 6 of this thesis, and have not shown to be more robust than HuBERT; we do not reproduce those results here.

8.4.1 Comparing AdvHuBERT to HuBERT

Lines 1-3 correspond to our AdvHuBERT model finetuned with 1-step attacks on different LibriSpeech subsets. We can observe that these models are considerably more robust than undefended HuBERT models finetuned on the same data (lines 7-9) on all white-box attacks. Against the 100-step PGD adversary, where the baselines are entirely fooled with WERs of 100%, AdvHuBERT's WER reaches 50-60% when training on at least 100h, and under 80% for the 10h subset. As for the targeted CW attack, while it succeeds in fooling all models at least partially, it requires an SNR of 24dB to do so on AdvHuBERT, where only 47dB are needed for baselines. In other words, where imperceptible noise easily fools undefended models, AdvMP-pretrained models require much more audible noise. Interestingly, increasing the amount of finetuning data from 100h to 960h benefits clean performance (-3.3% WER), but not necessarily robustness (+5% with 100 steps).

We note that the WER of the robust models on clean test data is much higher than their non-robust counterparts, or than any recent results on LibriSpeech for similar architectures. Our strongest robust model achieves a WER of 6.7% on the test-clean set. We emphasize that as mentioned above, such a drop is the expected cost of any robust model on all benchmarks.

8.4.2 Influence of finetuning mechanisms

We evaluated several forms of finetuning on On AdvHuBERT and report their results in lines 1-6. We achieve our best results when finetuning adversarially with one attack step - in which case the

Model	#	Finetuning data	Finetuning adv. steps	Clean WER↓	PGD WER 1 step↓	PGD WER 100 steps↓	CW WER tgt.↑	CW SNR↑	transferred (6) WER tgt.↑	transferred (6) WER true↓
AdvHuBERT	1	960h	1	6.7	16.8	56.5	0.4	**24dB**	100	**13.5**
	2	100h	1	10.0	22.6	**51.9**	1.2	**24dB**	100	16.7
	3	10h	1	22.8	37.4	78.6	0	**24dB**	100	32.9
	4	960h	0	5.2	71.1	114	2.4	35dB	97.6	83.0
	5	100h	0	10.5	64.5	69.1	0	36dB	99.6	59.5
	7	960h	10	33.7	54.1	76.9	63.0	**24dB**	100	36.2
HuBERT baseline	7	960h	0	3.5	20.4	110	0	47dB	55.7	100
	8	100h	0	6.3	28.2	103	0	47dB	53.9	100
	9	10h	0	11.5	36.7	98.1	0	47dB	73.5	100
	10	960h	1	5.6	**14.8**	72.8	0	28dB	100	13.6
w/o SSL	11	960h	1	21.9	17.8	97.0	0.9	**24dB**	100	39.9
Smoothing, $\sigma = 0.01$	12	960h	0	5.8	10.7	63.1	53	**24dB**	100	53.8
Smoothing, $\sigma = 0.02$	13	960h	0	11.4	19.6	55.9	76	**24dB**	100	31.9

Table 8.1: Performance and attack results for ASR models finetuned from the AdvHuBERT 100k checkpoint and HuBERT. For each model, we specify its finetuning data and the number of attack steps during finetuning (0, 1, or 10). We report in that order the WER on the unmodified LibriSpeech test-clean set, on the true transcription under PGD attacks, the WER on the attack target and the achieved SNR for the CW attack, and the WER on both the attack target and true transcription for the transferred attack. All attacks are run on 100 sentences from the test-clean set. Arrows on metrics indicate which change shows an increase in robustness or performance.

Word-Error rate achieves 60%. More pretraining steps worsen clean performance significantly (to 33%) which leads to high WERs under attack as well.

Finetuning without adversarial noise can lead to some robustness, as shown on the 100h model. However, trying to finetune on 960h, with more steps, the model reaches better clean performance but performs on par or worse than the baselines under attack. This suggests that too much clean finetuning can make models forget the robustness learned during pretraining.

8.4.3 Adversarial finetuning of undefended SSL models

Since adversarial finetuning helps achieve robustness in ASR, one may wonder whether adversarial pretraining is necessary in the first place. Couldn't similar robustness be achieved by adversarially finetuning a standard baseline? To answer this question, we run one-step finetuning on HuBERT, and report the results in line 10. This approach is beneficial to clean performance, reflected by a WER of 5.9% on the test-clean set; and during finetuning the model learns robustness to one-step PGD (14.8% WER).

However, compared to AdvHuBERT this robustness carries over more poorly to stronger white-box adversaries. The 100-step PGD attack reaches 84.4% WER, compared to the $< 60\%$ of AdvHuBERT; and CW finds perfectly successful examples with an SNR higher by 4dB. This suggests that the features learned during standard masked prediction are vulnerable to adversarial attacks and that some of this vulnerability is not unlearned with adversarial finetuning. In contrast, learning robust features from scratch with AdvMP leads to more robust models.

What about training adversarially a model for ASR from scratch, without any pretraining? We attempt such a training, using 100k CTC training steps under 1-step attacks, starting from a random initialization of the HuBERT weights. We report the results of that model in line 11 of Table 8.1. Even more than the adversarially finetuned HuBERT, that model "overfits" the 1-step adversary, to the point that its WER under that attack is better than its clean WER. The model shows no "real" robustness as measured by the 100-step PGD attack (WWER 97%). Its clean performance is also subpar compared to most AdvHuBERT models.

However, more forms of adversarial training are effective against targeted adversaries. The model trained adversarially without SSL, and the adversarially finetuned HuBERT achieve 24dB and 28dB SNRs against the CW attack, which is comparable or close to AdvHuBERT. The transferred CW attack cannot fool those models to predict the target, and they recover 39.9% and 13.6% true WER respectively. This shows that substantial protection against targeted attacks is simple to achieve compared to untargeted attacks.

8.4.4 Comparison with sequential smoothing

Finally, we evaluate how AdvMP compares to sequential smoothing, which we proposed in Chapter 7. We apply this defense to HuBERT with $\sigma = 0.01$ and $\sigma = 0.02$ and report the results in lines 11 and 12 of Table 8.1.

At first sight, we observe that AdvMP is competitive with sequential smoothing, which achieves a WER of 55.9% for $\sigma = 0.02$. However, we must recall that smoothing is designed to be robust against L_2 attacks, while we are currently running L_∞ attacks. When applying attacks of equivalent SNR in L_2 norm, as in Chapter 7, we obtain a much better WER of 23.5%. It is highly unlikely that an application of AdvMP identical to ours but with L_2 PGD attacks would lead to a WER as low. That AdvMP is only as good as smoothing on L_∞ perturbations tends to indicate in fact that sequential smoothing is, in fact, more effective.

We argue that applying AdvMP with a stronger attack and more intensive training would fill the gap between both methods. Despite our adoption of strategies to train under limited resources, our current setting is likely suboptimal. Unfortunately, adding attack steps in training would require allocating considerable resources to model training. Thus the main limiting factor of AdvMP is, at this time, its high computational cost.

8.5 Related Work

Adversarial perturbations on computer vision are a large research area. Past works have studied robust self-supervised methods, often with a focus on contrastive learning Jiang et al. [2020], Kim et al. [2020], Fan et al. [2021].

Few adversarial defenses for ASR models have been proposed, and they have focused on either randomized smoothing or defenses using Speech processing methods to detect adversarial examples Yang et al. [2019]. Both methods have weaknesses, respectively a very important performance tradeoff and limitation to certain threat models (Chapter 7) and a vulnerability to some adaptive attacks Tramer et al. [2020].

Adversarial training as a defense has to our knowledge never been proposed for ASR or SSL speech models. It is not to be confused with methods using an adversarial model to generate augmented data or discriminate model outputs, also referred to as adversarial training Liu et al. [2018].

8.6 Discussion

We have proposed AdvMP, an adversarial pretraining paradigm, and applied it to pretrain the AdvHuBERT model. This speech encoder can be successfully finetuned into ASR models that display robustness to adversarial attacks. AdvHuBERT is both the first robust SSL speech model and the first adversarially trained ASR model. We have compared several finetuning strategies and found one-step finetuning to be the most efficient one. We have also discussed the limitations of our current approach.

Based on the results of this chapter and the previous one, how can we optimally make ASR models robust? And is the cost of that implementation acceptable in practice? In the next and final chapter, we recapitulate our results, make recommendations and discuss future perspectives for the field of speech robustness and security.

Chapter 9

Conclusion

In this thesis, we have explored the evaluation, execution, and mitigation of adversarial attacks and how those affect the security of Speech recognition systems. We have spanned multiple modalities, model architectures, training paradigms, and attack threat models, while switching perspectives between attacker and defender. We have striven to fit all of those perspectives within a consistent framework, but have had to remain flexible in our approaches: adversarial robustness and security are highly versatile concepts that are not easily narrowed down.

A key takeaway from this thesis is that robustness insights are not universal across modalities, models, and datasets. Variance in adversarial accuracy is only observable at a small scale (Chapter 3). Modeling decisions that seem to not affect robustness may improve it depending on the metric (Chapter 4). Attacks that are highly effective on one architecture or dataset may be ineffective on another (Chapter 5). The transferability of adversarial perturbations is easier on images than on ASR and is highly dependent on modeling choices (Chapter 6). Thus to improve robustness, it is worth paying considerable attention to the context at hand. The recent literature has largely focused on context-agnostic adversarial defenses, but domain knowledge and careful modeling choices can prove highly beneficial when applying those same defenses (Chapters 7 and 8). We summarize all of our contributions in more detail in section 9.1

Another important lesson from our work is that the speech modeling field is overall headed towards a more vulnerable state. The impressive progress of ASR has not carried over in terms of adversarial robustness (Chapter 5), while models have become increasingly ubiquitous. Against certain threat models, the most beneficial factors of improvements, especially in low-resource settings, are direct causes of increasing vulnerability. This widening gap between adoption and safety is a cause for concern, which we hope that our work on ASR robustness (Chapters 7 and 8) can mitigate. In Section 9.2 we discuss possible future directions for the field of speech and audio security and how our thesis fits in them.

9.1 Summary of key contributions

- In **Part I** we introduce new methodologies for evaluating adversarial robustness. In **Chapter 3**, we propose a probabilistic measure of adversarial accuracy under fixed threat models over randomized experiments. The surprisingly large variance in robustness in many of our experiments makes deterministic robustness evaluation unreliable. We show that this method is effective at quantifying the effect of hyperparameter choices on robustness at a small scale.

In **Chapter 4**, we propose *adversarial sparsity*, a robustness metric that scales to high dimensions while offering a finer granularity than adversarial accuracy. We motivate sparsity geometrically, define it formally, and tie it mathematically to the size of the adversarial set over a point. We propose a general formulation for sparsity that can be adapted to specific bounded threat models, which we do for L_∞ and L_2 attacks. We show how sparsity can measure high differences in robustness between models that accuracy would deem non-robust.

Both accidental robustness and adversarial sparsity help us make informed modeling choices.

- In **Part II**, we evaluate the robustness of ASR models in particular. We start with white-box attacks in **Chapter5** and show that they can pose a significant threat in certain cases. We illustrate how the recent improvements in ASR (from LSTM to Transformers, monolingual to multilingual, etc.) tend to make models more vulnerable. We show that model and data scaling are not solutions. We show that those attacks have numerous consequences, including indirect effects on data privacy leakage which are particularly relevant for commercial ASR models.

  In **Chapter 6** we extend those considerations to black-box threat models, by showing that the family of self-supervised ASR models are uniquely vulnerable to *transferable* targeted attacks. We harness that phenomenon to train an adversarial dataset which fools a large panel of state-of-the-art ASR models. We conduct an ablation study and show that SSL pretraining is indeed a cause of this vulnerability. Our results indicate that the need for robust ASR training is growing fast over time.

- We address this need in **Part III**. In **Chapter 7** we combine the generic randomized smoothing framework with speech processing tools that help mitigate its inherent toll on model performance. The result is a **sequential smoothing** mechanism that can be applied off-the-shelf to any ASR model. We apply sequential smoothing to standard and recent models and show that it drastically increases their robustness against white-box L_2 perturbations. We also show how this robustness is to an extent certifiable. Our results are the current state-of-the-art for L_2 robustness on the LibriSpeech dataset.

  Because sequential smoothing only applies to a certain class of perturbations, we also evaluate the applicability of adversarial training to ASR in **Chapter 8**. We propose an unlabeled adversarial training objective which we term **adversarial masked prediction**. We show that it is applicable as a self-supervised objective to pretrain speech encoders. Using adversarial masked prediction we pretrain, then finetune a model called **AdvHuBERT**. We evaluate AdvHuBERT and show that it displays improved robustness against white-box attacks and transferred attacks. Both sequential smoothing and adversarial masked pretraining are imperfect but promising methods for robust ASR training.

9.2 Future directions

We expect speech technologies to occupy a growing share of the AI landscape. Not only are ASR systems getting more efficient, popular and widely adopted in their current form, it is likely that similar systems will find new applications as well. For instance, the most widely used AI systems currently rely on conversational, chatbot-type applications based on Large Language Models (LLM) [Brown et al., 2020, Touvron et al., 2023], with a core focus on the text modality. Yet the full scope of human conversation relies on spoken language primarily, and in our opinion, it is only a matter of time before the rise of audio-first, massively used conversational agents. Whether those agents cascade ASR, LLMs, and speech synthesizers[1] or use end-to-end architectures [Hassid et al., 2023], we believe that this transition will bring important security risks. The vulnerabilities of LLMs are already a vivid topic of discussion [Perez et al., 2022], with applications of adversarial attacks to prompt injection and other threats. However, as a continuous modality, audio is subject to a wider diversity of attacks, as we have discussed throughout this thesis. Thus we believe that the importance of secure audio technology will grow in importance.

In that landscape, *adversarial robustness as we have studied it an important tool*, though not the only one. Perturbations in L_p norm are one of the best-defined, simpler-to-grasp settings, and almost a "toy problem" for more complex scenarios. It is unlikely that the research community can safely address those scenarios without designing adversarially robust ASR models first. This thesis lays the foundations for such work, and we have shown that the objective, while challenging, is not out of reach. Methods like adversarial training can be adapted to speech models, not just for ASR but for the pretraining of all tasks. The biggest challenge in this defense at this point is its heavy cost. We are confident that an application of our adversarial training framework to the massive pretraining of a large model, with multiple attack iteration steps, would lead to robust models with sufficient performance to be deployed. However, the computational cost of such an endeavor makes it unrealistic.

However, recent works in all modalities have shown a promising trend of *dropping training cost while matching the performance* of very strong models [Baevski et al., 2022a, Geiping and Goldstein, 2022, Leclerc et al., 2023]. We are hopeful that such progress in speech will make costly algorithms like adversarial training more accessible. Symmetrically, we believe that improvements in model performance, scaling, and noise robustness will continue to mitigate the performance toll of randomized smoothing, as we have shown that recent improvements have already done. One promising approach is the use of diffusion models on the audio modality. As some works on images have shown, those can both serve as data augmentation methods which adversarial training benefits from [Wang et al., 2023], and within randomized smoothing as denoisers [Carlini et al., 2023]. The latter application is still lagging on audio because the most effective diffusion models generate data in the spectrogram domain while adversarial attacks operate in the waveform domain. But work on AI models generating waveforms directly is growing and we expect this line of robustness approaches to gain relevance in the near future.

Finally, we would like to emphasize that, now and in the future, *the risks of audio models are not limited to adversarial attacks*s by any means. Privacy attacks, data poisoning, and deepfake generation are all massive threats that the audio community must continue to address. But as we have started to show in this work for privacy, many of those threats are linked to adversarial attacks. They are for instance a common way to inject imperceptible poison in data points without label access [Shafahi et al., 2018]. Deepfake detection methods are bypassed by attackers using adversarial perturbations [Gandhi and Jain, 2020]. Thus the importance of those many threats largely cements the importance of adversarial robustness as a goal for the research community.

[1]`https://github.com/AIGC-Audio/AudioGPT`

Bibliography

H. Abdullah, M. S. Rahman, W. Garcia, L. Blue, K. Warren, A. S. Yadav, T. Shrimpton, and P. Traynor. Hear "no evil", see "Kenansville": Efficient and Transferable Black-box Attacks on Speech Recognition and Voice Identification Systems. In *IEEE Symposium on Security and Privacy (IEEE S&P)*, 2021a.

H. Abdullah, K. Warren, V. Bindschaedler, N. Papernot, and P. Traynor. SoK: The Faults in our ASRs: An Overview of Attacks against Automatic Speech Recognition and Speaker Identification Systems. In *IEEE Symposium on Security and Privacy (IEEE S&P)*, 2021b.

H. Abdullah, A. Karlekar, V. Bindschaedler, and P. Traynor. Demystifying limited adversarial transferability in automatic speech recognition systems. In *International Conference on Learning Representations*, 2022. URL `https://openreview.net/forum?id=l5aSHXi8jG5`.

H. Abdullah, A. Karleka, S. Prasad, M. S. Rahman, L. Blue, L. A. Bauer, V. Bindschaelder, and P. Traynor. Attacks as Defenses: Designing Robust Audio CAPTCHAs Using Attacks on Automatic Speech Recognition Systems. In *NDSS*, 2023.

J.-B. Alayrac, J. Uesato, P.-S. Huang, A. Fawzi, R. Stanforth, and P. Kohli. Are labels required for improving adversarial robustness? In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL `https://proceedings.neurips.cc/paper/2019/file/bea6cfd50b4f5e3c735a972cf0eb8450-Paper.pdf`.

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

D. Amodei, S. Ananthanarayanan, R. Anubhai, J. Bai, E. Battenberg, C. Case, J. Casper, B. Catanzaro, Q. Cheng, G. Chen, J. Chen, J. Chen, Z. Chen, M. Chrzanowski, A. Coates, G. Diamos, K. Ding, N. Du, E. Elsen, J. Engel, W. Fang, L. Fan, C. Fougner, L. Gao, C. Gong, A. Hannun, T. Han, L. Johannes, B. Jiang, C. Ju, B. Jun, P. LeGresley, L. Lin, J. Liu, Y. Liu, W. Li, X. Li, D. Ma, S. Narang, A. Ng, S. Ozair, Y. Peng, R. Prenger, S. Qian, Z. Quan, J. Raiman, V. Rao, S. Satheesh, D. Seetapun, S. Sengupta, K. Srinet, A. Sriram, H. Tang, L. Tang, C. Wang, J. Wang, K. Wang, Y. Wang, Z. Wang, Z. Wang, S. Wu, L. Wei, B. Xiao, W. Xie, Y. Xie, D. Yogatama, B. Yuan, J. Zhan, and Z. Zhu. Deep speech 2 : End-to-end speech recognition in english and mandarin. In M. F. Balcan and K. Q. Weinberger, editors, *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine*

Learning Research, pages 173–182, New York, New York, USA, 20–22 Jun 2016. PMLR. URL http://proceedings.mlr.press/v48/amodei16.html.

M. Andriushchenko, F. Croce, N. Flammarion, and M. Hein. Square attack: A query-efficient black-box adversarial attack via random search. In A. Vedaldi, H. Bischof, T. Brox, and J.-M. Frahm, editors, *Computer Vision – ECCV 2020*, pages 484–501, Cham, 2020. Springer International Publishing. ISBN 978-3-030-58592-1.

R. Ardila, M. Branson, K. Davis, M. Henretty, M. Kohler, J. Meyer, R. Morais, L. Saunders, F. M. Tyers, and G. Weber. Common voice: A massively-multilingual speech corpus. In *LREC*, 2020.

A. Athalye, N. Carlini, and D. Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. In *ICML*, 2018a. URL https://arxiv.org/abs/1802.00420.

A. Athalye, L. Engstrom, A. Ilyas, and K. Kwok. Synthesizing robust adversarial examples. In J. Dy and A. Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 284–293. PMLR, 10–15 Jul 2018b. URL https://proceedings.mlr.press/v80/athalye18b.html.

M. Augustin, A. Meinke, and M. Hein. Adversarial robustness on in- and out-distribution improves explainability. In *ECCV*, 2020.

A. Baevski, Y. Zhou, A. Mohamed, and M. Auli. wav2vec 2.0: A framework for self-supervised learning of speech representations. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 12449–12460. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/file/92d1e1eb1cd6f9fba3227870bb6d7f07-Paper.pdf.

A. Baevski, W.-N. Hsu, A. CONNEAU, and M. Auli. Unsupervised speech recognition. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 27826–27839. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper_files/paper/2021/file/ea159dc9788ffac311592613b7f71fbb-Paper.pdf.

A. Baevski, A. Babu, W.-N. Hsu, and M. Auli. Efficient self-supervised learning with contextualized target representations for vision, speech and language, 2022a.

A. Baevski, W.-N. Hsu, Q. Xu, A. Babu, J. Gu, and M. Auli. data2vec: A general framework for self-supervised learning in speech, vision and language, 2022b. URL http://arxiv.org/abs/2202.03555. cite arxiv:2202.03555.

B. Biggio and F. Roli. Wild patterns: Ten years after the rise of adversarial machine learning. *Pattern Recognition*, 84:317–331, 2018. ISSN 0031-3203. doi: https://doi.org/10.1016/j.patcog.2018.07.023. URL https://www.sciencedirect.com/science/article/pii/S0031320318302565.

B. Biggio, G. Fumera, and F. Roli. Adversarial pattern classification using multiple classifiers and randomisation. In *SSPR/SPR*, 2008.

W. Brendel, J. Rauber, and M. Bethge. Decision-based adversarial attacks: Reliable attacks against black-box machine learning models. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018. URL https://openreview.net/forum?id=SyZI0GWCZ.

T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. M. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei. Language models are few-shot learners. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS'20, Red Hook, NY, USA, 2020. Curran Associates Inc. ISBN 9781713829546.

J. Buckman, A. Roy, C. Raffel, and I. Goodfellow. Thermometer encoding: One hot way to resist adversarial examples. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=S18Su--CW.

N. Carlini and D. Wagner. Adversarial examples are not easily detected: Bypassing ten detection methods. In *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*, AISec '17, page 3–14, New York, NY, USA, 2017. Association for Computing Machinery. ISBN 9781450352024. doi: 10.1145/3128572.3140444. URL https://doi.org/10.1145/3128572.3140444.

N. Carlini and D. A. Wagner. Towards evaluating the robustness of neural networks. *CoRR*, 2016.

N. Carlini and D. A. Wagner. Audio adversarial examples: Targeted attacks on speech-to-text. *2018 IEEE Security and Privacy Workshops (SPW)*, pages 1–7, 2018.

N. Carlini, A. Athalye, N. Papernot, W. Brendel, J. Rauber, D. Tsipras, I. Goodfellow, A. Madry, and A. Kurakin. On evaluating adversarial robustness. *arXiv preprint arXiv:1902.06705*, 2019.

N. Carlini, S. Chien, M. Nasr, S. Song, A. Terzis, and F. Tramer. Membership Inference Attacks from First Principles. In *2022 IEEE Symposium on Security and Privacy (SP)*, pages 1897–1914. IEEE, 2022.

N. Carlini, F. Tramer, K. D. Dvijotham, L. Rice, M. Sun, and J. Z. Kolter. (certified!!) adversarial robustness for free! In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=JLg5aHHv7j.

Y. Carmon, A. Raghunathan, L. Schmidt, P. Liang, and J. Duchi. Unlabeled data improves adversarial robustness. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

F. Carrara, R. Caldelli, F. Falchi, and G. Amato. On the robustness to adversarial examples of neural ode image classifiers. In *2019 IEEE International Workshop on Information Forensics and Security (WIFS)*, pages 1–6, 2019. doi: 10.1109/WIFS47025.2019.9035109.

A. Chakraborty, M. Alam, V. Dey, A. Chattopadhyay, and D. Mukhopadhyay. Adversarial attacks and defences: A survey. *CoRR*, abs/1810.00069, 2018. URL `http://arxiv.org/abs/1810.00069`.

W. Chan, N. Jaitly, Q. Le, and O. Vinyals. Listen, attend and spell: A neural network for large vocabulary conversational speech recognition. In *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4960–4964, 2016. doi: 10.1109/ICASSP.2016.7472621.

S. Chen, C. Wang, Z. Chen, Y. Wu, S. Liu, Z. Chen, J. Li, N. Kanda, T. Yoshioka, X. Xiao, J. Wu, L. Zhou, S. Ren, Y. Qian, Y. Qian, M. Zeng, and F. Wei. Wavlm: Large-scale self-supervised pre-training for full stack speech processing. *IEEE Journal of Selected Topics in Signal Processing*, 16:1505–1518, 2021.

S. Chen, Y. Wu, C. Wang, Z. Chen, Z. Chen, S. Liu, J. Wu, Y. Qian, F. Wei, J. Li, and X. Yu. Unispeech-sat: Universal speech representation learning with speaker aware pre-training, 2022.

C.-C. Chiu, J. Qin, Y. Zhang, J. Yu, and Y. Wu. Self-supervised learning with random-projection quantizer for speech recognition. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 3915–3924. PMLR, 17–23 Jul 2022. URL `https://proceedings.mlr.press/v162/chiu22a.html`.

C. A. Choquette-Choo, F. Tramer, N. Carlini, and N. Papernot. Label-only Membership Inference Attacks. In *International conference on machine learning*, pages 1964–1974. PMLR, 2021.

M. M. Cisse, Y. Adi, N. Neverova, and J. Keshet. Houdini: Fooling deep structured visual and speech recognition models with adversarial examples. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30, pages 6977–6987. Curran Associates, Inc., 2017. URL `https://proceedings.neurips.cc/paper/2017/file/d494020ff8ec181ef98ed97ac3f25453-Paper.pdf`.

J. M. Cohen, E. Rosenfeld, and J. Z. Kolter. Certified adversarial robustness via randomized smoothing. *CoRR*, 2019.

F. Croce and M. Hein. Minimally distorted adversarial examples with a fast adaptive boundary attack. In *ICML*, 2020a.

F. Croce and M. Hein. Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks. In *ICML*, 2020b.

F. Croce, M. Andriushchenko, V. Sehwag, E. Debenedetti, N. Flammarion, M. Chiang, P. Mittal, and M. Hein. Robustbench: a standardized adversarial robustness benchmark. In *NeurIPS Datasets and Benchmarks Track*, 2021. URL `https://openreview.net/forum?id=SSKZPJCt7B`.

N. Dalvi, P. Domingos, Mausam, S. Sanghai, and D. Verma. Adversarial classification. In *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '04, page 99–108, New York, NY, USA, 2004. Association for Computing Machinery. ISBN 1581138881. doi: 10.1145/1014052.1014066. URL `https://doi.org/10.1145/1014052.1014066`.

N. Das, M. Shanbhogue, S.-T. Chen, F. Hohman, S. Li, L. Chen, M. E. Kounavis, and D. H. Chau. Compression to the rescue: Defending from adversarial attacks across modalities. In *Project Showcase, Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD)*. ACM, 2018.

G. W. Ding, L. Wang, and X. Jin. AdverTorch v0.1: An adversarial robustness toolbox based on pytorch. *arXiv:1902.07623*, 2019.

L. Dong, S. Xu, and B. Xu. Speech-transformer: A no-recurrence sequence-to-sequence model for speech recognition. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5884–5888, 2018a. doi: 10.1109/ICASSP.2018.8462506.

Y. Dong, F. Liao, T. Pang, H. Su, J. Zhu, X. Hu, and J. Li. Boosting adversarial attacks with momentum. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9185–9193, Los Alamitos, CA, USA, jun 2018b. IEEE Computer Society. doi: 10.1109/CVPR.2018.00957. URL `https://doi.ieeecomputersociety.org/10.1109/CVPR.2018.00957`.

A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021. URL `https://openreview.net/forum?id=YicbFdNTTy`.

R. Ehlers. Formal verification of piece-wise linear feed-forward neural networks. In *ATVA*, 2017.

T. W. et al. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online, Oct. 2020. Association for Computational Linguistics. URL `https://www.aclweb.org/anthology/2020.emnlp-demos.6`.

K. Eykholt, I. Evtimov, E. Fernandes, B. Li, A. Rahmati, C. Xiao, A. Prakash, T. Kohno, and D. Song. Robust physical-world attacks on deep learning visual classification. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1625–1634, 2018. doi: 10.1109/CVPR.2018.00175.

L. Fan, S. Liu, P.-Y. Chen, G. Zhang, and C. Gan. When does contrastive learning preserve adversarial robustness from pretraining to finetuning? In M. Ranzato, A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 21480–21492. Curran Associates, Inc., 2021. URL `https://proceedings.neurips.cc/paper/2021/file/b36ed8a07e3cd80ee37138524690eca1-Paper.pdf`.

A. Fawzi, S.-M. Moosavi-Dezfooli, and P. Frossard. Robustness of classifiers: From adversarial to random noise. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, NIPS'16, page 1632–1640, Red Hook, NY, USA, 2016. Curran Associates Inc. ISBN 9781510838819.

R. Feinman, R. R. Curtin, S. Shintre, and A. B. Gardner. Detecting adversarial samples from artifacts. In *International Conference on Machine Learning*, 2017.

A. Galloway, A. Golubeva, T. Tanay, M. Moussa, and G. Taylor. Batch normalization is a cause of adversarial vulnerability. *ICML Workshop on Identifying and Understanding Deep Learning Phenomena*, 2019.

A. Gandhi and S. Jain. Adversarial perturbations fool deepfake detectors. *2020 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8, 2020.

T. Gehr, M. Mirman, D. Drachsler-Cohen, P. Tsankov, S. Chaudhuri, and M. Vechev. Ai2: Safety and robustness certification of neural networks with abstract interpretation. In *2018 IEEE Symposium on Security and Privacy (SP)*, pages 3–18, 2018. doi: 10.1109/SP.2018.00058.

J. Geiping and T. Goldstein. Cramming: Training a language model on a single gpu in one day, 2022.

A. Globerson and S. Roweis. Nightmare at test time: Robust learning by feature deletion. In *Proceedings of the 23rd International Conference on Machine Learning*, ICML '06, page 353–360, New York, NY, USA, 2006. Association for Computing Machinery. ISBN 1595933832. doi: 10.1145/1143844.1143889. URL `https://doi.org/10.1145/1143844.1143889`.

I. Goodfellow, J. Shlens, and C. Szegedy. Explaining and harnessing adversarial examples. In *International Conference on Learning Representations*, 2015. URL `http://arxiv.org/abs/1412.6572`.

S. Gowal, K. Dvijotham, R. Stanforth, R. Bunel, C. Qin, J. Uesato, R. Arandjelovic, T. A. Mann, and P. Kohli. On the effectiveness of interval bound propagation for training verifiably robust models. *CoRR*, 2018.

S. Gowal, C. Qin, J. Uesato, T. Mann, and P. Kohli. Uncovering the limits of adversarial training against norm-bounded adversarial examples. *arXiv preprint arXiv:2010.03593*, 2020. URL `https://arxiv.org/pdf/2010.03593`.

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*, ICML '06, page 369–376, New York, NY, USA, 2006. Association for Computing Machinery. ISBN 1595933832. doi: 10.1145/1143844.1143891. URL `https://doi.org/10.1145/1143844.1143891`.

A. Graves, A.-r. Mohamed, and G. Hinton. Speech recognition with deep recurrent neural networks. In *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 6645–6649, 2013. doi: 10.1109/ICASSP.2013.6638947.

A. Gulati, C.-C. Chiu, J. Qin, J. Yu, N. Parmar, R. Pang, S. Wang, W. Han, Y. Wu, Y. Zhang, and Z. Zhang, editors. *Conformer: Convolution-augmented Transformer for Speech Recognition*, 2020.

C. Guo, M. Rana, M. Cisse, and L. van der Maaten. Countering adversarial images using input transformations. In *International Conference on Learning Representations*, 2018. URL `https://openreview.net/forum?id=SyJ7ClWCb`.

X. Haihua, Z. Jie, and G. Wu. An efficient multistage rover method for automatic speech recognition. In *2009 IEEE International Conference on Multimedia and Expo*, pages 894–897, 2009. doi: 10.1109/ICME.2009.5202639.

M. Hassid, T. Remez, T. A. Nguyen, I. Gat, A. Conneau, F. Kreuk, J. Copet, A. Defossez, G. Synnaeve, E. Dupoux, R. Schwartz, and Y. Adi. Textually pretrained speech language models, 2023.

K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016. doi: 10.1109/CVPR.2016.90.

W. He, J. Wei, X. Chen, N. Carlini, and D. Song. Adversarial example defenses: Ensembles of weak defenses are not strong. In *Proceedings of the 11th USENIX Conference on Offensive Technologies*, WOOT'17, page 15, USA, 2017. USENIX Association.

Y. He, T. N. Sainath, R. Prabhavalkar, I. McGraw, R. Alvarez, D. Zhao, D. Rybach, A. Kannan, Y. Wu, R. Pang, Q. Liang, D. Bhatia, Y. Shangguan, B. Li, G. Pundak, K. C. Sim, T. Bagby, S.-y. Chang, K. Rao, and A. Gruenstein. Streaming end-to-end speech recognition for mobile devices. In *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6381–6385, 2019. doi: 10.1109/ICASSP.2019.8682336.

D. Hendrycks and T. Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. In *International Conference on Learning Representations*, 2019. URL `https://openreview.net/forum?id=HJz6tiCqYm`.

Y. Higuchi, N. Moritz, J. L. Roux, and T. Hori. Momentum pseudo-labeling for semi-supervised speech recognition. In *Interspeech*, 2021.

Y. Higuchi, B. Yan, S. Arora, T. Ogawa, T. Kobayashi, and S. Watanabe. BERT meets CTC: New formulation of end-to-end speech recognition with pre-trained masked language model. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 5486–5503, Abu Dhabi, United Arab Emirates, Dec. 2022. Association for Computational Linguistics. URL `https://aclanthology.org/2022.findings-emnlp.402`.

S. Hochreiter and J. Schmidhuber. Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780, 11 1997. ISSN 0899-7667. doi: 10.1162/neco.1997.9.8.1735. URL `https://doi.org/10.1162/neco.1997.9.8.1735`.

T. Hori, S. Watanabe, and J. Hershey. Joint CTC/attention decoding for end-to-end speech recognition. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 518–529, Vancouver, Canada, July 2017. Association for

Computational Linguistics. doi: 10.18653/v1/P17-1048. URL https://aclanthology.org/P17-1048.

W.-N. Hsu, B. Bolte, Y.-H. Tsai, K. Lakhotia, R. Salakhutdinov, and A. Mohamed. Hubert: Self-supervised speech representation learning by masked prediction of hidden units. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, PP:1–1, 10 2021. doi: 10.1109/TASLP.2021.3122291.

H. Hu, Z. Salcic, L. Sun, G. Dobbie, P. S. Yu, and X. Zhang. Membership Inference Attacks on Machine Learning: A survey. *ACM Computing Surveys (CSUR)*, 54(11s):1–37, 2022.

H. Huang, Y. Wang, S. M. Erfani, Q. Gu, J. Bailey, and X. Ma. Exploring architectural ingredients of adversarially robust deep neural networks. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan, editors, *Advances in Neural Information Processing Systems*, 2021. URL https://openreview.net/forum?id=OdklztJBBYH.

A. Ilyas, S. Santurkar, D. Tsipras, L. Engstrom, B. Tran, and A. Madry. Adversarial examples are not bugs, they are features. In H. M. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché Buc, E. B. Fox, and R. Garnett, editors, *NeurIPS*, pages 125–136, 2019.

S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In F. Bach and D. Blei, editors, *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 448–456, Lille, France, 07–09 Jul 2015. PMLR. URL https://proceedings.mlr.press/v37/ioffe15.html.

H. Iwamida, S. Katagiri, and E. McDermott. Speaker-independent large vocabulary word recognition using an lvq/hmm hybrid algorithm. In *[Proceedings] ICASSP 91: 1991 International Conference on Acoustics, Speech, and Signal Processing*, pages 553–556 vol.1, 1991. doi: 10.1109/ICASSP.1991.150399.

B. Jayaraman, L. Wang, K. Knipmeyer, Q. Gu, and D. Evans. Revisiting Membership Inference Under Realistic Assumptions. *Proceedings on Privacy Enhancing Technologies*, 2021(2), 2021.

Z. Jiang, T. Chen, T. Chen, and Z. Wang. Robust pre-training by adversarial contrastive learning. In H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 16199–16210. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/file/ba7e36c43aff315c00ec2b8625e3b719-Paper.pdf.

F. T. Johansen. A comparison of hybrid hmm architecture using global discriminating training. *Proceeding of Fourth International Conference on Spoken Language Processing. ICSLP '96*, 1: 498–501 vol.1, 1996.

M. Khoury and D. Hadfield-Menell. On the geometry of adversarial examples. *ArXiv*, abs/1811.00525, 2018.

M. Kim, J. Tack, and S. J. Hwang. Adversarial self-supervised contrastive learning. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 2983–2994. Curran Associates, Inc., 2020. URL `https://proceedings.neurips.cc/paper/2020/file/1f1baa5b8edac74eb4eaa329f14a0361-Paper.pdf`.

D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015. URL `http://arxiv.org/abs/1412.6980`.

A. Kolcz and C. H. Teo. Feature weighting for improved classifier robustness. In *CEAS 2009*, 2009.

J. Z. Kolter and E. Wong. Provable defenses against adversarial examples via the convex outer adversarial polytope. *CoRR*, 2017.

A. Krizhevsky, G. Hinton, et al. Learning multiple layers of features from tiny images. Technical report, Citeseer, 2009.

A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012.

G. Leclerc, A. Ilyas, L. Engstrom, S. M. Park, H. Salman, and A. Mądry. Ffcv: Accelerating training by removing data bottlenecks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12011–12020, June 2023.

Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 1998.

Y. LeCun, C. Cortes, and C. Burges. Mnist handwritten digit database. *ATT Labs*, 2010.

M. Lecuyer, V. Atlidakis, R. Geambasu, D. Hsu, and S. Jana. Certified robustness to adversarial examples with differential privacy. In *2019 IEEE Symposium on Security and Privacy (SP)*, pages 656–672, 2019. doi: 10.1109/SP.2019.00044.

M. Lécuyer, V. Atlidakis, R. Geambasu, D. J. Hsu, and S. S. Jana. Certified robustness to adversarial examples with differential privacy. *2019 IEEE Symposium on Security and Privacy (SP)*, pages 656–672, 2019.

E. Levin. Word recognition using hidden control neural architecture. In *International Conference on Acoustics, Speech, and Signal Processing*, pages 433–436 vol.1, 1990. doi: 10.1109/ICASSP.1990.115740.

Q. Lhoest, A. Villanova del Moral, Y. Jernite, A. Thakur, P. von Platen, S. Patil, J. Chaumond, M. Drame, J. Plu, L. Tunstall, J. Davison, M. Šaško, G. Chhablani, B. Malik, S. Brandeis, T. Le Scao, V. Sanh, C. Xu, N. Patry, A. McMillan-Major, P. Schmid, S. Gugger, C. Delangue, T. Matussière, L. Debut, S. Bekman, P. Cistac, T. Goehringer, V. Mustar, F. Lagunas, A. Rush, and T. Wolf. Datasets: A community library for natural language processing. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 175–184, Online and Punta Cana, Dominican Republic, Nov. 2021.

Association for Computational Linguistics. URL https://aclanthology.org/2021.emnlp-demo.21.

H. Li, S. J. Pan, S. Wang, and A. C. Kot. Domain generalization with adversarial feature learning. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5400–5409, 2018. doi: 10.1109/CVPR.2018.00566.

J. Li, S. Qu, X. Li, J. Szurley, J. Z. Kolter, and F. Metze. Adversarial music: Real world audio adversary against wake-word detection system. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32, pages 11931–11941. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper/2019/file/ebbdfea212e3a756a1fded7b35578525-Paper.pdf.

S. Li. Concise formulas for the area and volume of a hyperspherical cap. *Asian Journal of Mathematics & Statistics*, 4:66–70, 2011.

T. Likhomanenko, Q. Xu, J. Kahn, G. Synnaeve, and R. Collobert. slimipl : Language-model-free iterative pseudo-labeling. In *Interspeech*, 2020.

A. H. Liu, H. yi Lee, and L.-S. Lee. Adversarial training of end-to-end speech recognition using a criticizing language model. *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6176–6180, 2018.

W. Liu, Y. Wen, Z. Yu, M. Li, B. Raj, and L. Song. Sphereface: Deep hypersphere embedding for face recognition. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6738–6746, 2017.

D. Lowd and C. Meek. Adversarial learning. In *Proceedings of the Eleventh ACM SIGKDD International Conference on Knowledge Discovery in Data Mining*, KDD '05, page 641–647, New York, NY, USA, 2005. Association for Computing Machinery. ISBN 159593135X. doi: 10.1145/1081870.1081950. URL https://doi.org/10.1145/1081870.1081950.

Z. Lu, W. Han, Y. Zhang, and L. Cao. Exploring Targeted Universal Adversarial Perturbations to End-to-End ASR Models. In *Proc. Interspeech 2021*, pages 3460–3464, 2021. doi: 10.21437/Interspeech.2021-1668.

L. Lugosch, T. Likhomanenko, G. Synnaeve, and R. Collobert. Pseudo-labeling for massively multilingual speech recognition. *ICASSP*, 2022.

X. Ma, B. Li, Y. Wang, S. M. Erfani, S. Wijewickrema, G. Schoenebeck, M. E. Houle, D. Song, and J. Bailey. Characterizing adversarial subspaces using local intrinsic dimensionality. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=B1gJ1L2aW.

A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu. Towards deep learning models resistant to adversarial attacks. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018. URL https://openreview.net/forum?id=rJzIBfZAb.

K. Markert, D. Mirdita, and K. Böttinger. Language dependencies in adversarial attacks on speech recognition systems. In *2021 ISCA Symposium on Security and Privacy in Speech Communication*. ISCA, nov 2021. doi: 10.21437/spsc.2021-6. URL `https://doi.org/10.21437%2Fspsc.2021-6`.

E. Mendes and K. Hogan. Defending against imperceptible audio adversarial examples using proportional additive gaussian noise, 2020.

J. H. Metzen, T. Genewein, V. Fischer, and B. Bischoff. On detecting adversarial perturbations. In *International Conference on Learning Representations*, 2017. URL `https://openreview.net/forum?id=SJzCSf9xg`.

Y. Miao, M. Xue, C. Chen, L. Pan, J. Zhang, B. Z. H. Zhao, D. Kaafar, and Y. Xiang. The Audio Auditor: User-Level Membership Inference in Internet of Things Voice Services. In *Proc. Priv. Enhancing Technol.*, pages 209–228, 2021. doi: 10.2478/popets-2021-0012.

M. Mirman, A. Hägele, P. Bielik, T. Gehr, and M. Vechev. Robustness certification with generative models. In *Proceedings of the 42nd ACM SIGPLAN International Conference on Programming Language Design and Implementation*, PLDI 2021, page 1141–1154, New York, NY, USA, 2021. Association for Computing Machinery. ISBN 9781450383912. doi: 10.1145/3453483.3454100. URL `https://doi.org/10.1145/3453483.3454100`.

A. Mohamed, H. yi Lee, L. Borgholt, J. D. Havtorn, J. Edin, C. Igel, K. Kirchhoff, S.-W. Li, K. Livescu, L. Maaløe, T. N. Sainath, and S. Watanabe. Self-supervised speech representation learning: A review. *IEEE Journal of Selected Topics in Signal Processing*, 16:1179–1210, 2022.

S.-M. Moosavi-Dezfooli, A. Fawzi, and P. Frossard. Deepfool: a simple and accurate method to fool deep neural networks. In *CVPR*, 2016.

N. Narodytska and S. Kasiviswanathan. Simple black-box adversarial attacks on deep neural networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 1310–1318, 2017. doi: 10.1109/CVPRW.2017.172.

P. Neekhara, S. S. Hussain, P. Pandey, S. Dubnov, J. McAuley, and F. Koushanfar. Universal adversarial perturbations for speech recognition systems. In *INTERSPEECH*, 2019.

J. Neyman and E. S. Pearson. On the problem of the most efficient tests of statistical hypotheses. *Philosophical Transactions of the Royal Society of London. Series A, Containing Papers of a Mathematical or Physical Character*, 231:289–337, 1933. ISSN 02643952. URL `http://www.jstor.org/stable/91247`.

R. Olivier and B. Raj. Sequential randomized smoothing for adversarially robust speech recognition. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, Punta Cana, Dominican Republic, Nov. 2021. Association for Computational Linguistics.

R. Olivier and B. Raj. Recent improvements of asr models in the face of adversarial attacks. *Interspeech*, 2022.

R. Olivier and B. Raj. How many perturbations break this model? evaluating robustness beyond adversarial accuracy. In *Proceedings of the 40th International Conference on Machine Learning*, Proceedings of Machine Learning Research, Honolulu, Hawaii, USA, 23–29 Jul 2023a.

R. Olivier and B. Raj. Adversarial masked prediction for robust self-supervised speech models. *Under review*, 2023b.

R. Olivier and B. Raj. There is more than one kind of robustness: Fooling whisper with adversarial examples. *Interspeech*, 2023c. URL `https://arxiv.org/abs/2210.17316`.

R. Olivier, B. Raj, and M. Shah. High-frequency adversarial defense for speech and audio. In *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2995–2999, 2021. doi: 10.1109/ICASSP39728.2021.9414525.

R. Olivier, H. Abdullah, and B. Raj. The surprising vulnerability of self-supervised speech recognition models to transferable adversarial perturbations. *Under review*, 2023a.

R. Olivier, F. Teixeira, K. Pizzi, A. Abad, I. Trancoso, and B. Raj. Exploring loss-based features for membership inference on asr. *Under review*, 2023b.

S. Onn and I. Weissman. Generating uniform random vectors over a simplex with implications to the volume of a certain polytope and to multivariate extremes. *Annals of Operations Research*, 189(1):331–342, September 2011. doi: 10.1007/s10479-009-0567-7. URL `https://ideas.repec.org/a/spr/annopr/v189y2011i1p331-34210.1007-s10479-009-0567-7.html`.

M. Ott, S. Edunov, A. Baevski, A. Fan, S. Gross, N. Ng, D. Grangier, and M. Auli. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of NAACL-HLT 2019: Demonstrations*, 2019.

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)*, pages 5206–5210, 2015. doi: 10.1109/ICASSP.2015.7178964.

T. Pang, X. Yang, Y. Dong, K. Xu, J. Zhu, and H. Su. Boosting adversarial training with hypersphere embedding. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 7779–7792. Curran Associates, Inc., 2020. URL `https://proceedings.neurips.cc/paper/2020/file/5898d8095428ee310bf7fa3da1864ff7-Paper.pdf`.

N. Papernot, P. D. McDaniel, X. Wu, S. Jha, and A. Swami. Distillation as a defense to adversarial perturbations against deep neural networks. *CoRR*, 2015.

N. Papernot, P. Mcdaniel, S. Jha, M. Fredrikson, Z. B. Celik, and A. Swami. The limitations of deep learning in adversarial settings. *2016 IEEE European Symposium on Security and Privacy (EuroS&P)*, pages 372–387, 2016a.

N. Papernot, P. D. McDaniel, and I. J. Goodfellow. Transferability in machine learning: from phenomena to black-box attacks using adversarial samples. *CoRR*, 2016b.

N. Papernot, F. Faghri, N. Carlini, I. Goodfellow, R. Feinman, A. Kurakin, C. Xie, Y. Sharma, T. Brown, A. Roy, A. Matyasko, V. Behzadan, K. Hambardzumyan, Z. Zhang, Y.-L. Juang, Z. Li, R. Sheatsley, A. Garg, J. Uesato, W. Gierke, Y. Dong, D. Berthelot, P. Hendricks, J. Rauber, and R. Long. Technical report on the cleverhans v2.1.0 adversarial examples library. *arXiv preprint arXiv:1610.00768*, 2018.

D. S. Park, W. Chan, Y. Zhang, C.-C. Chiu, B. Zoph, E. D. Cubuk, and Q. V. Le. SpecAugment: A Simple Data Augmentation Method for Automatic Speech Recognition. In *Proc. Interspeech 2019*, pages 2613–2617, 2019. doi: 10.21437/Interspeech.2019-2680.

S. Pascual, A. Bonafonte, and J. Serrà. Segan: Speech enhancement generative adversarial network. In *INTERSPEECH*, 2017.

A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019. URL `http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf`.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

E. Perez, S. Huang, F. Song, T. Cai, R. Ring, J. Aslanides, A. Glaese, N. McAleese, and G. Irving. Red teaming language models with language models. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 3419–3448, Abu Dhabi, United Arab Emirates, Dec. 2022. Association for Computational Linguistics. URL `https://aclanthology.org/2022.emnlp-main.225`.

R. Pinot, L. Meunier, A. Araujo, H. Kashima, F. Yger, C. Gouy-Pailler, and J. Atif. Theoretical evidence for adversarial robustness through randomization. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL `https://proceedings.neurips.cc/paper/2019/file/36ab62655fa81ce8735ce7cfdaf7c9e8-Paper.pdf`.

Y. Qin, N. Carlini, G. Cottrell, I. Goodfellow, and C. Raffel. Imperceptible, robust, and targeted adversarial examples for automatic speech recognition. In K. Chaudhuri and R. Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 5231–5240. PMLR, 09–15 Jun 2019. URL `http://proceedings.mlr.press/v97/qin19a.html`.

L. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989. doi: 10.1109/5.18626.

R. Rade and S.-M. Moosavi. Helper-based adversarial training: Reducing excessive margin to achieve a better accuracy vs. robustness trade-off. In *ICML 2021 Workshop on Adversarial Machine Learning*, 2021. URL https://openreview.net/forum?id=BuD2LmNaU3a.

A. Radford, J. W. Kim, T. Xu, G. Brockman, C. McLeavey, and I. Sutskever. Robust speech recognition via large-scale weak supervision., 2022. URL https://cdn.openai.com/papers/whisper.pdf.

M. Raghu, T. Unterthiner, S. Kornblith, C. Zhang, and A. Dosovitskiy. Do vision transformers see like convolutional neural networks? In *Neural Information Processing Systems*, 2021.

J. Rauber, W. Brendel, and M. Bethge. Foolbox: A python toolbox to benchmark the robustness of machine learning models. In *Reliable Machine Learning in the Wild Workshop, 34th International Conference on Machine Learning*, 2017. URL http://arxiv.org/abs/1707.04131.

J. Rauber, R. Zimmermann, M. Bethge, and W. Brendel. Foolbox native: Fast adversarial attacks to benchmark the robustness of machine learning models in pytorch, tensorflow, and jax. *Journal of Open Source Software*, 5(53):2607, 2020. doi: 10.21105/joss.02607. URL https://doi.org/10.21105/joss.02607.

M. Ravanelli, T. Parcollet, P. Plantinga, A. Rouhe, S. Cornell, L. Lugosch, C. Subakan, N. Dawalatabad, A. Heba, J. Zhong, J.-C. Chou, S.-L. Yeh, S.-W. Fu, C.-F. Liao, E. Rastorgueva, F. Grondin, W. Aris, H. Na, Y. Gao, R. D. Mori, and Y. Bengio. SpeechBrain: A general-purpose speech toolkit, 2021. arXiv:2106.04624.

S.-A. Rebuffi, S. Gowal, D. A. Calian, F. Stimberg, O. Wiles, and T. Mann. Data augmentation can improve robustness. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan, editors, *Advances in Neural Information Processing Systems*, 2021. URL https://openreview.net/forum?id=kgVJBBThdSZ.

L. Rice, E. Wong, and J. Z. Kolter. Overfitting in adversarially robust deep learning. In *ICML*, 2020.

L. Rice, A. Bair, H. Zhang, and J. Z. Kolter. Robustness between the worst and average case. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 27840–27851. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper/2021/file/ea4c796cccfc3899b5f9ae2874237c20-Paper.pdf.

A. Robey, L. Chamon, G. J. Pappas, and H. Hassani. Probabilistically robust learning: Balancing average and worst-case performance. In *International Conference on Machine Learning*, pages 18667–18686. PMLR, 2022.

S. Rosenthal, M. A. Bornea, and A. Sil. Are multilingual bert models robust? a case study on adversarial attacks for multilingual question answering. *ArXiv*, abs/2104.07646, 2021.

H. Salman, G. Yang, H. Zhang, C.-J. Hsieh, and P. Zhang. A convex relaxation barrier to tight robustness verification of neural networks. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL `https://proceedings.neurips.cc/paper/2019/file/246a3c5544feb054f3ea718f61adfa16-Paper.pdf`.

H. Salman, A. Ilyas, L. Engstrom, A. Kapoor, and A. Madry. Do adversarially robust imagenet models transfer better? In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 3533–3545. Curran Associates, Inc., 2020a. URL `https://proceedings.neurips.cc/paper/2020/file/24357dd085d2c4b1a88a7e0692e60294-Paper.pdf`.

H. Salman, M. Sun, G. Yang, A. Kapoor, and J. Z. Kolter. Denoised smoothing: A provable defense for pretrained classifiers. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 21945–21957. Curran Associates, Inc., 2020b. URL `https://proceedings.neurips.cc/paper/2020/file/f9fd2624beefbc7808e4e405d73f57ab-Paper.pdf`.

P. Samangouei, M. Kabkab, and R. Chellappa. Defense-gan: Protecting classifiers against adversarial attacks using generative models. In *ICLR*, 2018.

P. Scalart and J. Filho. Speech enhancement based on a priori signal to noise estimation. *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, 2: 629 – 632 vol. 2, 06 1996. doi: 10.1109/ICASSP.1996.543199.

L. Schmidt, S. Santurkar, D. Tsipras, K. Talwar, and A. Madry. Adversarially robust generalization requires more data. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, NIPS'18, page 5019–5031, Red Hook, NY, USA, 2018. Curran Associates Inc.

L. Schönherr, K. Kohls, S. Zeiler, T. Holz, and D. Kolossa. Adversarial attacks against automatic speech recognition systems via psychoacoustic hiding. In *Network and Distributed System Security Symposium (NDSS)*, 2019.

L. Schönherr, T. Eisenhofer, S. Zeiler, T. Holz, and D. Kolossa. Imperio: Robust over-the-air adversarial examples for automatic speech recognition systems. In *Annual Computer Security Applications Conference*, ACSAC '20, page 843–855, New York, NY, USA, 2020. Association for Computing Machinery. ISBN 9781450388580. doi: 10.1145/3427228.3427276. URL `https://doi.org/10.1145/3427228.3427276`.

V. Sehwag, S. Mahloujifar, T. Handina, S. Dai, C. Xiang, M. Chiang, and P. Mittal. Robust learning meets generative models: Can proxy distributions improve adversarial robustness? *ICLR Workshop on Security and Safety in Machine Learning Systems*, 2021. URL `https://arxiv.org/abs/2104.09425`.

A. Shafahi, W. R. Huang, M. Najibi, O. Suciu, C. Studer, T. Dumitras, and T. Goldstein. Poison frogs! targeted clean-label poisoning attacks on neural networks. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, NIPS'18, page 6106–6116, Red Hook, NY, USA, 2018. Curran Associates Inc.

M. Shah, R. Olivier, and B. Raj. Uncovering the robustness potential of neural architectures by measuring the probability of high adversarial accuracy. *Under review*, 2022.

M. A. Shah, R. Olivier, and B. Raj. Towards adversarial robustness via compact feature representations. In *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3845–3849, 2021a. doi: 10.1109/ICASSP39728.2021.9414696.

M. A. Shah, R. Olivier, and B. Raj. Exploiting non-linear redundancy for neural model compression. In *2020 25th International Conference on Pattern Recognition (ICPR)*, pages 9928–9935, 2021b. doi: 10.1109/ICPR48806.2021.9413178.

M. A. Shah, J. Szurley, M. Mueller, A. Mouchtaris, and J. Droppo. Evaluating the Vulnerability of End-to-End Automatic Speech Recognition Models to Membership Inference Attacks. In *Proc. Interspeech 2021*, pages 891–895, 2021c. doi: 10.21437/Interspeech.2021-1188.

R. Shao, Z. Shi, J. Yi, P. Chen, and C. Hsieh. On the adversarial robustness of visual transformers. *CoRR*, abs/2103.15670, 2021. URL `https://arxiv.org/abs/2103.15670`.

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*, CCS '16, page 1528–1540, New York, NY, USA, 2016. Association for Computing Machinery. ISBN 9781450341394. doi: 10.1145/2976749.2978392. URL `https://doi.org/10.1145/2976749.2978392`.

R. Shin and D. Song. Jpeg-resistant adversarial images. In *NIPS 2017 Workshop on Machine Learning and Computer Security*, 2017.

R. Shokri, M. Stronati, C. Song, and V. Shmatikov. Membership Inference Attacks against Machine Learning Models. In *2017 IEEE symposium on security and privacy (SP)*, pages 3–18. IEEE, 2017.

K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv:1409.1556*, 2014.

Y. Song, T. Kim, S. Nowozin, S. Ermon, and N. Kushman. Pixeldefend: Leveraging generative models to understand and defend against adversarial examples. In *International Conference on Learning Representations*, 2018.

N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(56): 1929–1958, 2014. URL `http://jmlr.org/papers/v15/srivastava14a.html`.

R. K. Srivastava, K. Greff, and J. Schmidhuber. Highway networks. *CoRR*, abs/1505.00387, 2015. URL `http://arxiv.org/abs/1505.00387`.

J. Su, D. V. Vargas, and K. Sakurai. One pixel attack for fooling deep neural networks. *IEEE Transactions on Evolutionary Computation*, 23:828–841, 2019.

Subramanian, E. Benetos, M. Sandler, and Events. Robustness of adversarial attacks in sound event classification. In *Workshop on Detection and Classification of Acoustic Scenes and Events*, 2019.

S. Sun, C. feng Yeh, M. Ostendorf, M.-Y. Hwang, and L. Xie. Training augmentation with adversarial examples for robust speech recognition. In *INTERSPEECH*, 2018.

G. Synnaeve, Q. Xu, J. Kahn, T. Likhomanenko, E. Grave, V. Pratap, A. Sriram, V. Liptchinsky, and R. Collobert. End-to-end ASR: from supervised to semi-supervised learning with modern architectures. In *ICML 2020 Workshop on Self-supervision in Audio and Speech*, 2020. URL `https://openreview.net/forum?id=OSVxDDc360z`.

C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus. Intriguing properties of neural networks. In *ICLR*, 2014. URL `http://arxiv.org/abs/1312.6199`.

J. Szurley and J. Z. Kolter. Perceptual based adversarial audio attacks. *ArXiv*, abs/1906.06355, 2019.

R. Taori, A. Kamsetty, B. Chu, and N. Vemuri. Targeted adversarial examples for black box audio systems. In *2019 IEEE Security and Privacy Workshops (SPW)*, pages 15–20, 2019. doi: 10.1109/SPW.2019.00016.

V. Tjeng, K. Y. Xiao, and R. Tedrake. Evaluating robustness of neural networks with mixed integer programming. In *International Conference on Learning Representations*, 2019. URL `https://openreview.net/forum?id=HyGIdiRqtm`.

H. Touvron, T. Lavril, G. Izacard, X. Martinet, M.-A. Lachaux, T. Lacroix, B. Rozière, N. Goyal, E. Hambro, F. Azhar, A. Rodriguez, A. Joulin, E. Grave, and G. Lample. Llama: Open and efficient foundation language models, 2023.

F. Tramèr, N. Papernot, I. Goodfellow, D. Boneh, and P. Mcdaniel. The space of transferable adversarial examples. *ArXiv*, abs/1704.03453, 2017.

F. Tramer, N. Carlini, W. Brendel, and A. Madry. On adaptive attacks to adversarial example defenses. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 1633–1645. Curran Associates, Inc., 2020. URL `https://proceedings.neurips.cc/paper/2020/file/11f38f8ecd71867b42433548d1078e38-Paper.pdf`.

W.-C. Tseng, W.-T. Kao, and H. yi Lee. Membership Inference Attacks Against Self-supervised Speech Models. In *Proc. Interspeech 2022*, pages 5040–5044, 2022. doi: 10.21437/Interspeech.2022-11245.

J. Uesato, B. O'Donoghue, P. Kohli, and A. van den Oord. Adversarial risk and the dangers of evaluating against weak attacks. In J. Dy and A. Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 5025–5034. PMLR, 10–15 Jul 2018a. URL `https://proceedings.mlr.press/v80/uesato18a.html`.

J. Uesato, B. O'Donoghue, P. Kohli, and A. van den Oord. Adversarial risk and the dangers of evaluating against weak attacks. In J. Dy and A. Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 5025–5034. PMLR, 10–15 Jul 2018b. URL `https://proceedings.mlr.press/v80/uesato18a.html`.

A. van den Oord, O. Vinyals, and k. kavukcuoglu. Neural discrete representation learning. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL `https://proceedings.neurips.cc/paper_files/paper/2017/file/7a98af17e63a0ac09ce2e96d03992fbc-Paper.pdf`.

A. van den Oord, Y. Li, and O. Vinyals. Representation learning with contrastive predictive coding. *ArXiv*, abs/1807.03748, 2018.

A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL `https://proceedings.neurips.cc/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf`.

C. Wang, Y. Tang, X. Ma, A. Wu, D. Okhonko, and J. Pino. fairseq s2t: Fast speech-to-text modeling with fairseq. In *Proceedings of the 2020 Conference of the Asian Chapter of the Association for Computational Linguistics (AACL): System Demonstrations*, 2020.

C. Wang, Y. Wu, Y. Qian, K. Kumatani, S. Liu, F. Wei, M. Zeng, and X. Huang. Unispeech: Unified speech representation learning with labeled and unlabeled data. In M. Meila and T. Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pages 10937–10947. PMLR, 2021. URL `http://proceedings.mlr.press/v139/wang21y.html`.

G. Wang and K. C. Sim. Sequential classification criteria for nns in automatic speech recognition. In *INTERSPEECH*, 2011.

H. Wang, X. Wu, P. Yin, and E. P. Xing. High frequency component helps explain the generalization of convolutional neural networks. *CoRR*, 2019.

J. Wang and H. Zhang. Bilateral adversarial training: Towards fast training of more robust models against adversarial attacks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019.

Z. Wang, T. Pang, C. Du, M. Lin, W. Liu, and S. Yan. Better diffusion models further improve adversarial training. In *International Conference on Machine Learning (ICML)*, 2023.

P. Warden. Speech commands: A dataset for limited-vocabulary speech recognition. *CoRR*, abs/1804.03209, 2018. URL `http://arxiv.org/abs/1804.03209`.

G. L. Wittel and S. F. Wu. On attacking statistical spam filters. In *CEAS*, 2004.

E. Wong, F. R. Schmidt, and J. Z. Kolter. Wasserstein adversarial examples via projected sinkhorn iterations. In K. Chaudhuri and R. Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pages 6808–6817. PMLR, 2019. URL `http://proceedings.mlr.press/v97/wong19a.html`.

E. Wong, L. Rice, and J. Z. Kolter. Fast is better than free: Revisiting adversarial training. In *International Conference on Learning Representations*, 2020. URL `https://openreview.net/forum?id=BJx040EFvH`.

H. Wu, B. Zheng, X. Li, X. Wu, H. Lee, and H. Meng. Characterizing the adversarial vulnerability of speech self-supervised learning. In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2022, Virtual and Singapore, 23-27 May 2022*, pages 3164–3168. IEEE, 2022. doi: 10.1109/ICASSP43922.2022.9747242. URL `https://doi.org/10.1109/ICASSP43922.2022.9747242`.

C. Xie and A. Yuille. Intriguing properties of adversarial training at scale. In *ICLR 2020 : Eighth International Conference on Learning Representations*, 2020.

C. Xie, J. Wang, Z. Zhang, Z. Ren, and A. Yuille. Mitigating adversarial effects through randomization. In *International Conference on Learning Representations*, 2018. URL `https://openreview.net/forum?id=Sk9yuql0Z`.

C. Xie, M. Tan, B. Gong, J. Wang, A. L. Yuille, and Q. V. Le. Adversarial examples improve image recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

Y. Xie, Z. Li, C. Shi, J. Liu, Y. Chen, and B. Yuan. Enabling fast and universal audio adversarial attack using generative model. In *AAAI*, 2021.

W. Xu, D. Evans, and Y. Qi. Feature squeezing: Detecting adversarial examples in deep neural networks. In *25th Annual Network and Distributed System Security Symposium, NDSS 2018, San Diego, California, USA, February 18-21, 2018*. The Internet Society, 2018. URL `http://wp.internetsociety.org/ndss/wp-content/uploads/sites/25/2018/02/ndss2018_03A-4_Xu_paper.pdf`.

H. Yakura and J. Sakuma. Robust audio adversarial example for a physical attack. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*, pages 5334–5341. International Joint Conferences on Artificial Intelligence Organization, 7 2019. doi: 10.24963/ijcai.2019/741. URL `https://doi.org/10.24963/ijcai.2019/741`.

Z. Yang, P. Chen, B. Li, and D. Song. Characterizing audio adversarial examples using temporal dependency. In *7th International Conference on Learning Representations, ICLR 2019*, 2019.

X. Yuan, Y. Chen, Y. Zhao, Y. Long, X. Liu, K. Chen, S. Zhang, H. Huang, X. Wang, and C. A. Gunter. Commandersong: A systematic approach for practical adversarial voice recognition.

In *27th USENIX Security Symposium (USENIX Security 18)*, pages 49–64, Baltimore, MD, Aug. 2018. USENIX Association. ISBN 978-1-939133-04-5. URL `https://www.usenix.org/conference/usenixsecurity18/presentation/yuan-xuejing`.

S. Yun, D. Han, S. J. Oh, S. Chun, J. Choe, and Y. J. Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 6022–6031, 2019.

R. Zhai, T. Cai, D. He, C. Dan, K. He, J. E. Hopcroft, and L. Wang. Adversarially robust generalization just requires more unlabeled data. *CoRR*, abs/1906.00555, 2019. URL `http://arxiv.org/abs/1906.00555`.

G. Zhang, C. Yan, X. Ji, T. Zhang, T. Zhang, and W. Xu. Dolphinattack: Inaudible voice commands. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, CCS '17, page 103–117, New York, NY, USA, 2017. Association for Computing Machinery. ISBN 9781450349468. doi: 10.1145/3133956.3134052. URL `https://doi.org/10.1145/3133956.3134052`.

H. Zhang, Y. Yu, J. Jiao, E. Xing, L. E. Ghaoui, and M. Jordan. Theoretically principled trade-off between robustness and accuracy. In K. Chaudhuri and R. Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 7472–7482. PMLR, 09–15 Jun 2019. URL `https://proceedings.mlr.press/v97/zhang19p.html`.

Z. Zhang, L. Yu Zhang, X. Zheng, B. Hussain Abbasi, and S. Hu. Evaluating Membership Inference Through Adversarial Robustness. *The Computer Journal*, 65(11):2969–2978, 2022.

Y. Zhou, Z. Jorgensen, and M. Inge. Combating good word attacks on statistical spam filters with multiple instance learning. In *19th IEEE International Conference on Tools with Artificial Intelligence(ICTAI 2007)*, volume 2, pages 298–305, 2007. doi: 10.1109/ICTAI.2007.120.

P. Żelasko, S. Joshi, Y. Shao, J. Villalba, J. Trmal, N. Dehak, and S. Khudanpur. Adversarial attacks and defenses for speech recognition systems, 2021.

Appendices

Appendix A

Supplementary material for chapter 3

A.1 Influence of hyperparameters on robust classifiers

Given a number m we define $F_{a_1,a_2,\dots,a_m}$ as the set of n-hidden-layer binary neural classifiers, that have a_1 neurons on the first hidden layer, a_2 on the second, etc. The final layer always has one neuron for binary classification. When there is no ambiguity on f, we name $h_{i,j}$ the j^{th} neuron of the i^{th} layer. We name $h'_{i,j}$ the pre-activation neuron (which is an affine function).

We can establish the existence or non-existence of valid solutions on $\mathcal{X}_\epsilon$ for certain values of ϵ, n, m, a_i, and certain activation functions.

Proposition 3.1. *For $n = 2$, with threshold activations, $F_{2,2}$ contains a classifier that is valid on $\mathcal{X}_\epsilon$ for all $0 < \epsilon < 1$.*

Proof. We exhibit an $f_r \in F_{2,2}$. We set:

$$h_{1,1} = (x_1 \geq 0) \ , \ h_{1,2} = (x_2 \geq 0) \tag{A.1}$$

$$h_{2,1} = h_{1,1} \wedge h_{1,2} = (h_{1,1} + h_{1,2} - 2 \geq 0) \tag{A.2}$$

$$h_{2,2} = \neg h_{1,1} \wedge \neg h_{1,2} = (-h_{1,1} - h_{1,2} \geq 0) \tag{A.3}$$

$$h_{3,1} = h_{2,1} \vee h_{2,2} = (h_{2,1} + h_{2,2} - 1 \geq 0) \tag{A.4}$$

In other words, h_1 turns the RXOR problem into the traditional BXOR problem, which the two-layer network h_2, h_3 can classically solve. Since h_1 sends (x_1, x_2) onto the associated vertex $(\pm 1, \pm 1)$, any point in $\mathcal{X}_\epsilon$ with $0 < \epsilon < 1$ is correctly classified by f_r.

□

It is obvious that any $F_{a_1,a_2,\dots,a_m}$ with $m \geq 2$ and a_i also contain classifier f_r, since layers can easily ignore some neurons or compute the identity function. On the other hand, results change if we drop one layer.

Proposition 3.2. *For $n = 2$, with threshold activations, given $0 < \epsilon_1 < \frac{2}{3}$ and $\frac{2}{3} < \epsilon_2 < 1$, F_2 contains a valid classifier on $\mathcal{X}_{\epsilon_1}$ but not on $\mathcal{X}_{\epsilon_2}$*

Proof. In F_2 the same h_1 as above cannot lead to a valid classifier: the first layer must project the dataset onto a linearly separable dataset, which binary XOR isn't. In fact, the only non-trivial

h_2	x, y	$-1, -1$	$-1, 1$	$1, -1$	$1, 1$
AND	$h_{1,1}$	0	1	1	1
	$h_{1,1}$	1	1	1	0
	$f(=$ XOR$)$	0	1	1	0
OR	$h_{1,1}$	0	1	0	0
	$h_{1,1}$	0	0	1	0
	$f =$ (XOR)	0	1	1	0

Table A.1: The necessary truth tables of $h_{1,1}$ and $h_{1,2}$ when $h2$ is either an AND gate or an OR gate

gates h_2 can modelize are AND ($a + b \geq 2$) or OR ($a + b \geq 1$), applied to the literals $(\neg)h_{1,1}$ and $(\neg)h_{1,2}$. Without loss of generality we can assume that $h_2 = h_{1,1} \vee h_{1,2}$ or $h_2 = h_{1,1} \wedge h_{1,2}$.

f must correctly classify vertices $(\pm 1, \pm 1)$. We can therefore reverse engineer the truth tables of $h_{1,1}$ and $h_{1,2}$ on each vertex, depending on what h_2 does.

- If h_2 is AND then we must have $h_{1,j}(-1, 1) = h_{1,j}(1, -1) = 1$ for $j \in \{1, 2\}$. For at least one j we have $h_{1,j}(-1, -1) = 0$: we assume it is $j = 1$. By affinity $h'_{1,1}(1, 1) = h'_{1,1}(-1, 1) + h'_{1,1}(1, -1) - h'_{1,1}(-1, -1) \geq 0$ as it is the sum of three positive terms: therefore $h_{1,1}(1, 1) = 1$. For at least one j we have $h_{1,j}(1, 1) = 0$: it must be is $j = 2$. Similarly we can infer that $h_{1,2}(-1, -1) = 1$. We write these truth tables in Table A.1.

- If h_2 is OR, we can apply a similar reasoning and figure out the truth tables of $h_{1,1}$ and $h_{1,2}$, which we also report in Table A.1.

We note that the $h'_{1,j}$ must keep a constant sign over each connected component of $\mathcal{X}_\epsilon$ (regions $[-1, \epsilon - 1] \times [-1, \epsilon - 1]$, $[-1, \epsilon - 1] \times [1 - \epsilon, 1]$, etc.), as any permissible change of activation for a single hidden neuron changes the final output. Respectively, if these conditions and the truth tables above are met for $h_{1,1}$ and $h_{1,2}$ then f is valid. In the h_2-OR case, writing

$$h_{1,1}(x, y) = ax + by \geq 1 \tag{A.5}$$

the conditions on $h_{1,1}$ are equivalent to the following system of inequations:

$$\begin{cases} -a - (1 - \epsilon)b < 1 \\ (1 - \epsilon)a + b < 1 \\ -(1 - \epsilon)a + (1 - \epsilon)b \geq 1 \end{cases}$$

The existence of a solution depending on ϵ can be solved easily using linear programming. We find that the (a, b) satisfying the equality case for the first two inequations are $a == \frac{1}{\epsilon}, b = \frac{1}{\epsilon}$. On this corner point, inequation 3 becomes

$$\frac{2 - 2\epsilon}{\epsilon} \geq 1 \tag{A.6}$$

Therefore F_2 contains a valid solution on $\mathcal{X}_\epsilon$ iff $\epsilon < \frac{2}{3}$.

$\square$

To an extent, a similar reasoning could be repeated with F_k for $k > 2$. However, if that width k becomes extremely large, it is possible for the network to become arbitrarily close to f_r. This is because arbitrarily wide two-layer networks are universal approximators. Although we did not derive it, we expect that the maximal value of ϵ would slowly increase from $\frac{2}{3}$ to 1 when increasing the width.

These very wide networks aside, two-layer networks are insufficient to achieve robust classification on XOR with threshold activations. This changes when picking the ReLU activation $\text{ReLU}(x) = x^+$:

Proposition 3.3. *For $n = 2$, with ReLU activations, F_3 contains a classifier that is valid on $\mathcal{X}_\epsilon$ for all* $0 < \epsilon < 1$.

Proof. It can be verified that with

$$h_{1,1} = x_1^+ \ , \ h_{1,2} = x_2^+ \ , \ h_{1,3} = (x_1 + x_2)^+ \tag{A.7}$$

$$h_{2,1} = (h_{1,1} + h_{1,2} - h_{1,3} \geq 0) \tag{A.8}$$

then $f = f_r$ □

The key difference with threshold cases is the continuity of ReLU. It makes it possible for a first-layer pre-activation neuron to change sign within a connected component like $[-1, \epsilon - 1] \times [1 - \epsilon, 1]$, as $h'_{1,3}$ does, while having a valid classifier.

A.2 Additional experiments with accidental robustness

A.2.1 Influence of Modeling Choices on Probability of Robustness

This section completes section 3.5.3 by analyzing the influence of more hyperparameters on robustness. We plot additional results in Figure A.1 using the same format as Figure 3.3

Activation Function

To determine the impact of the activation function on the probability of arriving at a robust solution we take the widest model architectures from 3.5.3, set the activation function to one of ReLU, Tanh or Sigmoid, and compute AUC_{ϵ_j} and $\text{AUC}^*_{\epsilon_j}$. We also add biases to these models because for some of the activation functions the models can not learn RXOR without affine computations. Figure A.1a shows that for all ϵ_j, AUC_{ϵ_j} and $\text{AUC}^*_{\epsilon_j}$ are maximal when ReLU activations are used.

Batch Normalization

To observe the effect of batch normalization, we take MLP-9-ReLU and add a batch normalization layer after the linear layer and before the ReLU to obtain MLP-9-ReLUBN. Figure A.1b shows AUC_{ϵ_j} and $\text{AUC}^*_{\epsilon_j}$ for the two models. We observe that the addition of batch normalization has reduced both AUC_{ϵ_j} and $\text{AUC}^*_{\epsilon_j}$ meaning that the introduction of batch normalization has made both robust and valid solutions less accessible. This observation conflicts with our expectation since batch normalization is known to improve accuracy if training and testing data are identically distributed, but, perhaps this effect is due to the simplistic nature of the RXOR problem. However, it is in line with our expectation and existing literature [Galloway et al., 2019] that batch normalization reduces the likelihood of finding a robust solution.

Dropout

To observe the effect of dropout, we take MLP-9-ReLU and apply dropout with probability p after the linear layer and before the ReLU to obtain MLP-9-ReLUDropout$p$, where $p \in \{0.3, 0.6\}$. From Figure A.1c we see that the addition of dropout with $p = 0.3$ increases AUC_{ϵ_j} but reduces $\text{AUC}^*_{\epsilon_j}$. This indicates that regularization provided dropout prevents the training algorithm from fitting the decision boundary too close to the training data. Increasing p to 0.6, however, leads to a reduction in both AUC_{ϵ_j} and $\text{AUC}^*_{\epsilon_j}$, which indicates that this level of regularization diminishes the model's capacity to model the data.

Skip Connections

To test the impact of skip connections we take a two-layer model with 9 ReLU units in each layer (MLP-9_9-ReLU) and add a skip connection from the output of the first layer to the output of the second layer (MLP-9_9-ReLU-wSkip_1>2). Our expectation is that the addition of skip connections would increase the likelihood of finding robust models compared to MLP-9_9-ReLU because the training algorithm can set the weights and biases in the second layer to zero and recover a single layer model with 9 hidden units, which, based on the above results, should be more robust than a two-layer model. Looking at Figure A.1d we note that it is indeed the case that skip-connections improve AUC_{ϵ_j} and, to a lesser extent, $\text{AUC}^*_{\epsilon_j}$.

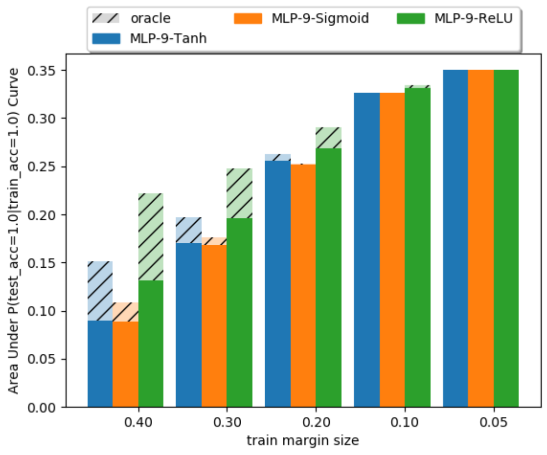
A.2.2 Generalization to More Complex Data and Models

This section extends the results of Section 3.5.4 with additional hyperparameters and prolongs the comparisons between 2D RXOR and MNIST models.

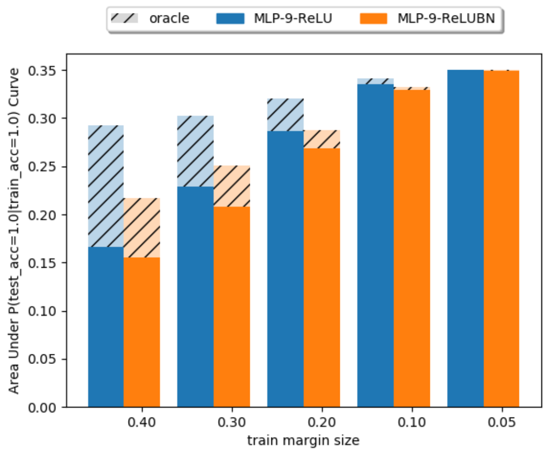
Trends in MLPs

In Figure A.2 we plot the results of our experiments on MLP using the same hyperparameters as Section A.2.1.

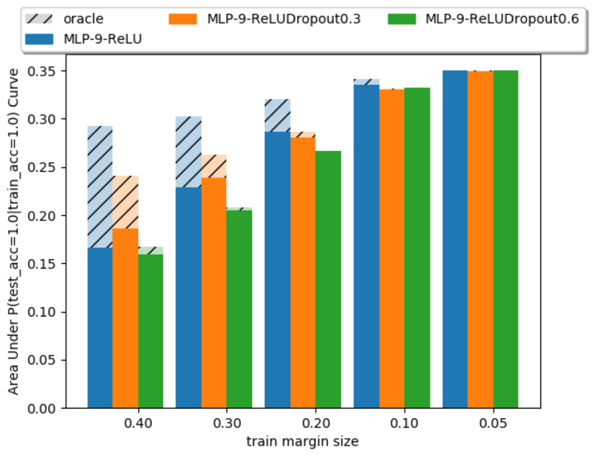
As discussed in section 3.5.4 the general trends on width and depth remain consistent between models trained on 2D RXOR and MNIST. However, as we can see in Figure A.2 there are some differences that are worth noting. Firstly, in the case of 2D RXOR models that used the ReLU activation performed better (see Section A.2.1) whereas we see from Figure A.2a that MLPs with Sigmoid units improve the robustness potential of the models trained on MNIST. This observation prevents us from making any claims about the relationship between robustness potential and activation function, rather we posit that certain activation functions are better suited to certain models and datasets. Secondly, we note from Figure A.2b that adding batch normalization to a model increases $\text{AUC}^*_{\epsilon_j}$ but decreases AUC_{ϵ_j}. This observation is consistent with our expectation and existing literature since it is common to use batch normalization to increase the classification



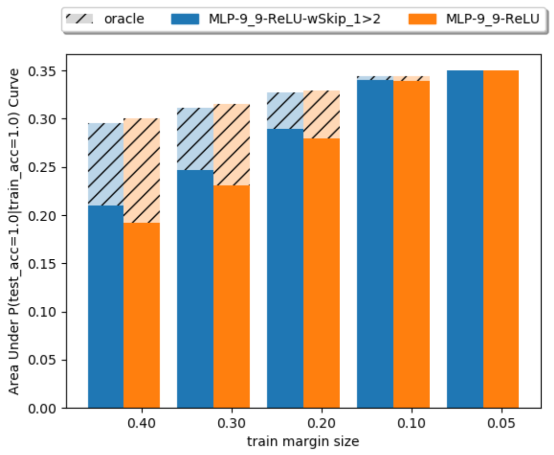
(a)



(b)

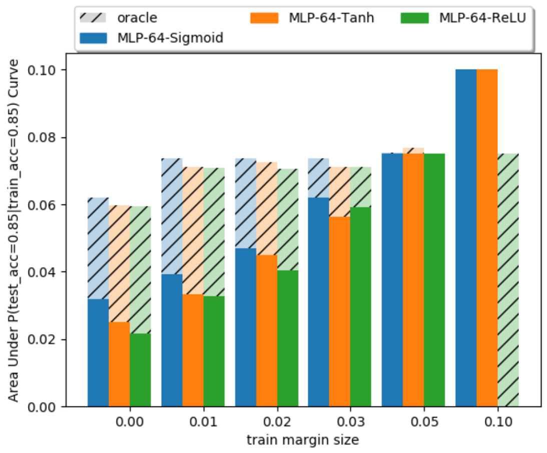


(c)

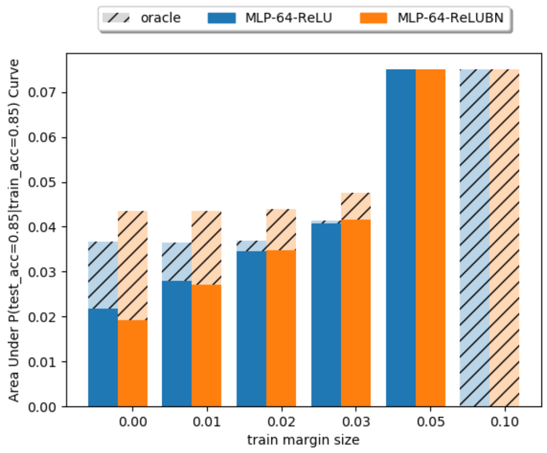


(d)

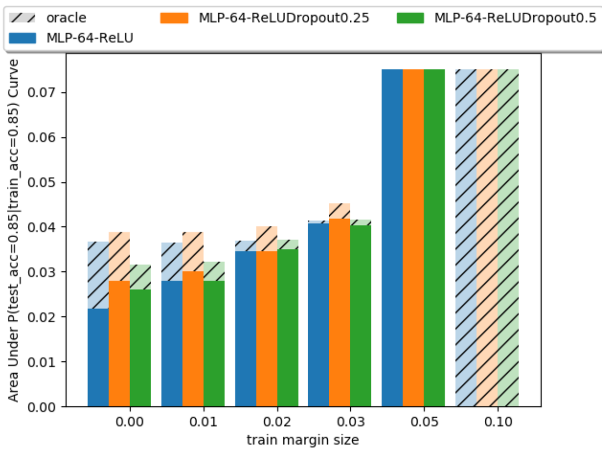
Figure A.1: The influence of various modeling choices on $P(\text{robust}|\epsilon_i, \epsilon_j)$ as measured by AUC_{ϵ_j} and $\text{AUC}^*_{\epsilon_j}$ for MLPs trained on 2D RXOR data. Subfigure (a) shows the effect of changing the activation function, (b) shows the impact of adding batch normalization, (c) shows the impact of adding dropout, and (d) shows the impact of adding skip connections in a 2-layer MLP.



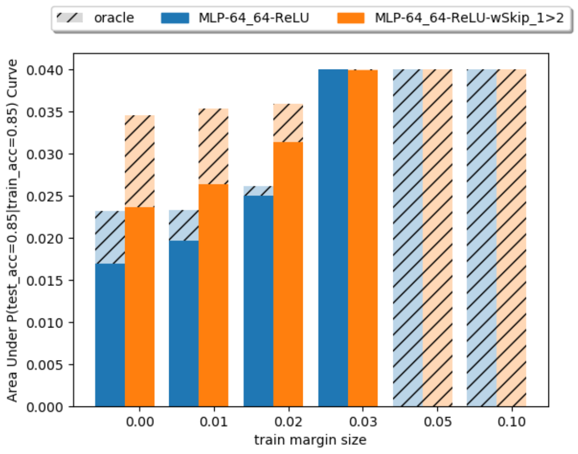
(a)



(b)



(c)



(d)

Figure A.2: The influence of various modeling choices on $P(\text{robust}|\epsilon_i, \epsilon_j)$ as measured by AUC_{ϵ_j} and $\text{AUC}^*_{\epsilon_j}$ for MLPs trained on MNIST. Subfigure (a) shows the effect of changing the activation function, (b) shows the impact of adding batch normalization, (c) shows the impact of adding dropout, and (d) shows the impact of adding skip connections in a 2-layer MLP.

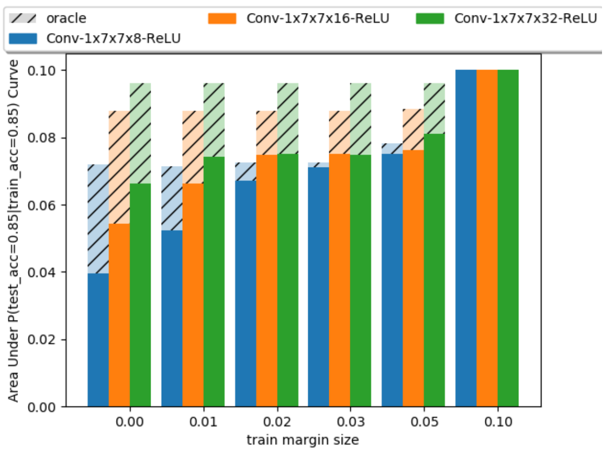
accuracy [Ioffe and Szegedy, 2015], and Galloway et al. [2019] have shown that batch normalization is detrimental to adversarial robustness because the normalization parameters are ill-suited for perturbed data.

Trends in Convolutional Neural Networks

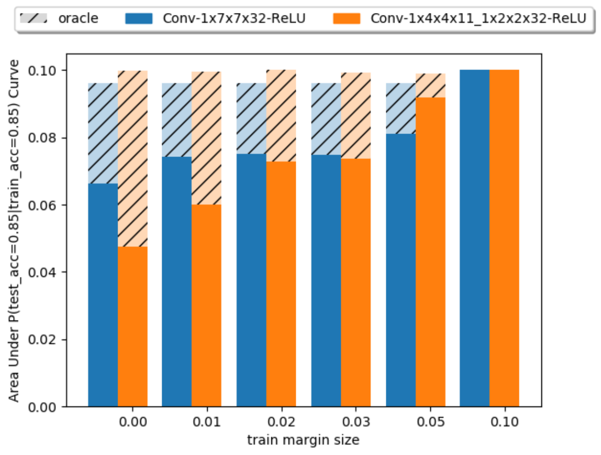
Having verified that the relationship between modeling choices and robustness potential transcends data complexity, we run experiments to determine if this relationship is maintained when the complexity of the model is increased. To this end, instead of using MLPs, the experimental results presented in this section use CNNs. At each layer of a CNN, the output is computed by diving the input (or the output of the previous layer) into segments and applying a single-layer MLP to each input segment. Therefore the modeling choices that apply to MLPs also apply to CNNs. Note that the number of convolutional filters represents the width of the model. In addition to the modeling choices common between CNNs and MLPs, there are some CNN-specific modeling choices related to how the input is to be segmented. In practice, segmentation is performed via sliding a window over the spatio-temporal dimensions of input and the modeler may choose the size of the window and the *stride* by which the window is moved in each step. Based on the sizes of the window and stride in the previous layers, we can determine for a layer its *receptive field*, i.e. the effective window size for the layer with respect to the input to the model.

Figure A.3 shows the AUC_{ϵ_j} and $\text{AUC}^*_{\epsilon_j}$ for different modelling choices. Considering the choices that are common between MLPs and CNNs first, we note that the general trend observed in sections 3.5 and 3.5.4 are present here as well with one exception that is increasing the width of the middle layer and the last layer yields higher AUC_{ϵ_j} and $\text{AUC}^*_{\epsilon_j}$ than increasing the width of the first layer (see Figure A.3c), with the middle layer yielding the highest AUC_{ϵ_j}. We hypothesize that this difference arises because unlike the MLP, the CNN processes segments of the input. The receptive field of the CNN increases at deeper layers so the additional width is more useful when the model is processing a larger part of the input.

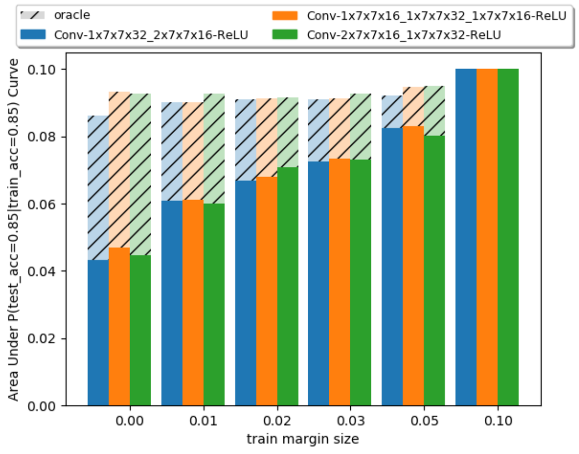
Turning our attention to modeling choices specific to CNNs, namely the size of the receptive field (Figure A.3h) and the stride (Figure A.3i), we note that models that have larger receptive fields and smaller strides tend to have greater robustness potential than those with smaller receptive field and larger strides, respectively. Changing the receptive field and the stride can change the total number of model parameters so we slightly modified the width of the network such that the total number of parameters in the models being compared remained similar. This change of width does not confound the results the widest model is not the one with the highest AUC_{ϵ_j} and $\text{AUC}^*_{\epsilon_j}$ in Figures A.3h and A.3i.


oracle
Conv-1x7x7x8-ReLU
Conv-1x7x7x16-ReLU
Conv-1x7x7x32-ReLU
Area Under P(test_acc=0.85|train_acc=0.85) Curve
train margin size

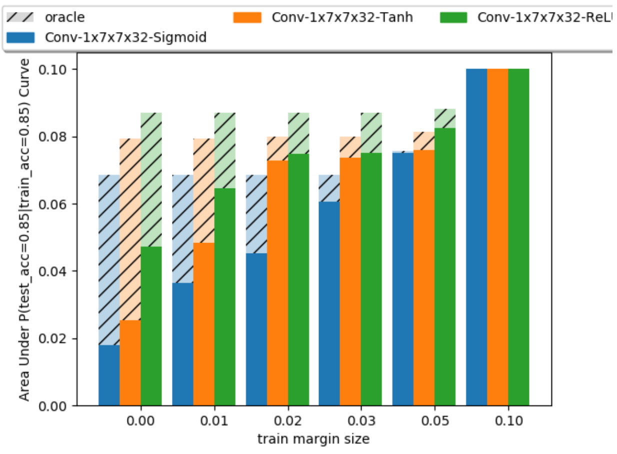
(a)


oracle
Conv-1x7x7x32-ReLU
Conv-1x4x4x11_1x2x2x32-ReLU
Area Under P(test_acc=0.85|train_acc=0.85) Curve
train margin size

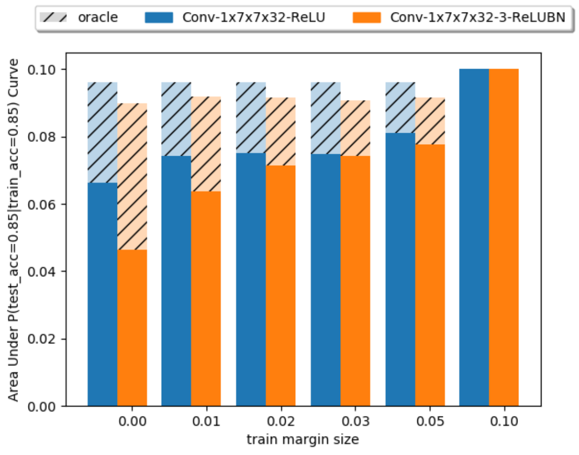
(b)


oracle
Conv-1x7x7x32_2x7x7x16-ReLU
Conv-1x7x7x16_1x7x7x32_1x7x7x16-ReLU
Conv-2x7x7x16_1x7x7x32-ReLU
Area Under P(test_acc=0.85|train_acc=0.85) Curve
train margin size

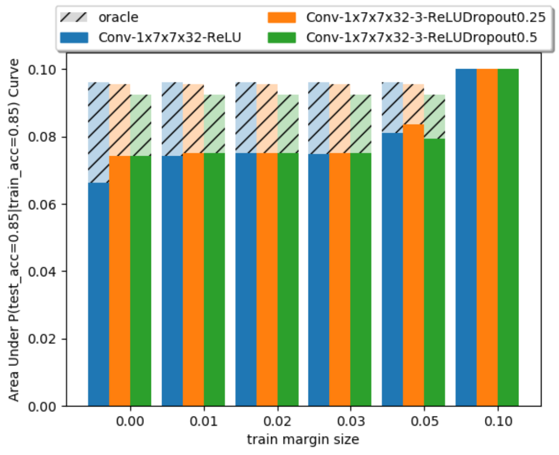
(c)


oracle
Conv-1x7x7x32-Sigmoid
Conv-1x7x7x32-Tanh
Conv-1x7x7x32-ReL
Area Under P(test_acc=0.85|train_acc=0.85) Curve
train margin size

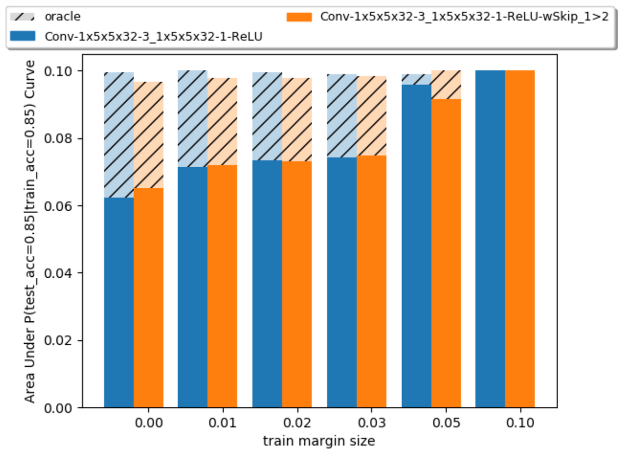
(d)


oracle
Conv-1x7x7x32-ReLU
Conv-1x7x7x32-3-ReLUBN
Area Under P(test_acc=0.85|train_acc=0.85) Curve
train margin size

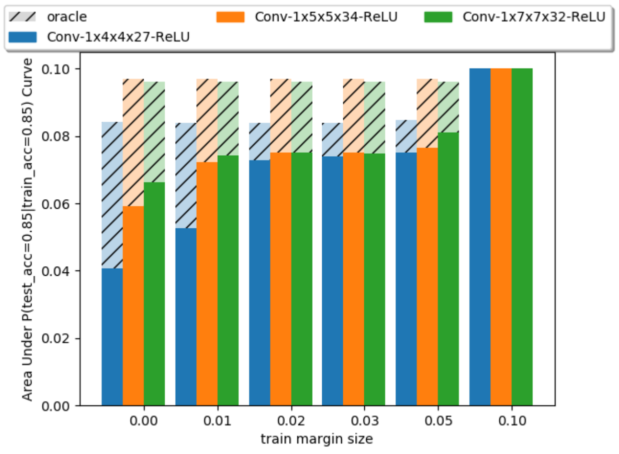
(e)


oracle
Conv-1x7x7x32-ReLU
Conv-1x7x7x32-3-ReLUDropout0.25
Conv-1x7x7x32-3-ReLUDropout0.5
Area Under P(test_acc=0.85|train_acc=0.85) Curve
train margin size

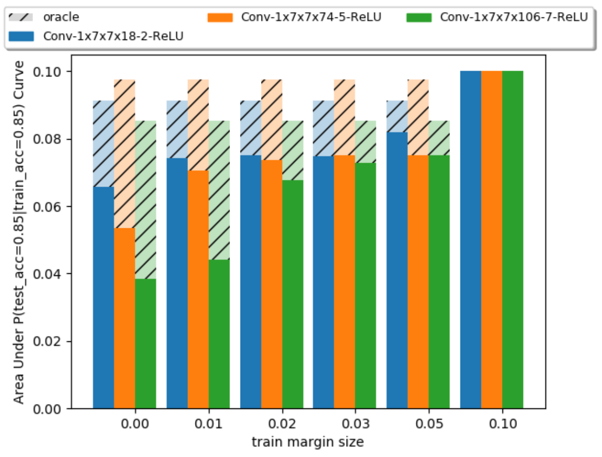
(f)



(g)



(h)



(i)

Figure A.3: The influence of various modeling choices on $P(\text{robust}|\epsilon_i, \epsilon_j)$ as measured by AUC$_{\epsilon_j}$ and AUC$^*_{\epsilon_j}$ for CNNs trained on MNIST. Subfigure (a) shows the areas for single-layer CNN with an increasing number of filters, (b) shows the areas for CNNs with increasing depth, (c) shows the effect of widening different layers in a 3-layer CNN, (d) shows the effect of changing the activation function, (e) shows the impact of adding batch normalization, (f) shows the impact of adding dropout, (g) shows the impact of adding skip connections in a 2 layer CNN, (H) shows the impact of increasing the size of the convolutional kernel, and (I) shows the impact of increasing the stride of the convolutional kernel.

Appendix B

Supplementary material for chapter 4

B.1 A Theoretical Setting for sparsity

We formalize and prove the results mentioned in section 4.2.3 linking adversarial sparsity to the number of perturbations. f,x, ϵ and $n \geq 3$ are fixed.

Proposition B.1. *Let μ be the probability measure associated with the uniform distribution over admissible set Δ. Let $\mathcal{D}$ a distribution of sequences of subsets $\Delta^m \subset \Delta$, indexed on M. Assume $\mathcal{D}$ is such that the volume of Δ^m only depends on m:*

$$\frac{\mu(\Delta^m)}{\Delta} = g(m) \tag{B.1}$$

Assume $Adv(f, x, \epsilon) = \{\delta_j, 1 \leq j \leq k\}$ with (δ_j) uniformly sampled iid. over Δ Then we have:

$$\mathbb{E}_{(\delta_j)\sim\mathcal{U}(\Delta)}[AS(f, \epsilon, x)] = \int_M (1 - g(m))^k dm \tag{B.2}$$

Proof. Let us note $X_j = \inf\{m, \delta_j \in \Delta^m\}$, such that

$$\text{AS}(f, x, \epsilon, (\Delta^m)) = \inf_{1\leq j\leq k} X_j$$

Recall that:

$$\begin{aligned}
&\mathbb{E}_{(\delta_j)\sim\mathcal{U}(\Delta)}[\text{AS}(f, \epsilon, x)] \\
&= \mathbb{E}_{(\delta_j)\sim\mathcal{U}(\Delta)}[\mathbb{E}_{(\Delta^m)\sim\mathcal{D})}[\text{AS}(f, \epsilon, x, (\Delta^m))]] \\
&= \mathbb{E}_{(\Delta^m)\sim\mathcal{D})}[\mathbb{E}_{(\delta_j)\sim\mathcal{U}(\Delta)}[\text{AS}(f, \epsilon, x, (\Delta^m))]] \\
&= \mathbb{E}_{(\Delta^m)\sim\mathcal{D})}[\int_M \mathbb{P}_{(\delta_j)\sim\mathcal{U}(\Delta)}[\text{AS}(f, \epsilon, x, (\Delta^m)) > m]dm]
\end{aligned} \tag{B.3}$$

Note that

$$\begin{aligned}
\mathbb{P}_{(\delta_j)\sim\mathcal{U}(\Delta)}[X_j > m] &= \mathbb{P}_{(\delta_j)\sim\mathcal{U}(\Delta)}[\delta_j \notin \Delta^m] \\
&= \frac{\mu(\bar{\Delta^m})}{\mu(\Delta)} = 1 - g(m)
\end{aligned} \tag{B.4}$$

Moreover, since $X_1, ..., X_k$ are independent,

$$\begin{aligned}
&\mathbb{P}_{(\delta_j)\sim\mathcal{U}(\Delta)}[(\inf_{1\leq j\leq k} X_j) > m] \\
&= \mathbb{P}_{(\delta_j)\sim\mathcal{U}(\Delta)}[X_1 > m \wedge ... \wedge X_k > m] \\
&= \Pi_{1\leq j\leq k}\mathbb{P}_{(\delta_j)\sim\mathcal{U}(\Delta)}[\delta_j \notin \Delta^m] \\
&= (1 - g(m))^k
\end{aligned} \tag{B.5}$$

It follows

$$\begin{aligned}
&\mathbb{E}_{(\delta_j)\sim\mathcal{U}(\Delta)}[\text{AS}(f, \epsilon, x)] \\
&= \mathbb{E}_{(\Delta^m)\sim\mathcal{D})}[\int_M (1 - g(m))^k dm] \\
&= \int_M (1 - g(m))^k dm
\end{aligned} \tag{B.6}$$

□

In the L_2 case $M = [0, \pi]$, $\Delta = S_2^n$ and Δ^α is a spherical cap of angle α. When $\alpha \leq \frac{\pi}{2}$ formula expressing the area of a spherical cap is derived in Li [2011] and equal to:

$$A(\alpha) = \frac{\pi^{\frac{n}{2}}}{\Gamma(\frac{n}{2})} I_{\sin\alpha^2}(\frac{n-1}{2}, \frac{1}{2}) \tag{B.7}$$

where Γ is the Gamma function and I is the regularized incomplete beta function. Given that $I_1(a, b) = 1$ it follows that:

$$g(m) = g(\alpha) = t_\alpha := I_{\sin\alpha^2}(\frac{n-1}{2}, \frac{1}{2}) \tag{B.8}$$

.

When $\frac{\pi}{2} < \alpha < \pi$ the cap of angle α is merely the complementary set on the sphere of the cap of angle $\pi - \alpha$. Therefore $g(\alpha) = 1 - g(\pi - \alpha)$ and:

$$\begin{aligned}
\mathbb{E}[\text{AS}(f, x, \epsilon)] &= \int_0^\pi ((1 - g(\alpha))^k d\alpha \\
&= \int_0^{\frac{\pi}{2}} ((1 - t_\alpha)^k + t_\alpha^k) d\alpha
\end{aligned} \tag{B.9}$$

In the L_∞ case, $M = \{0, ..., n\}$, $\Delta = \{\pm 1\}^n$, $\Delta^m = \{\delta \in \Delta \mid \forall q > m \quad \delta_{\sigma(q)} = u_{\sigma(q)}\}$. Therefore $g(m) = 2^{m-n}$ and

$$\mathbb{E}[\text{AS}(f, \epsilon, x)] = \sum_{m=0}^{n} (1 - 2^{m-n})^k = \sum_{m=0}^{n} (1 - 2^{-m})^k \tag{B.10}$$

We now show that $\mathbb{E}[\text{AS}(f, \epsilon, x)] \in \Theta(n - \log_2 k)$. On the one hand,

$$\ln((1 - 2^{-m})^k = k. \ln(1 - 2^{-m}) \leq -k.2^{-m} \tag{B.11}$$

from which it follows:

$$\begin{aligned}
\mathbb{E}[\mathrm{AS}(f,\epsilon,x)] &\leq \sum_{m=0}^{\lfloor \log_2 k \rfloor} e^{-2^{\log_2 k - m}} + \sum_{m=\lfloor \log_2 k \rfloor+1}^{n} (1-2^{-m})^k \\
&\leq e^{-1} + ... + e^{-m} + (n - \lfloor \log_2 k \rfloor).1 \\
&\leq n - \log_2 k + 1 + \frac{1}{e-1} \\
&= n - \log_2 k + \frac{e}{e-1}
\end{aligned} \tag{B.12}$$

On the other hand,

$$k.\ln(1-\frac{1}{k}) \geq k.(1-\frac{1}{1-\frac{1}{k}}) = \frac{-k}{k-1} \tag{B.13}$$

Thus for $k \geq 2$

$$\begin{aligned}
\mathbb{E}[\mathrm{AS}(f,\epsilon,x)] &\geq \sum_{m=\lfloor \log_2 k \rfloor+1}^{n} (1-2^{-m})^k \\
&\geq (n - \lfloor \log_2 k \rfloor).(1-2^{-\lfloor \log_2 k \rfloor - 1})^k \\
&\geq (n - \log_2 k).(1-\frac{1}{k})^k \\
&\geq (n - \log_2 k).e^{-\frac{k}{k-1}} \\
&\geq (n - \log_2 k).e^{-2}
\end{aligned} \tag{B.14}$$

Since $(1-\frac{1}{2})^2 = \frac{1}{4}$,$(1-\frac{1}{3})^3 = \frac{8}{27}$ and for $k \geq 4$ $e^{=\frac{k}{k-1}} \geq e^{\frac{4}{3}} > \frac{1}{4}$ we can conclude:

$$\mathbb{E}[\mathrm{AS}(f,\epsilon,x)] \geq \frac{(n-\log_2 k)}{4} \tag{B.15}$$

This also applies when $k = 1$, since $\mathbb{E}[\mathrm{AS}(f,\epsilon,x)] = n - \sum_{m \leq n} 2^{-m} \geq n - 2 \geq \frac{n}{4}$

B.2 Additional experiments

B.2.1 ImageNet

L_2 attacks on ImageNet are not as popular as on CIFAR10. We however apply them, both for reference and to verify that angular sparsity computation scales reasonably to a larger input size. We evaluate pretrained ResNet models provided in the Robustbench framework. One is trained in a standard fashion and the other adversarially against L_∞ perturbations, following Salman et al. [2020a]. All take inputs of size 224x224. We apply L_2 perturbations of radius $\epsilon = 1.0$. Everything else is similar to the CIFAR10 experiments.

We report the results in Figure B.1. We report higher robustness (under all metrics) than we did for CIFAR10; this is consistent with the fact that $\epsilon = 1.0$ in a 224x224 vector space allows a smaller perturbation per pixel budget than $\epsilon = 0.5$ in a 32x32 space. In the absence of L_2-robust, easily available ImageNet models we chose a small value of ϵ.

1	Architecture	Defenses	Natural accuracy	Adv. accuracy	Sparsity
2	ResNet-50	None	87.8%	0%	0.194
3	ResNet-18	Salman et al. [2020a]	52.9%	39.5%	0.426
4	ResNet-50	Salman et al. [2020a]	64.0%	53.4%	0.415

Table B.1: Evaluation of defended and undefended ImageNet models under L_2 attack with 20 iterations and $\epsilon = 1.0$. We report Natural accuracy (without attack), adversarial accuracy (AutoPGD), and adversarial angular (residual) sparsity.

B.2.2 Visual Transformers

We evaluate a B-16 Visual Transformer Architecture (ViT), pretrained on ImageNet and fine-tuned on CIFAR10 [Dosovitskiy et al., 2021]. Recent works have shown that these models learn different features than Convolutional Neural Networks [Raghu et al., 2021] which raises the question of their behavior against adversarial examples. In fact, some works have already claimed that ViTs are more robust to attacks than CNNs [Shao et al., 2021]. Investigating these claims is an interesting use case of adversarial sparsity.

We evaluate this ViT model on L_2 perturbations with the same parameters as in section 4.5. It reaches a natural accuracy of 97%, greater than any of our ResNet-18 models, and an adversarial accuracy of 0%. Its L_2 sparsity is 0.183, almost equal to that of the ResNet-18 model trained with data augmentation (line 3). The training recipe we use to fine-tune the model employs similar augmentation methods. This suggests that ViTs and ResNets behave similarly against the same L_2 PGD threat model. This challenges the conclusions of Shao et al. [2021] on the adversarial robustness of ViTs.

Appendix C

Supplementary material for chapter 5

C.1 `robust_speech`: an ASR robustness framework

C.1.1 Motivation

There exist multiple robustness-oriented development frameworks: `Advertorch` [Ding et al., 2019], `Foolbox` [Rauber et al., 2020, 2017], `Cleverhans` [Papernot et al., 2018], etc. However, nearly all of these packages suppose that the model performs image classification. Yet applying standard classification-based attacks to Speech Recognition is not trivial, as ASR models are typically more complex than a stack of PyTorch layers. For instance, they must handle variable length inputs, are trained with tricky losses, contain recurrent networks, etc. Some attacks have been developed with the explicit goal of fooling ASR models (for instance by relying on acoustic models), and these attacks are rarely included in general robustness packages.

`robust_speech`[1] fills that gap and offers a simple way to evaluate ASR models against Adversarial attacks. It is based on Speechbrain [Ravanelli et al., 2021], a flexible Speech Toolkit that handles loading ASR datasets, running ASR models, and accessing their loss, predictions, and error rates - all of which are often necessary to run attacks. `robust_speech` adds useful features and metrics to SpeechBrain and reproduces multiple general and ASR-specific attacks.

Chapters 5,6, 7 and 8 all use `robust_speech` as a primary codebase for running and evaluating attacks and defenses.

C.1.2 Supported features

A non-exhaustive list of features supported by `robust_speech` includes:

- Attacks: PGD [Madry et al., 2018], CW [Carlini and Wagner, 2016, 2018], Imperceptible [Qin et al., 2019] Genetic [Taori et al., 2019], Universal [Neekhara et al., 2019], Kenansville [Abdullah et al., 2021a], Yeehaw [Abdullah et al., 2023]...
- Defenses: Randomized smoothing [Cohen et al., 2019, Olivier and Raj, 2021], Adversarial training [Madry et al., 2018, Olivier and Raj, 2023b]

[1] `https://github.com/RaphaelOlivier/robust_speech`

- Models: DeepSpeech2 [Amodei et al., 2016], Whisper [Radford et al., 2022], all SpeechBrain models, and customizable Transformers and LSTM models
- Support for self-supervised models on Fairseq [Ott et al., 2019] and HuggingFace [et al., 2020], such as Wav2Vec2 [Baevski et al., 2020], HuBERT [Hsu et al., 2021] and Data2Vec [Baevski et al., 2022b]
- Datasets: CommonVoice [Ardila et al., 2020], LibriSpeech [Panayotov et al., 2015], SpeechCommand [Warden, 2018], with evaluation metrics WER, CER, SER, SNR.
- Run attacks over several models at once, or from a source to a target model

C.2 Additional details on Membership Inference experiments

This section reports all of our results on Membership inference (MI) experiments. It is a more thorough version of section 5.5.3, which evaluates both the impact of white-box features and input perturbations on MI performance.

C.2.1 Experimental Setup

Baseline MI features

Our baseline feature extractor follows Shah et al. [2021c] and corresponds to a set of errors computed between the target and output transcriptions of the ASR model M, combined with the model's confidence with regard to these transcriptions. Specifically, we use: the Word Error Rate (WER); the total number of edits, substitutions, insertions, and deletions; the length ratio between prediction and target transcription; the confidence of the model with regard to the transcription, i.e. the log-likelihood of the transcription. We compute all these features for the top four transcription hypotheses of the model and dub their combination as the Errors feature set. Those features can be computed in a *black-box* setting where the attacker has only access to model outputs and confidence.

Data

Our datasets to train ASR target and shadow models and MI classifiers all use data taken from LibriSpeech (LS) [Panayotov et al., 2015]. Specifically, for the target ASR model, $\mathcal{D}^{\text{trg}}_{\text{ASR}_{\text{train}}}$ is composed of 300h from LS's train-clean-360 partition. The remaining 60h of this partition were held out for preliminary experiments and to create the *negative* samples of the $\mathcal{D}^{\text{trg}}_{\text{MI}}$ partitions. Similarly, for the shadow model, $\mathcal{D}^{\text{shd}}_{\text{ASR}_{\text{train}}}$ contains 80h from LS's train-clean-100 partition, whereas the remaining 20h are held-out data for preliminary experiments and to create the *negative* MI subsets. Both $\mathcal{D}_{\text{MI}_{\text{train}}}$ are composed of 5,000 utterances ($\sim$ 18h), whereas each $\mathcal{D}_{\text{MI}_{\text{test}}}$ contains 1,000 utterances (~ 3.5h).

Table C.1: Results for experiments with full and partial knowledge of the training data. For every experiment, we report the mean and standard deviation for different metrics, obtained by fitting the MI classifier 10 times with different random seeds.

Knowledge	**#**	**Type of Input**	**Features**	**Acc**	**AUC**	**AUC$_{\text{FPR}=0.1}$**	**Access**
	1	Natural	Errors	70.5 ± 0.53	77.5 ± 0.27	58.8 ± 0.58	Black
	2	Natural	Att	84.6 ± 0.14	89.3 ± 0.10	62.2 ± 0.34	
	3	Natural	Att + CTC	87.2 ± 0.39	92.7 ± 0.09	72.4 ± 0.29	Gray
	4	Natural	Att + CTC + Errors	88.2 ± 0.18	93.8 ± 0.14	75.3 ± 0.72	
	5	Gaussian	Att	87.4 ± 0.32	93.8 ± 0.13	76.5 ± 0.83	
	6	Gaussian	Att + CTC	87.5 ± 0.23	93.8 ± 0.15	76.5 ± 0.58	Gray
	7	Gaussian	Att + CTC + Errors	88.6 ± 0.37	94.5 ± 0.14	77.6 ± 0.83	
Full	8	Adversarial	Att	86.2 ± 0.21	92.8 ± 0.20	73.7 ± 0.76	
	9	Adversarial	Att + CTC	86.8 ± 0.26	93.1 ± 0.13	75.2 ± 0.29	White
	10	Adversarial	Att + CTC + Errors	86.0 ± 0.58	92.6 ± 0.25	74.1 ± 0.94	
	11	Nat. + Gauss.	Att + CTC	89.4 ± 0.33	94.8 ± 0.14	78.1 ± 0.57	Gray
	12	Nat. + Adv.	Att + CTC	89.1 ± 0.23	$\mathbf{95.1 \pm 0.09}$	$\mathbf{79.5 \pm 0.45}$	White
	13	Nat. + Gauss. + Adv	Att	$\mathbf{89.5 \pm 0.25}$	95.1 ± 0.12	79.0 ± 0.68	
	14	Nat. + Gauss. + Adv.	Att + CTC	89.4 ± 0.36	95.0 ± 0.16	78.9 ± 0.73	White
	15	Nat. + Gauss. + Adv.	Att + CTC + Errors	88.3 ± 0.23	94.0 ± 0.12	75.9 ± 0.49	
	16	Natural	Errors	71.4 ± 0.45	79.0 ± 0.23	57.3 ± 0.34	Black
	17	Natural	Att + CTC + Errors	87.4 ± 0.23	92.3 ± 0.17	70.6 ± 0.92	Gray
Partial	18	Gaussian	Att + CTC	85.9 ± 0.36	93.1 ± 0.19	74.0 ± 0.86	Gray
	19	Adversarial	Att + CTC	85.9 ± 0.41	91.7 ± 0.14	70.4 ± 0.37	White
	20	Nat. + Gauss. + Adv.	Att	87.7 ± 0.21	$\mathbf{94.3 \pm 0.10}$	$\mathbf{76.9 \pm 0.65}$	White
	21	Nat. + Gauss. + Adv.	Att + CTC	$\mathbf{88.1 \pm 0.17}$	94.2 ± 0.19	75.9 ± 0.87	

ASR target and shadow models

Our target and shadow models correspond to SpeechBrain's Transformer model. As mentioned above, the *target* model was trained using the $\mathcal{D}^{\text{trg}}_{\text{ASR}_{\text{train}}}$ dataset, whereas the *shadow* model was trained using the $\mathcal{D}^{\text{shd}}_{\text{ASR}_{\text{train}}}$ dataset. We evaluated the two models using the *test-clean* and *test-other* partitions of LS. Both models follow SpeechBrain's default configuration and training parameters with the exception of: the number of epochs = 60; batch size = 16; gradient accumulation factor = 2. All experiments were run without the use of a language model. The target model obtains a WER of 5.45% and 15.17% for LS's test-clean and test-other partitions, whereas the shadow model obtains a WER of 10.32% and 24.70%, for the same partitions. When computing features, decoding was performed with a beam size of 30.

Attack perturbations

We experimented with several choices of hyperparameters for the perturbations detailed in Algorithms 3 and 2. In the results we report, for Gaussian perturbations, we use 8 different SNRs linearly spaced between 0 and 50dB. For each SNR we perturb the signal 4 times, after which we average the resulting features. For adversarial perturbations, we use 16 adversarial radii ϵ: nine evenly spaced from 0.001 to 0.009 and seven from 0.01 to 0.07. We fix $\eta = 1$ and $N = 1$, which we find are as effective for MI as more computationally expensive hyperparameters.

Evaluation and implementation details

To allow our work to be compared to other approaches in the literature, our main metrics of evaluation are accuracy (Acc) and area under the ROC curve (AUC). However, the authors in Carlini et al. [2022] argue that MI attacks should be evaluated for low false-positive rates, as identifying a few data records with high precision should be evaluated higher than identifying many unreliably. In line with this argument, we additionally report the performance of our attack in terms of the normalized AUC for a maximum false positive rate (FPR) of 0.1.

Experiments using adversarial noise were built with `robust_speech`. MI is performed using Scikit-Learn's Random Forest classifier [Pedregosa et al., 2011] with 100 estimators.

C.2.2 Results

The results for our full ablation study are aggregated in Table C.1. We are interested in evaluating the effectiveness of loss-based features on the one hand, and perturbation-based features on the other.

Lines 1–15 correspond to a full knowledge attack – where the shadow model is equal to the target model – while lines 16-21 correspond to a partial knowledge attack, where we leverage a shadow model to train the MI classifier. In the table, a *Natural* input corresponds to an experiment with features computed from the original data, while *Gaussian* and *Adversarial* correspond to features computed from perturbed data.

Lines 1 and 16 report the results obtained with the Errors feature set for the original input. These experiments are based on black-box access to the model, previously explored in the literature [Shah et al., 2021c]. From lines 2–4, we see that allowing gray-box access, and consequently loss computation, substantially improves all success metrics in comparison to the black-box access of line 1. In fact, by simply using the attention loss (Att) as an isolated feature, we are able to improve results by over 10% for both Acc and AUC, while the CTC loss and Errors only provide incremental improvements over the attention loss.

From lines 5–7 and 8–10, we observe a similar trend for the contribution of each feature for perturbed inputs. In addition, we see that considering only the information on either the Gaussian or the adversarial perturbations provides results that are comparable to the natural inputs. Interestingly, while for the Gaussian perturbations, the Errors set seems to improve results, this does not hold for adversarial perturbations. We hypothesize that since the adversarial perturbation moves the input to a "worst-case" location in terms of loss, the Errors may not provide any meaningful information.

When combining natural inputs with each of the perturbed inputs, lines 11–12, we see that both types of perturbations bring a similar level of improvement, reaching values of close to 89% and 95% for Acc and AUC. Moreover, even though we reach the best result for Acc – and very close performance in terms of the remaining metrics, when compared to line 12 –when combining natural and both types of perturbed inputs, this combination of features does not improve MI performance in a meaningful way. Still, it is worth noting that lines 11–13 bring an improvement of nearly 20% when compared to line 1, using only loss-based features. On the other hand, as was observed in line 10, when adding error features to the full combination of inputs (line 15), performance degrades for all metrics. This may be due to classification overfitting, as the total number of features grows to 225.

For the more realistic scenario of lines 16–21, we select only the feature combinations that provide us with the best results in lines 1–15. Here results follow a similar trend, with the combination of natural and both perturbed inputs giving us the best overall results, allowing for an accuracy close to 88%, a 1% degradation over the full-knowledge attack. However, for both settings, considering only natural or Gaussian perturbed inputs already results in very high accuracy. This may be helpful when computational resources are limited and calculating adversarial perturbations is not an option. Further, adversarial perturbations should only be considered as an additional feature, as they alone do not perform better than just the natural input.

Finally, for a maximum FPR of 10%, the proposed features are generally able to reach values close to 80% AUC, almost 20% above Errors features. This shows that low-FPR operation points, which are particularly relevant to MI applications [Carlini et al., 2022], also benefit from loss and perturbation-based features.

Appendix D

Supplementary material for chapter 6

D.1 Black box attack results on speech recognition

Beyond transferability, black-box optimization and optimization-free attacks are other approaches to fool models without gradient access. We evaluate the validity of those approaches in this section, by analyzing the models of Section 5.2 on existing attacks. We focus on the untargeted setting, which is much simpler than the targeted one.

D.1.1 Attacks

Genetic ASR attack

Alzantot et al. [2018] propose an indirect optimization attack for ASR models. It employs a genetic algorithm where a population is maintained and renewed, using the adversarial objective as a fitness score. Such indirect attacks are useful to attack models that are only available as a black-box oracle, without gradient information; but also to fool models whose gradients are not useful due to the *obfuscation* effect of an incomplete defense [Athalye et al., 2018a].

The authors of Alzantot et al. [2018] use this method as a targeted attack. However, they only evaluated it on the Speech Commands dataset with 10 possible labels. To run such an attack on a large ASR model with a full English sentence as the target is considerably more challenging. To keep some amount of attack success we use the same algorithm for a simpler, untargeted objective: the loss on the original label is used as a fitness score. Perturbations are L_∞-bounded by a hyperparameter ϵ.

Kenansville attack

The DFT Kenansville attack [Abdullah et al., 2021a] is a Signal Processing attack: it does not consider model predictions or loss to modify the audio signal. Instead, this attack removes the spectral components of the input that have the lowest power spectral density (PSD). This leaves human perception mostly unchanged but typically affects the predictions of ASR models, which tend to exploit high-frequency components more than humans. Like L_2-PGD the distortion of that attack is controlled with the SNR.

D.1.2 Experimental setting

We reproduce the setting of Section 5.2. We attack SpeechBrain LSTM and Transformer, Seq2Seq and CTC models, as well as DeepSpeech2 and Wav2Vec2 models. We run attacks on the LibriSpeech dataset. We use 2000 optimization steps for the Genetic attack. We evaluate attacks with the WER on the original input.

D.1.3 Results

Model	Clean WER	Genetic WER↑
SpeechBrain LSTM Seq2Seq	5.02	25.49
SpeechBrain LSTM CTC	6.15	24.27
SpeechBrain Transformer Seq2Seq	4.11	21.6
SpeechBrain Transformer CTC	5.88	20.63
Wav2Vec2-BASE-100h	6.27	13.19
Wav2Vec2-BASE-960h	3.53	16.63
Wav2Vec2-LARGE-960h	2.85	11.08
DeepSpeech2	9.44	74.51

Table D.1: results of all models on clean data and under the (targeted) Carlini&Wagner attack and the (untargeted) Genetic attack. For CW we report the WER, SNR, and attack accuracy (i.e. success rate). The bound for the Genetic attack corresponds to an average SNR of $20dB$. The up and down arrows indicate whether large or small WER values mean successful attacks.

The Kenansville DFT attack (Figure D.1) has little effect in our evaluation setting: SNRs greater than 20dB fail to increase the WER significantly, and adversarial examples with lower SNRs are easily detected by the human ear. Comparing these results to Abdullah et al. [2021a] is not easy for two reasons: the authors report different metrics to measure distortion and evaluate on a different dataset. The adversarial examples they publicly released show that the original speech inputs are of lower quality than the LibriSpeech clean test set. It is therefore possible that Kenansville is most effective on inputs that are already challenging to the ASR models.

The Genetic attack [Alzantot et al., 2018] (Table D.1 column 3) also only achieves a WER degradation of each model by 8 to 20 points, even with an SNR of 20dB, for most models. One major exception is the DeepSpeech2 model, which was proposed earliest and on which this attack was initially evaluated by its original authors. Therefore we confirm the effectiveness of this attack when reproducing its original evaluation condition, but it loses most of its effectiveness when applied to stronger models.

DeepSpeech2 is also the most affected model by the Kenansville attack, by a 5% WER margin. Beyond architecture changes, one important improvement since DeepSpeech2 has been the introduction of data augmentation methods like SpecAug [Park et al., 2019]. We hypothesize that this method plays a role in defending models against some attacks. The Kenansville attack in particular relies heavily on alterations of the input's frequency spectrum, and models trained with spectral augmentation are likely to be less vulnerable to those simple attack methods.

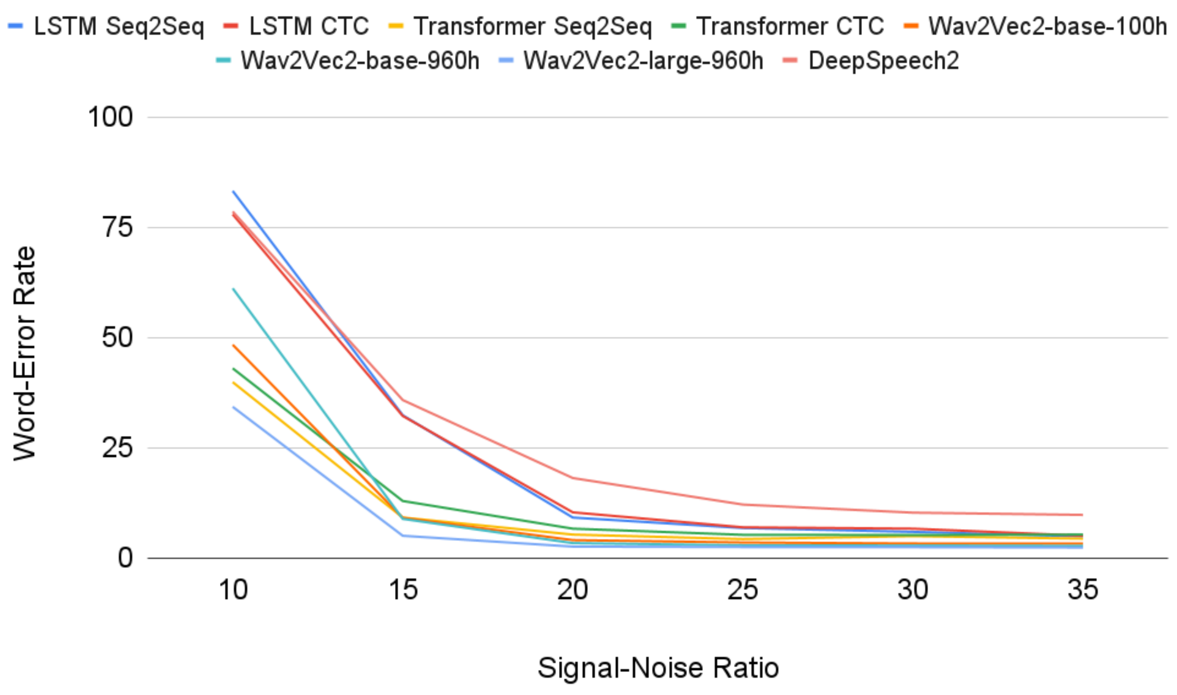


Figure D.1: Attacks results for the untargeted DFT Kenansville attack. We plot the WER as a function of attack SNR.

In conclusion, it seems that recent progress in ASR performance and model scaling was sufficient to block existing black-box attacks. Since we observe this in the simple untargeted setting, it naturally extends to targeted attacks as well. The success of transferable adversarial attacks, the other gradient-free approach to fool models, is therefore closely related to the threat that adversarial attacks can cause in practical settings.

D.2 Experimental details for LibriSpeech experiments

D.2.1 Frameworks

We compute adversarial examples using the robust_speech framework (section C.1). This library uses Speechbrain [Ravanelli et al., 2021] to load and train ASR models and offers implementations of various adversarial attack algorithms. Models and attacks are implemented using PyTorch [Paszke et al., 2019].

We use robust_speech to evaluate SpeechBrain-supported models. In section 6.4 we export a HuggingFace Dataset [Lhoest et al., 2021], then evaluate models via the HuggingFace Transformers [et al., 2020] library. Finally, we use Fairseq [Ott et al., 2019] for training models from scratch

D.2.2 Attack Hyperparameters

We exploit the Carlini&Wagner attack with the following hyperparameters:

- initial ϵ: 0.015 (and 0.04 in appendix D.5.1)
- learning rate: 0005
- number of decreasing ϵ values: 1
- Regularization constant c: 10
- optimizer: SGD
- attack iterations: 10000 in section 6.3.1, 1000 in section 6.5

D.2.3 Dataset and targets

Our adversarial dataset in section 6.3.1 consists of 85 sentences from the LibriSpeech test-clean set. To extract these sentences we take the first 200 sentences in the manifest, then keep only those shorter than 7 seconds. In section 6.5, we take the first 100 sentences and filter those shorter than 14 seconds.

As attack targets, we use actual LibriSpeech sentences sampled from the test-other set. Our candidate targets are:

- Let me see how can I begin
- Now go I can't keep my eyes open
- So you are not a grave digger then
- He had hardly the strength to stammer
- What can this mean she said to herself
- Not years for she's only five and twenty
- What does not a man undergo for the sake of a cure
- It is easy enough with the child you will carry her out
- Poor little man said the lady you miss your mother don't you
- At last the little lieutenant could bear the anxiety no longer
- Take the meat of one large crab scraping out all of the fat from the shell
- Tis a strange change and I am very sorry for it but I'll swear I know not how to help it
- The bourgeois did not care much about being buried in the Vaugirard it hinted at poverty Pere Lachaise if you please

For each sentence we attack, we assign the candidate target with the closest length to the sentence's original target.

D.2.4 Models

Training Wav2Vec2 models from scratch

We use Fairseq to train BASE and LARGE Wav2Vec2 models from scratch. Unfortunately, no configuration or pretrained weights have been released for that purpose, and we resort to using Wav2Vec2 fine-tuning configurations while simply skipping the pretraining step. Despite our attempts to tune training hyperparameters, we do not match the expected performance of a Wav2Vec2 model trained from scratch: [Baevski et al., 2020] report a WER of 3.0% for a large model, while we only get 9.1%.

Generating adversarial examples

Wav2Vec2, HuBERT, and Data2Vec models are all supported directly in robust_speech and are therefore those we use for generating adversarial examples. We use the HuggingFace backend of Speechbrain for most pretrained models, and its Fairseq backend for a few (Wav2Vec2-BASE models fine-tuned on 10h and 1h, and models trained from scratch). In both cases, the model's original tokenizer cannot be loaded in SpeechBrain directly. Therefore, we fine-tune the final projection layer of each model on 1h of LibriSpeech train-clean data.

The Wav2Vec2 model pretrained and fine-tuned on CommonVoice is a SpeechBrain original model. Similarly, we fine-tune it on 1h of LibriSpeech data as a shift from the CommonVoice output space to the LibriSpeech one. As a result, all our models share the same character output space.

Evaluating pretrained models

In section 6.4, we directly evaluate models from HuggingFace Transformers and SpeechBrain on our adversarial dataset, without modification.

D.3 Experimental details and results for small-scale experiments

This section describes the experimental details used in section 6.6.

D.3.1 CIFAR10 experiments

We use a pretrained ResNet18 as a proxy, and a pretrained ResNet50 as a private model.

Our "very targeted attack" PGD$_k$ consists in applying the following steps for each input:

- **target selection**. We sample uniformly an ordered subset of k classes out of 10 (E.g. with $k = 3$: $(2, 5, 6)$). We also sample a point uniformly on the unit k-simplex $\{x_1, ..., x_k \in [0, 1]^n / \sum_i X_i = 1\}$, by sampling from an exponential distribution and normalizing [Onn and Weissman, 2011] (e.g. $(0.17, 0.55, 0.28)$). We combine the two to obtain a 10-dimensional vector with zero probability on all but the selected k classes ($y = (0, 0.17, 0, 0, 0.55, 0.28, 0, 0, 0, 0)$). This is our target.

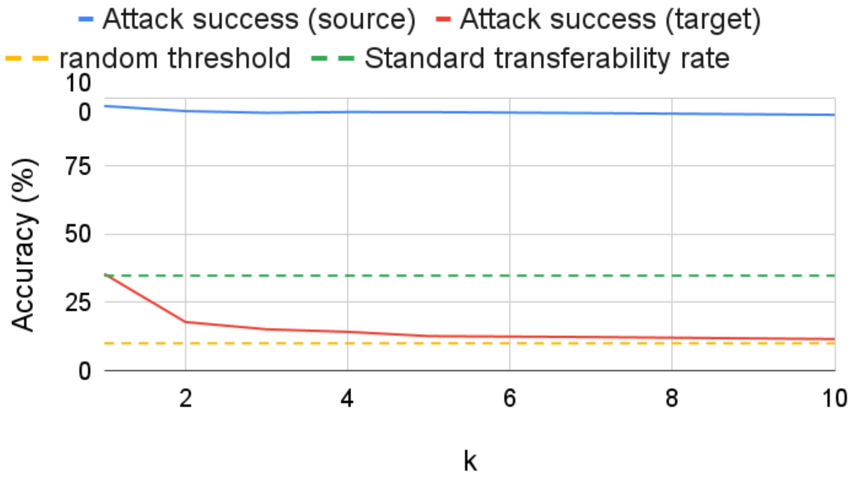


Figure D.2: Transferred attack success rate when varying the "target precision" on a CIFAR10 model. The more targeted the attack, the worse its transferability at an equal white-box success rate

- During the attack, we use Projected Gradient Descent [Madry et al., 2018] to minimize the KL divergence $\mathrm{KL}(f(x), y)$ between the softmax output and the target, within L_2 radius $\epsilon = 0.5$. We use learning rate 0.1 for $k * 1000$ attack steps.
- We measure attack success rate by measuring the top-k match between $f(x)$ and y:

$$\mathrm{Acc} = \frac{1}{k}\sum_{i=1}^{k} \mathbf{1}[\mathrm{argsort}(f(x))_i = \mathrm{argsort}(y)_i \tag{D.1}$$

  with $\mathrm{argsort}(y)$ returning the indices of the sorted elements of y in decreasing order. For instance $f(x) = (0.1, 0.05, 0.05, 0.05, 0.35, 0.2, 0.05, 0.05, 0.05, 0.05)$ would get an accuracy of $0.66\underline{6}$, as the top 2 classes match with y but not the third.

We evaluate attacks on 256 random images from the CIFAR10 dataset. For each value of k between 1 and 10 we repeat the experiment 3 times and average the attack success rates. In figure D.2 we plot the L_∞ attack success rate ($\epsilon = 0.03$) for both white-box and transferred attacks as a function of the "target precision" k.

D.3.2 Mildly targeted ASR attacks

We train 5 identical conformer encoder models with 8 encoder layers, 4 attention heads, and hidden dimension 144. We train them with CTC loss for 30 epochs on the LibriSpeech train-clean-100 set, with different random seeds.

We run a L_2-PGD attack with SNR bound 30dB, in which we minimize the cross-entropy loss between the utterance and its transcription prepended with the word "But". The utterances we attack are the first 100 sentences in the LibriSpeech test-clean set, to which we remove 7 sentences already starting with the word "But". We generate adversarial examples using each of the 5 models as a proxy and evaluate these examples on all 5 models. We report the full results in Table D.2.

Model\Proxy	1	2	3	4	5
1	98%	14%	12%	13%	19%
2	23%	100%	11%	14%	24%
3	18%	22%	96%	20%	16%
4	12%	19%	20%	100%	17%
5	28%	14%	22%	22%	96%

Table D.2: Success rate of our mildly targeted attack, using each of the 5 conformer networks both as proxy and model. The attack is considered successful on an input if the prepended target word is the first word in the transcription.

D.4 Full results table for cross-model LibriSpeech and LibriVox attacks

Table D.3 completes the ablation study in section 6.5 by evaluating all pairwise Proxy-Model combinations in our pool of Wav2Vec2-type models, trained on LibriSpeech and/or LibriLight.

D.5 Influence of hyperparameters on attack results

D.5.1 Attack radius

In Table D.4 we extend the results of Table 6.2 by comparing attack results for two different attack radii. These radii are $\epsilon = 0.015$ and $\epsilon = 0.04$, corresponding respectively to Signal-Noise Ratios of 30dB and 22dB respectively. The former is identical to Table D.4; the latter is substantially larger, and corresponds to a more easily perceptible noise.

Looking at the white-box attack results on the proxy models the difference is drastic: with larger noise the targeted success rate jumps from 88% to 98%. The transferred attack results on SSL-pretrained models also increase overall, with success increases ranging from 0% (Wav2Vec2-LARGE) to 20% (Data2Vec-LARGE) with a median increase of 10%. Crucially, however, the targeted success does not increase at all and even decreases for ASR models trained from scratch. This confirms that there is a structural difference between the robustness of ASR models with and without SSL, that cannot be bridged simply by increasing the attack strength.

D.5.2 Language models

In section 6.4 we report the results of our adversarial dataset on multiple Wav2Vec2-type models, enhanced with an N-gram language model whenever available. In Table D.5 we evaluate the influence of that language model on attack results.

We observe that the attack success rate systematically increases by 8 to 17% when adding a language model to the ASR model. This is understandable considering that our targets are sound English sentences: if a model tends to transcribe that target with mistakes, the language model can bridge that gap. To put it differently, the more prone an ASR model is to output sentences in a given distribution, the more vulnerable it is to attacks with targets sampled from that distribution.

Model\Proxy	**W2V- B**		**W2V-B**		**W2V-B**		**W2V-B**		**D2V-B**	
	LS960 960h		**LS960 100h**		**LS960 10h**		**LS960 1h**		**LS960 960h**	
	CER	*WER*	*CER*	*WER*	*CER*	*WER*	*CER*	*WER*	*CER*	*WER*
W2V-B LS960 960h	96.37	84.72	80.61	49.01	64.11	30.69	53.41	18.46	20.61	0
W2V-B LS960 100h	81.42	54.46	99.24	97.74	81.18	47.6	64.18	25.04	25.23	0
W2V-B LS960 10h	42.64	42.64	87.9	60.25	99.117	97.6	72.91	30.13	23.47	0
W2V-B LS960 1h	69.3	33.52	78.84	43.71	81.12	45.12	99.498	98.66	20.91	0
D2V-B LS960 960h	37.9	0	17.68	0	10.88	0	7.94	0	98.44	94.13
W2V-L LS960 960h	44.61	11	20.36	0	13.46	0	8.32	0	16.8	0
D2V-L LS960 960h	28.72	0	8.68	0	5.36	0	4.94	0	25.03	0
W2V-L LV60k 960h	29.24	0	11.12	0	5.68	0	3.19	0	13.73	0
HB-L LV60k 960h	23.83	0	7.29	0	4.83	0	3.91	0	14.92	0
HB-XL LV60k 960h	26.55	0	6.71	0	5.21	0	4.37	0	17.53	0
W2V-L CV CV+1h	27.38	0	12.59	0	11.01	0	9.61	0	19.24	0
W2V-B None 960h	7.84	0	4.45	0	4.05	0	3.83	0	5.51	0
W2V-L None 960h	8.12	0	4.63	0	4.55	0	3.44	0	5.44	0
	W2V- L		**D2V-L**		**W2V-L**		**HB-L**		HB-XL	
	LS960 960h		**LS960 960h**		**LV60k 960h**		**LV60k 960h**		**LV60k 960h**	
W2V-B LS960 960h	47.08	8.49	24.9	0	44.7	9.48	55.55	19.17	47.46	8.98
W2V-B LS960 100h	46.01	5.73	26.77	0	48.57	9.76	58.41	18.03	48.42	8.13
W2V-B LS960 10h	43.14	0	25.1	0	42.67	0	53.12	5.59	44.36	0
W2V-B LS960 1h	41.21	8.63	25.48	0.57	36.68	4.74	45.32	6.65	42.95	10.18
D2V-B LS960 960h	34.49	0	24.2	0	47.15	0.92	58.75	14.29	46.71	0.14
W2V-L LS960 960h	67.07	30.69	20.89	0	37.34	1.84	56.87	19.02	42.21	5.87
D2V-L LS960 960h	28.27	0	94.53	80.69	47.75	15.21	68.97	38.61	51.02	18.6
W2V-L LV60k 960h	25.19	0	16.05	0	97.13	88.61	71.78	39.18	46.61	11.88
HB-L LV60k 960h	27.19	0	30.08	0	49.27	17.47	97	87.98	56.83	28.71
HB-XL LV60k 960h	33.31	0	30.5	0	51.68	14.99	83.92	55.3	87.66	62.38
W2V-L CV CV+1h	27.8	0	26.85	0	56.72	11.67	46.94	0	39.95	0
W2V-B None 960h	11.19	0	9.6	0	7.16	0	6.72	0	11.07	0
W2V-L None 960h	11.15	0	9.19	0	7.52	0	7.45	0	11.23	0
	W2V- L				**W2V-B**				**W2V-L**	
	CV CV+1h				**None 960h**				**None 960h**	
W2V-B LS960 960h	10.81	0			2.62	0			2.53	0
W2V-B LS960 100h	11.01	0			2.82	0			2.58	0
W2V-B LS960 10h	11.19	0			2.65	0			2.66	0
W2V-B LS960 1h	11.81	0			3.03	0			3.04	0
D2V-B LS960 960h	8.01	0			2.32	0			2.38	0
W2V-L LS960 960h	8.2	0			2.39	0			2.54	0
D2V-L LS960 960h	8.76	0			2.44	0			2.39	0
W2V-L LV60k 960h	9.08	0			2.59	0			2.47	0
HB-L LV60k 960h	8.65	0			2.5	0			2.55	0
HB-XL LV60k 960h	8.41	0			2.49	0			2.36	0
W2V-L CV CV+1h	97.46	88.68			3.25	0			3.18	0
W2V-B None 960h	5.77	0			99.57	99.01			19.05	0
W2V-L None 960h	5.53	0			22.93	0			99.93	99.58

Table D.3: Targeted Character-level and Word-level success rate for adversarial attacks when varying the proxy and the target model within a pool of 13 models. The format is [Model]-[Size] [Unlabeled data] [Labeled data]. [Model] is equal to W2V (Wav2Vec2), D2V (Data2Vec), or HB (HuBERT). [Size] is equal to B (BASE), L (LARGE), or XL (XLARGE).

Model	Unlabeled data	Labeled data	Attack SNR	Attack success rate (word level)	
				targeted	untargeted
Wav2Vec2-LARGE	LV60k	LS960	30dB	88.0%	100%
			22dB	98.4%	100%
HuBERT-LARGE	LV60k	LS960	30dB	87.2%	100%
			22dB	98.5%	100%
Data2Vec-BASE	LS960	LS960	30dB	63.4%	100%
			22dB	92%	100%
Wav2Vec2-BASE	LS960	LS960	30dB	55.7%	100%
			22dB	62.9%	100%
Wav2Vec2-BASE	LS960	LS100	30dB	53.9%	100%
			22dB	59.5%	100%
Wav2Vec2-LARGE	LS960	LS960	30dB	50.7%	100%
			22dB	49.4%	100%
Data2Vec-LARGE	LS960	LS960	30dB	66%	100%
			22dB	86.4%	100%
HuBERT-XLARGE	LV60k	LS960	30dB	80.9%	100%
			22dB	95.5%	100%
UniSpeech-Sat-BASE	LS960	LS100	30dB	50.4%	100%
			22dB	62.4%	100%
WavLM-BASE	LV60k+VoxP+GS	LS100	30dB	21.7%	100%
			22dB	22.9%	100%
Wav2Vec2-LARGE	CV	CV+LS1	30dB	19.7%	100%
			22dB	36.1%	100%
M-CTC-Large	None	CV	30dB	7.5%	76.4%
			22dB	3.5%	83.4%
Speech2Text	None	LS960	30dB	7.3%	63.3%
			22dB	2.3%	74.6%
SB CRDNN	None	LS960	30dB	5.9%	86.39%
			22dB	1.5%	76.8%
SB Transformer	None	LS960	30dB	6.49%	90.56%
			22dB	1.2%	76.1%

Table D.4: Results of the transferred adversarial attack on different ASR models, with multiple Signal-Noise Ratios. The first three models correspond to the proxies used to generate the adversarial examples. On all other models, the inputs have been transferred directly. We report for each model how much unlabeled data was used for SSL pretraining and for ASR finetuning. We also report its Word-Error-Rate on the LibriSpeech test-clean set, and the targeted and untargeted word-level attack success rate (see section 6.3.2)

Language models are therefore more of a liability than a defense against attacks, and most likely so would be many tricks applied to an ASR model in order to improve its general performance.

Model	**Unlabeled data**	**Labeled data**	**Clean WER**		**Attack success rate (word level)**	
			w/o LM	with LM	w/o LM	with LM
Wav2Vec2-LARGE	LV60k	LS960	2.2%	2.0%	80.2%	88.0%
HuBERT-LARGE	LV60k	LS960	2.1%	1.9%	77.3%	87.2%
Data2Vec-BASE	LS960	LS960	3.2%	2.5%	51.7%	63.4%
Wav2Vec2-BASE	LS960	LS960	3.4%	2.6%	43.6%	55.7%
Wav2Vec2-BASE	LS960	LS100	6.2%	3.4%	41.8%	53.9%
Wav2Vec2-LARGE	LS960	LS960	2.8%	2.3%	41.4%	50.7%
Data2Vec-LARGE	LS960	LS960	2.2%	1.9%	56.9%	66%
HuBERT-XLARGE	LV60k	LS960	2.0%	1.8%	63.9%	80.9%
UniSpeech-Sat-BASE	LS960	LS100	6.4%	3.5%	39.5%	50.4%

Table D.5: Results of the transferred adversarial attack on different ASR models, with and without language models. We report for each model how much unlabeled data was used for SSL pretraining and for ASR finetuning. We also report its Word-Error-Rate on the LibriSpeech test-clean set, and the targeted word-level attack success rate (see section 6.3.2)

D.5.3 Effect of model regularization on transferability

As mentioned in section 6.3.1 we use regularization tricks like dropout in all proxy models when optimizing the adversarial perturbation. In Figure 6.2b we plot the loss on proxy and private models without that regularization, for comparison with Figure 6.2a. We observe that the loss degrades significantly on private models without regularization.

On the other hand, the loss on the proxy converges much faster in Figure 6.2b: removing model regularization makes for better, faster white-box attacks, at the cost of all transferability. To the extent of our knowledge, past work like Carlini and Wagner [2018] have not used regularization for generation, explaining why they report better white-box attacks than we do in terms of WER and SNR. However, as we have established above, applying regularization against standard ASR models does *not* lead to transferable adversarial examples: for that SSL pretraining is also required.

Appendix E

Supplementary material for chapter 7

E.1 Results on vanilla (non-adaptive) attacks

All the results we report in the main work are computed against the above adaptive attack, using gradient batches of size 16. In Table E.1 we report results obtained by the PGD attack on our trained defense, with $\sigma = 0.01$, with and without Expectation over Transformation. The WER is significantly lower using vanilla attacks, demonstrating why using adaptive attacks is necessary to correctly evaluate a defense. This also illustrates that our claims do **not** reflect obfuscation phenomena, but rather actual adversarial robustness.

Model, Smoothing	Adaptive	SNR-PGD					
		35	30	25	20	15	10
Trained defense, $\sigma = 0.02$	No	29	39	54	66	90	100
Trained defense, $\sigma = 0.02$	Yes	34	46	60	70	92	100

Table E.1: Word Error Rate (%) for Our defense on the first 100 utterances of the LibriSpeech clean test set under untargeted PGD attacks using various SNR. The first line corresponds to the vanilla attack, the second uses an adaptive attack that averages gradients on 16 forward+backward passes.

E.2 ROVER accuracy/time tradeoff

One drawback of ROVER voting is its important time consumption when using many inputs. This may be partially due to its third-party, black-box implementation that does not use GPU computation. However, in Figure E.1 we show that ROVER voting time increases superlinearly with the number of samples (where averaging and counting are of course linear): this is most likely an irreducible complexity of the algorithm. Using more than 50 iterations is not practically feasible[1]. This is why we use $N = 16$ in most of our experiments: even though more iterations may bring marginal WER improvement, this value enables us to improve performance substantially while

[1]It is, in fact, forbidden by default in the publicly available implementation

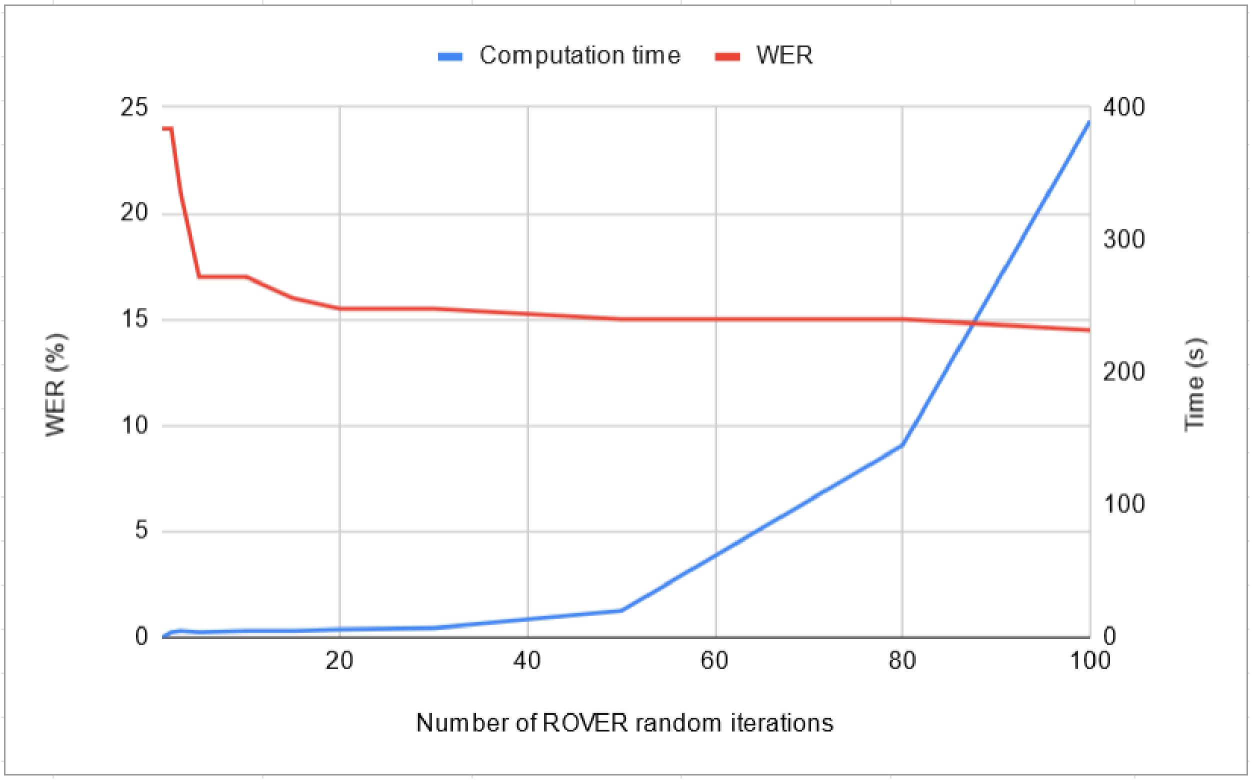


Figure E.1: Average ROVER Word-Error Rate and computation time (DeepSpeech2) for a single utterance when varying N.

keeping the voting time negligible. Besides, 16 inputs can typically be fed to the model in one batch, thus keeping the overall computation time low.

E.3 Certifying ASR smoothing

.

E.3.1 Robustness properties for classification

For multiclass classification, Cohen et al. [2019] are able to provide a robustness certificate based on the probability of the most probable class A: if

$$p_A = \mathbb{P}(f(x+\epsilon) = A) \tag{E.1}$$

and

$$p_B = \max_{B \neq A} \mathbb{P}(f(x+\epsilon) = B) \tag{E.2}$$

then $g(x+\delta) = A$ for all $\|\delta\|_2 < R$ with

$$R = \frac{\sigma}{2}(\phi^{-1}(p_A) - \phi^{-1}(p_B) \tag{E.3}$$

where g is the smoothed classifier:

$$g(x) = \arg \max_{k \in \{1,2,\ldots,m\}} \mathbb{P}(f(x+\epsilon) = k) \tag{E.4}$$

$\epsilon \sim \mathcal{N}(0, \sigma^2 I)$ and ϕ the standard gaussian CDF.

This result extends naturally from the binary classification case, which itself is a consequence of the Neyman-Pearson lemma [Neyman and Pearson, 1933]. In the case of ASR, reducing the problem to binary classification is not as trivial. We propose such a reduction by using thresholds on the evaluation metric d (typically the WER).

E.3.2 Robustness properties for ASR

Name f an ASR model. We assume given an *empirical target* sentence $\tilde{y}$ (different from the golden transcription y) and a threshold $k \in]0,1[$. We define the binary classifier $\tilde{f}$ as:

$$\tilde{f}(x) = \begin{cases} 1 & \text{if } d(f(x), \tilde{y}) < k \\ 0 & \text{otherwise} \end{cases} \tag{E.5}$$

And $\tilde{g}(x) = \arg\max_{c \in \{0,1\}} \mathbb{P}(\tilde{f}(x+\epsilon) = c)$. $\tilde{g}$ is informative only if $\tilde{g}(x) = 1$.

By immediate application of Cohen et al. [2019]'s result to $\tilde{f}$ we obtain the following guarantee:

Proposition E.1. $\tilde{g}(x+\delta) = 1$ *for all* $\|\delta\|_2 < R$ *with*

$$R = \frac{\sigma}{2}(\phi^{-1}(p) - \phi^{-1}(1-p)),\ p = \mathbb{P}(\tilde{f}(x) = 1) \tag{E.6}$$

Certification algorithm for sequential smoothing

This result allows us to use on $\tilde{f}$ the CERTIFY algorithm from Cohen et al. [2019] (section 3.2.2). We do not reproduce it here; the only change to our use case is that rather than generating a "top class" c_A based on counts, we use our ROVER prediction strategy to generate the "top transcription" t_A. A policy to estimate the bound k could perhaps be designed, or k can simply be fixed to a value that seems reasonable with respect to applications.

Proposition E.2. *With probability at least* $1-\alpha$ *over the randomness in CERTIFY, if CERTIFY returns a transcription* t_A *and a radius* R *(i.e. does not abstain), then the model predicts* t_A *at WER* $\leq k$ *within radius* R *around* x.

Feasibility

Despite our good experimental results, and while we showed in this section that robustness certification is *possible*, this algorithm requires estimating probabilities, which is computationally infeasible using a large ASR model. As Cohen et al. [2019] note, certification with informative bounds and confidence requires many samples (in the tens of thousands). We are unable to run this many forward loops per sentence[2] under reasonable time constraints. Nonetheless, these results

[2]Using one NVIDIA GTX 2080Ti

show a possible way to reduce ASR to finite-class classification for certification purposes and open a path toward guaranteeing the robustness of ASR models using faster architectures.

E.4 Additional experiments

In Table E.2 we report the results of both the PGD and CW attacks on our baselines and proposed models. We include models that are partially defended (without voting/augmented training/etc) to serve as an ablation study.

σ	Model, Smoothing	Nat.	SNR-PGD						CW		
			35	30	25	20	15	10	GT	TGT	SNR
0.0	Deepspeech2	8	63	89	100	100	100	100	100	0	27
	MP3 compression	9	59	71	79	100	100	100	100	3	16
0.01	AUG	12	50	64	82	100	100	100	100	0	19
	RS	34	58	66	84	95	100	100	100	13	15
	RS + AUG	11	30	46	57	82	100	100	100	11	14
	+ RS + ASNR	23	35	45	60	71	91	100	100	11	10
	RS + ROVER	29	51	64	81	93	100	100	100	7	15
	RS + σ-AUG + ROVER	9	31	43	59	81	100	100	100	8	14
	RS + ASNR + ROVER	20	34	46	60	70	92	100	100	4	10
0.02	AUG	13	46	63	83	100	100	100	100	0	15
	RS	72	79	82	87	96	97	100	100	12	10
	RS + AUG	19	35	46	64	77	100	100	100	9	8
	RS + ASNR	41	44	55	62	74	84	100	100	13	5
	RS + ROVER	62	72	77	84	93	98	100	100	7	10
	RS + AUG + ROVER	14	34	46	60	77	100	100	100	6	8
	RS + ASNR + ROVER	36	40	46	56	71	83	100	100	6	5

Table E.2: Word Error Rate (%) for Deepspeech2 on the first 100 utterances of the LibriSpeech clean test set under various attacks and defenses. + AUG stands for gaussian augmentation of deviation σ in training - the same deviation used at inference. ASNR means A priori SNR filtering of inputs. + ROVER refers to the ROVER voting strategy using 16 forward passes. For the PGD attack, we specify the minimal SNR we use as L_∞ bound. For the unbounded CW attack, we report both the WER on the ground truth (GT) and the attack target (TGT), and the SNR required to achieve it. All attacks run on models using smoothing are adaptive and average gradients on 16 forward+backward passes.