Adversarial robustness via robust low rank representations

Pranjal Awasthi[*] Himanshu Jain[†] Ankit Singh Rawat[‡]
Aravindan Vijayaraghavan[§]

Abstract

Adversarial robustness measures the susceptibility of a classifier to imperceptible perturbations made to the inputs at test time. In this work we highlight the benefits of natural low rank representations that often exist for real data such as images, for training neural networks with certified robustness guarantees.

Our first contribution is for certified robustness to perturbations measured in ℓ_2 norm. We exploit low rank data representations to provide improved guarantees over state-of-the-art randomized smoothing-based approaches on standard benchmark datasets such as CIFAR-10 and CIFAR-100.

Our second contribution is for the more challenging setting of certified robustness to perturbations measured in ℓ_∞ norm. We demonstrate empirically that natural low rank representations have inherent robustness properties, that can be leveraged to provide significantly better guarantees for certified robustness to ℓ_∞ perturbations in those representations. Our certificate of ℓ_∞ robustness relies on a natural quantity involving the $\infty \to 2$ matrix operator norm associated with the representation, to translate robustness guarantees from ℓ_2 to ℓ_∞ perturbations. A key technical ingredient for our certification guarantees is a fast algorithm with provable guarantees based on the multiplicative weights update method to provide upper bounds on the above matrix norm. Our algorithmic guarantees improve upon the state of the art for this problem, and may be of independent interest.

1 Introduction

It is now well established across several domains like images, audio and natural language, that small input perturbations that are imperceptible to humans can fool deep neural networks at test time [SZS+13, BCM+13, CW18, ERLD17]. This phenomenon known as *adversarial robustness* has led to flurry of research in recent years (see Section 5 for a discussion of related work). Following most prior work in this area [GSS14, MMS+17, ZYJ+19, SNG+19, WRK20], we will study the setting where adversarial perturbations to an input x are measured in an ℓ_p norm ($p = 2$ or $p = \infty$).

In this work, we study methods for *certified adversarial robustness* in the framework developed in [LAG+19, CRK19]. The goal is to output a classifier f that on input $x \in \mathbb{R}^n$ outputs a prediction y in the label space $\mathcal{Y}$, along with a certified radius $r_f(x)$. The classifier is guaranteed to be robust

[*]Google Research and Department of Computer Science, Rutgers University. The author acknowledges support from the NSF HDR TRIPODS award CCF-1934924. pranjal.awasthi@rutgers.edu.

[†]Google Research

[‡]Google Research

[§]Department of Computer Science, Northwestern University. Supported by the National Science Foundation (NSF) under Grant No. CCF-1652491 and CCF-1637585. aravindv@northwestern.edu.

at x up to the radius $r_f(x)$ (with high probability), i.e., $\forall z : \|z\|_p \leq r_f(x), f(x+z) = f(x)$. For an ℓ_p norm and $\varepsilon > 0$, the certified accuracy of a classifier f is defined as

$$\mathrm{acc}_{\varepsilon}^{(\ell_p)}(f) = \mathop{\mathbb{P}}_{(x,y)\sim D}\left[f(x) = y \text{ and } r_f(x) \geq \varepsilon\right], \tag{1}$$

where D is the underlying data distribution generating test inputs. We call the radius $r_f(x)$ returned by the classifier as the *certified radius* on x. When $\varepsilon = 0$ this is the natural accuracy of f.

For certified adversarial robustness to ℓ_2 perturbations, the *randomized smoothing* procedure proposed in [LAG+19, CRK19] is a simple and efficient method that can be applied to *any* neural network. Randomized smoothing works by creating a smoothed version of a given classifier by adding Gaussian noise to the inputs (see Section 2). The smoothed classifier exhibits certain Lipschitzness properties, and one can derive good certified robustness guarantees from it. The study of randomized smoothing for certified ℓ_2 robustness is an active research area and the current best guarantees are obtained by incorporating the smoothed classifier into the training process [SYL+19] (see Section 5).

It seems much more challenging to obtain certified adversarial robustness to ℓ_∞ perturbations [RSL18, GDS+18, WK18]. In particular, the design of a procedure akin to randomized smoothing has been difficult to achieve for ℓ_∞ perturbations. One approach to obtain certified ℓ_∞ robustness is to translate a certified radius guarantee of ε for ℓ_2 perturbations (via randomized smoothing) into an $\varepsilon/\sqrt{n}$ certified radius guarantee for ℓ_∞ perturbations; here n is the dimensionality of the ambient space. Furthermore, recent work [BDMZ20, YDH+20, KLGF20] has established lower bounds proving that randomized smoothing based methods cannot break the above $\sqrt{n}$ barrier for ℓ_∞ robustness in the worst case.

However real data such as images are not worst case and often exhibits a natural low rank structure. In this work we show how we can leverage such natural low-rank representations for the data, in order to design algorithms based on randomized smoothing with improved certified robustness guarantees for both ℓ_2 and ℓ_∞ perturbations.

Our Contributions. We now describe our main contributions.

Improved certified ℓ_2 robustness: Our first contribution is to design new smoothed classifiers for achieving certified robustness to ℓ_2 perturbations. These classifiers achieve improved tradeoffs between natural accuracy and certified accuracy at higher radii. We achieve this by leveraging the existence of good low-rank representation for the data. We modify the randomized smoothing approach to instead selectively inject more noise along certain directions, without compromising the accuracy of the classifier. The large amount of noise leads to classifier that is less sensitive to ℓ_2 perturbations, and hence achieves higher certified accuracy across a wide range of radii. We empirically demonstrate the improvements obtained by our approach on image data in Section 2.

Fast algorithms for translating certified robustness guarantees from ℓ_2 to ℓ_∞: For the more challenging setting of ℓ_∞ robustness we consider classifiers of the form $f(Px)$ where P is an arbitrary linear map, and f represents an arbitrary neural network. When translating certified robustness guarantees for ℓ_2 perturbations to obtain guarantees for ℓ_∞ perturbations, the loss incurred is captured by the $\infty \to 2$ operator norm of matrix P. While computing this operator norm is NP-hard, we design a fast approximate algorithm based on the multiplicative weights update method with provable guarantees. Our algorithmic guarantees give significant improvements over the best known bounds [AHK05, Kal07] for this problem and may be of independent interest (see Section 3.1).

Certified ℓ_∞ robustness in natural data representations: Real data such as images have natural representations that are often used in image processing e.g., via the Discrete Cosine Transform (DCT). Via an empirical study we highlight the need for achieving ℓ_∞ robustness in the DCT basis. More importantly, we demonstrate that the representation in the DCT basis is robust, i.e., there exist low rank projections that capture most of the signal in the data and that at the same time have small $\infty \to 2$ operator norm.[1] We develop a fast heuristic based on sparse PCA to find such robust projections. When combined with our multiplicative weights based algorithm, this leads to a new training procedure based on randomized smoothing. Our procedure can be applied to any network architecture and provides stronger guarantees on robustness to ℓ_∞ perturbations in the DCT basis.

2 Certified Robustness to ℓ_2 Perturbations

We build upon the *randomized smoothing* technique proposed in [LAG+19, CRK19] and further developed in [SYL+19]. Consider a multiclass classification problem and a classifier $f : \mathbb{R}^n \to \mathcal{Y}$, where $\mathcal{Y}$ is the label set. Given f, randomized smoothing produces a smoothed classifier g where

$$g(x) = \arg\max_{y \in \mathcal{Y}} \mathbb{P}(f(x+\delta) = y). \tag{2}$$

Here $\delta \sim N(0, \sigma^2 I)$ is the Gaussian noise added. The following proposition holds.

Proposition 2.1 ([LAG+19, CRK19])**.** *Given a classifier f, let g be its smoothed version as defined in (2) above. On an input x, and for $\delta \sim N(0, \sigma^2 I)$ define $y_A := \arg\max_y \mathbb{P}(f(x+\delta) = y)$, and let $p_A = \mathbb{P}(f(x+\delta) = y_A)$. Then the prediction of g at x is unchanged up to ℓ_2 perturbations of radius*

$$r(x) = \frac{\sigma}{2}(\Phi^{-1}(p_A) - \Phi^{-1}(p_B)). \tag{3}$$

Here $p_B = \max_{y \neq y_A} \mathbb{P}(f(x+\delta) = y)$ and Φ^{-1} is the inverse of the standard Gaussian CDF.

Hence, randomized smoothing provides a fast method to certify the robustness of any given classifier on various inputs. In order to get robustness to large perturbations it is desirable to choose the noise magnitude σ as large as possible. However, there is a natural tradeoff between the amount of noise added and the natural accuracy of the classifier. As an example consider an input $x \in \mathbb{R}^n$ of ℓ_2 length $\sqrt{n}$. If σ is the average amount of noise added then one is restricted to choosing σ to be a small constant in order for the noise to not overwhelm the signal.

However, it is well known that natural data such as images are low dimensional in nature. Figure 5 in Appendix A shows that for the CIFAR-10 and CIFAR-100 datasets, even when projected, via PCA, onto 200 dimensions, the reconstruction error remains small. If the input is close to an r-dimensional subspace, then it is natural to add noise only within the subspace for smoothing. Formally, let Π be the projection matrix on to an r-dimensional subspace and x be such that $\|\Pi x\|_2 \approx \|x\|_2 = \sqrt{n}$. For $\delta \sim N(0, \sigma^2 I)$ we have $\|\Pi\delta\|_2 \approx \sigma\sqrt{r}$. Hence if we only add noise within the subspace, then σ can be as large as $\sqrt{n/r}$ as opposed to a constant without significantly affecting the natural accuracy.

[1]This is also true for domains such as audio in the DCT basis. See Appendix F for experimental evidence.

We formalize this into an efficient training algorithm as follows: we take a base classifier/neural network $f(x)$ and replace it with the smoothed classifier $g_\Pi(x)$ where

$$g_\Pi(x) = \arg\max_{y \in \mathcal{Y}} \mathbb{P}(f(\Pi x + \delta_\Pi) = y). \tag{4}$$

where Π is a projection matrix onto an r-dimensional subspace and δ_Π is a standard Gaussian of variance σ^2 that lies within Π. For data such as images, good projections Π can be obtained via methods like PCA. Furthermore, certifying the robustness of our proposed smoothed classifier can be easily incorporated into existing pipelines for adversarial training with minimal overhead. In particular using the rotational symmetry of Gaussian distributions it is easy to show the following

Proposition 2.2. *Given a base classifier $f : \mathbb{R}^n \to \mathcal{Y}$ and a projection matrix Π, on any input x, the smoothed classifier $g_\Pi(x)$ as defined in (4) is equivalent to the classifier given by*

$$\tilde{g}_\Pi(x) = \arg\max_{y \in \mathcal{Y}} \mathbb{P}(f(\Pi(x + \delta)) = y). \tag{5}$$

Here δ is a standard Gaussian of variance σ^2 in every direction.

Hence constructing our proposed smoothed classifier simply requires adding a linear transformation layer to any existing network architecture before training via randomized smoothing. We propose to train the smoothed classifier as defined in (5) by minimizing its adversarial standard cross entropy loss as proposed in [MMS+17]. However, since dealing with $\arg\max$ is hard from an optimization point of view, we follow the approach of [SYL+19] and instead minimize the cross entropy loss of the following soft classifier

$$G_\Pi(x) = \mathop{\mathbb{E}}_{\delta \sim N(0,\sigma^2 I)} [f(\Pi(x + \delta))]. \tag{6}$$

This leads to the following objective where ℓ_{ce} is the standard cross-entropy objective and $\varepsilon > 0$ is perturbation radius chosen for the training procedure.

$$\arg\min_f \mathop{\mathbb{E}}_{(x,y)} \Big[\max_{z : \|z\|_2 \le \varepsilon} \ell_{ce}(G_\Pi(x + z), y) \Big]. \tag{7}$$

Following [MMS+17, SYL+19], the inner maximization problem of finding adversarial perturbations is solved via projected gradient descent (PGD), and given the adversarial perturbations, the outer minimization uses stochastic gradient descent. Overall, this leads to the following training procedure.

Algorithm 1 Adversarial training via projections

```
1: function RobustTrain(training data $(x_1, y_1), \ldots, (x_m, y_m)$, subspace rank $r$, base noise magnitude $\sigma$, $\lambda \in [0,1]$, number of steps $T$, mini batch size $b$)
2:     Perform PCA on (unlabeled) data matrix $A \in \mathbb{R}^{n \times m}$ to obtain a rank-$r$ projection matrix $\Pi$.
3:     Set $G_\Pi$ as in (6) with $\sigma = \lambda\sqrt{n/r}$.
4:     for $t = 1, \ldots, T$ do
5:         Obtain a mini batch of $b$ examples $(x_{t_1}, y_{t_1}), \ldots, (x_{t_b}, y_{t_b})$.
6:         For each $x_{t_i}$ use projected gradient ascent on inner maximization in (7) to get $x'_{t_i}$.
7:         Given perturbed examples $\{(x'_{t_i}, y_{t_i})\}_{i \in [b]}$, update network parameters via SGD.
8:     Output the smoothed classifier $\tilde{g}_\Pi(x)$.
```

Empirical Evaluation. We compare Algorithm 1 with the algorithm of [SYL+19] for various values of σ and ε (used for training to optimize (7)). We choose $\varepsilon \in \{0.25, 0.5, 0.75, 1.0\}$ and for each ε we choose the value of σ as described in [SYL+19]. In each case, we train the classifier proposed in [SYL+19] using a noise magnitude σ, and we train our proposed smoothed classifier using higher noise values of $\lambda\sigma\sqrt{n/r}$, where λ is a parameter that we vary. In all experiments, we train a ResNet-32 network on the CIFAR-10 dataset by optimizing (7). Figure 1 shows a comparison of certified accuracies for different radii and different values of λ. See Appendix A for a description of the hyperparameters and additional experiments.

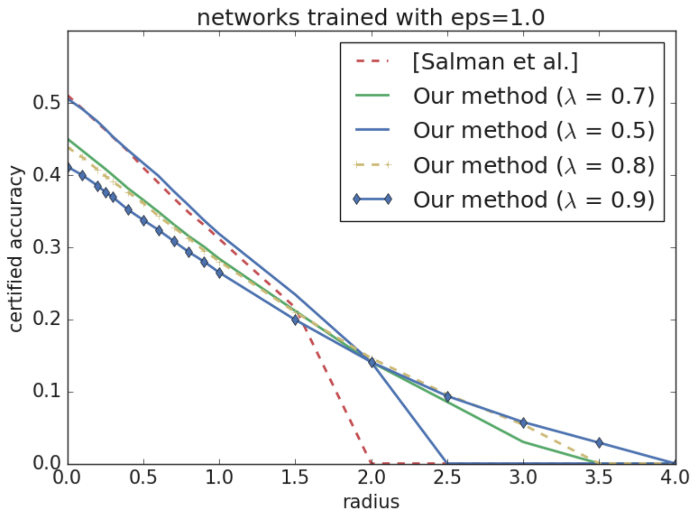

Figure 1: Plot of certified accuracy achieved at different radii when different values of λ are chosen to optimize the smoothed classifier in (5). We compare with the method of [SYL+19] The plot is obtained by training a ResNet-32 architecture on CIFAR-10 with $\varepsilon = 1.0$. Note that the y-intercept of each curve represents the natural accuracy of the corresponding classifier.

As can be seen from the figure, varying the value of λ lets us tradeoff lower accuracy at small values of the radius for a significant gain in certified accuracy at higher radii as compared to the method of [SYL+19]. In particular we find that choosing values of λ close to 0.5 leads to networks that can certify accuracy at much higher radii with minimal to no loss in the natural accuracy as compared to the approach of [SYL+19]. In Figure 2 we present the result of our training procedure for various values of ε and σ and compare with the ℓ_2 smoothing method of [SYL+19] on the CIFAR-10 and CIFAR-100 datasets. In each case, to obtain the projection matrix Π we perform a PCA onto each image channel separately and use the top 200 principal components to obtain the projection matrix Π. For both datasets, our trained networks outperform the method of [SYL+19] across a large range of radius values. In particular, for higher values of radius (say, $\gtrsim 0.5$) our method achieves a desired certified accuracy with significantly higher natural accuracy as compared to the method of [SYL+19]. For instance in the CIFAR-10 dataset, at a radius of 1.0 and a desired certified accuracy of at least 0.35, the method of [SYL+19] achieves a natural accuracy of ≈ 0.5 (yellow dotted curve at radius 0). In contrast our method achieves the same with a natural accuracy of ≈ 0.65 (green solid curve at radius 0). On the other hand, at very small radius values the method of [SYL+19] is better. This is expected as we suffer a small loss in natural accuracy due to the PCA step in Algorithm 1.

3 Methods for Certified ℓ_∞ Robustness

We now describe our algorithms for the more challenging problem of certified robustness to ℓ_∞ perturbations in a given basis or representation. Our approach is to leverage the existence of good representations of natural data measured by a certain robustness parameter, and translate

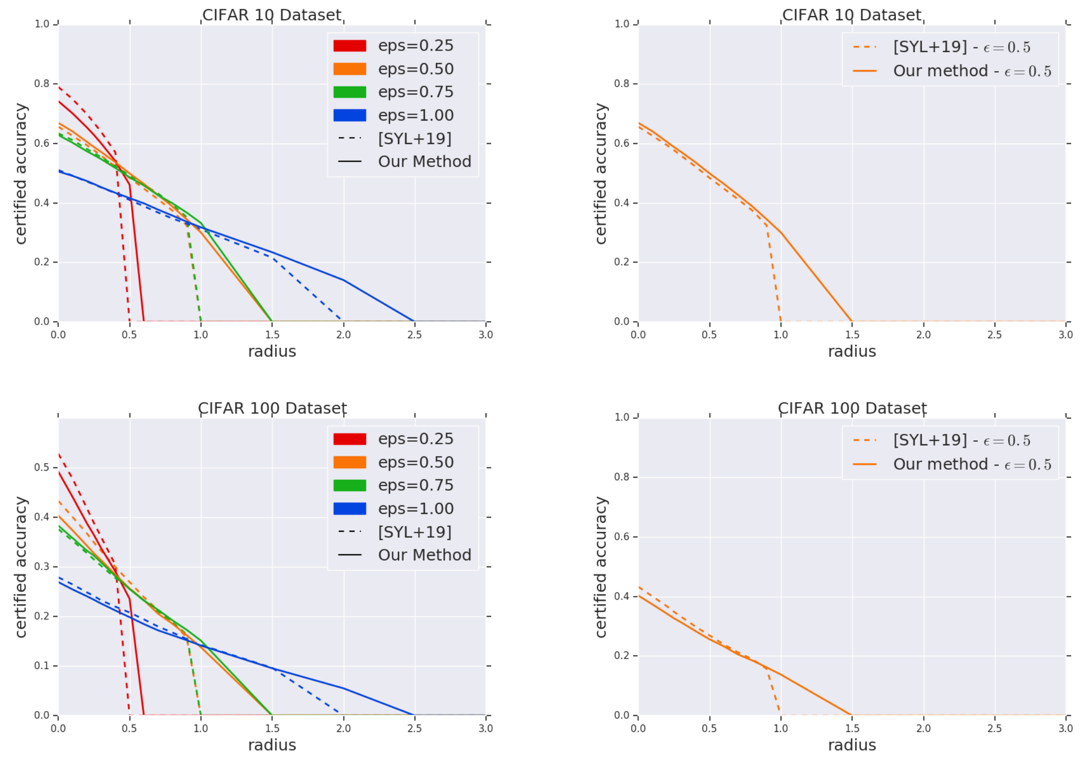

Figure 2: A comparison of certified radius guarantees obtained via Algorithm 1 as compared to the approach of [SYL+19]. The x-axis is the radius, and the y-axis represents the certified accuracy. The top row describe results for the CIFAR-10 dataset – (left) certified accuracies for various values of ε, (right) for $\varepsilon = 0.5$. Similarly, the bottom row describe the results for the CIFAR-100 dataset.

ℓ_2 robustness guarantees from Section 2 to get certified ℓ_∞ guarantees. Consider $f : \mathbb{R}^n \to \mathcal{Y}$ and $g(x) := f(\Pi x)$, where $\Pi \in \mathbb{R}^{n \times n}$ represents a projection matrix. The certified accuracy of g satisfies

$$\forall \varepsilon > 0, \quad \mathrm{acc}^{(\ell_\infty)}_{\varepsilon'}(g) \geq \mathrm{acc}^{(\ell_2)}_{\varepsilon}(g) \text{ for } 0 \leq \varepsilon' \leq \varepsilon / \|\Pi\|_{\infty \to 2}, \quad \text{where } \|\Pi\|_{\infty \to 2} = \max_{x : \|x\|_\infty \leq 1} \|\Pi x\|_2$$

is the $\infty \to 2$ operator norm of Π and represents a robustness parameter (see Proposition B.2 for a formal claim). Hence, to translate guarantees from ℓ_2 to ℓ_∞, we look for *robust* projections Π that have small $\infty \to 2$ operator norm. Our approach is inspired by the recent theoretical work of [ACCV19], and finds a low-dimensional representation given by an (orthogonal) projection matrix Π for the data that has small $\|\Pi\|_{\infty \to 2}$. By matrix norm duality, there is a nice characterization for $\|\Pi\|_{\infty \to 2}$ as the maximum ℓ_1 norm among Euclidean unit vectors in the subspace of Π (this is a notion of sparsity of the vectors). For a rank r projector Π, the range of values taken by $\|\Pi\|_{\infty \to 2}$ is $[\sqrt{r}, \sqrt{n}]$. Hence if the projection is not low-rank and sparse, $\|\Pi\|_{\infty \to 2}$ could be as large as $\sqrt{n}$ (e.g., when $\Pi = I$). This is consistent with the loss of $\sqrt{n}$ factor in robustness radius to ℓ_∞ perturbations for general datasets [BDMZ20, YDH+20]. Moreover as we have seen in Section 2, good low-rank representations of the data also give stronger certified ℓ_2 robustness guarantees (and in turn, stronger certified ℓ_∞ guarantees).

Our goal is to find a good robust rank-r projection of the data if it exists. We propose a heuristic based on sparse PCA [MBPS09] to find a robust low rank projection with low reconstruction error (see Section 3.2). Since we aim for certified robustness, an important step in this approach is to compute and certify an upper bound on $\|\Pi\|_{\infty\to 2}$, for our candidate projection Π. This is an NP-hard problem, related to computing the famous Grothendieck norm [AN04]. We describe a new, scalable, approximate algorithm for computing upper bounds on $\|\Pi\|_{\infty\to 2}$ with provable guarantees.

3.1 Certifying the $\infty \to 2$ operator norm.

Our fast algorithm is based on the multiplicative weights update (MWU) method for approximately solving a natural semi-definite programming (SDP) relaxation, and produce a good upper bound on $\|\Pi\|_{\infty\to 2}$. Our upper bound also comes with a certificate from the dual SDP i.e., a short proof of the correctness of the upper bound. Given a candidate Π our algorithm will compute an upper bound for $\|\Pi\|_{\infty\to 1}$ which by matrix norm duality satisfies

$$\|\Pi\|^2_{\infty\to 2} = \|\Pi\|_{\infty\to 1} = \max_x x^\top \Pi x \;\; \text{subject to } \|x\|_\infty \le 1. \tag{8}$$

In fact our algorithm will work for the following more general problem of Quadratic Programming:

$$\text{Given a symmetric matrix } M \text{ with } \; \forall i \in [n]: \; M_{ii} \ge 0, \qquad \max_{x:\|x\|_\infty \le 1} x^\top M x. \tag{9}$$

The standard SDP relaxation for the problem (see (12) in Appendix C.1), has primal variables represented by the positive semi-definite (PSD) matrix $X \in \mathbb{R}^{n\times n}$ satisfying constraints $X_{ii} \le 1$ for each $i \in [n]$. The SDP dual of this relaxation (given in (14) of Appendix C.1) has variables $y_1, \dots, y_n \ge 0$ corresponding to the n constraints in the primal SDP. Since the SDP is a valid relaxation for (9), it provides an upper bound for $\infty \to 1$ operator norm[2] . Classical results show that it is always within a factor of $\pi/2$ of the actual value of $\|M\|_{\infty\to 1}$ [Nes98, AMMN06]. However, it is computationally intensive to solve the SDP using off-the-shelf SDP solvers (even for CIFAR-10 images, X is 1024×1024). We design a fast algorithm based on the *multiplicative weight update* (MWU) framework [KL96, AHK05].

Description of the algorithm Our algorithm differs slightly from the standard MWU approach for solving the above SDP. The algorithm below takes as input a matrix M and always returns a valid upper bound upbd_{min} on the SDP value, along with a dual feasible solution y_{ubmin} that can act as a certificate, and a candidate primal solution X that attains the same value (and is potentially feasible). Theorem 3.1 proves that for the right setting of parameters, particularly the number of iterations $T_f = O(n \log n/\delta^3)$, the solution X is also guaranteed to be feasible up to small error $\delta > 0$.

[2] A fast algorithm that potentially finds a local optimum for the problem will not suffice for our purposes; we need an upper bound on the global optimum.

Algorithm 2 Fast Certification of $\infty \to 1$ norm and Quadratic Programming

```
1: function CertifySDP($M \in \mathbb{R}^{n\times n}$, iteration bound $T_f$, slack $\delta$, damping $\rho$)
2:     Initialize $\alpha = (1,1,\dots,1) \in \mathbb{R}^n$. primal $X = 0$, dual $y = 0^n$, $\text{upbd}_{\min} = \infty$, $y_{\text{ubmin}} = 0^n$.
3:     for $t = 0, 1 \dots, T$ do
4:         $\tilde{\alpha} \leftarrow (1-\delta)\alpha + \delta(1,1,\dots,1)$.
5:         $\lambda \leftarrow$ max-eigenvalue($\text{diag}(\tilde{\alpha})^{-1/2} M \text{diag}(\tilde{\alpha})^{-1/2}$) and $u \in \mathbb{R}^n$ be its eigenvector.
6:         $v \leftarrow \sqrt{n}\cdot \text{diag}(\tilde{\alpha})^{-1/2}u$, $y \leftarrow \frac{1}{t+1}(ty + \lambda\tilde{\alpha})$, $X \leftarrow \frac{1}{t+1}(tX + vv^\top)$.
7:         if $\|v\|_\infty \le 1+\delta$ or $\max_i X_{ii} \le 1+\delta$ then, do early stop and return appropriate values.
8:         Update $\forall i \in [n], \alpha(i) \leftarrow \alpha(i)\exp\left(\frac{\delta}{\rho}(v(i)^2 - 1)\right)$, and renormalize s.t. $\sum_{i=1}^n \alpha(i) = n$.
9:         if $\text{upbd}_{\min} > n\lambda$, then set $\text{upbd}_{\min} = n\lambda$ and $y_{\text{ubmin}} = y$.
10:    Output $\text{upbd}_{\min}$, dual solution $y_{\text{ubmin}}$, and primal candidate $X$.
```

Recall that from (8) an estimate of the the $\infty \to 1$ norm immediately translates to an estimate of the $\infty \to 2$ norm. In the above algorithm, there are weights given by α for n different constraints of the form $X_{ii} \le 1$. At each iteration, the algorithm maximizes the objective subject to *one* constraint of the form $\sum_i \tilde{\alpha}_i X_{ii} \le n$, where $\tilde{\alpha}$ involves a small correction to α that is crucial to ensure the run-time guarantees. The maximization is done using a maximum eigenvalue/eigenvector computation. The weights α are then updated using a multiplicative update based on the violation of the solution found in the current iterate. The damping factor ρ determines the rate of progress of the iterations – the smaller the value of ρ the faster the progress, but a very small value may lead to oscillations. A more aggressive choice of ρ compared to the one in Theorem 3.1 seems to work well in practice. Finally, we remark that for *every* choice of α and ρ we get a valid upper bound (due to dual feasibility). We show the following guarantee for our algorithm for the more general problem of (9).

Theorem 3.1. *Suppose $\delta > 0$, and M be any symmetric matrix with $M_{ii} \ge 0$ $\forall i \in [n]$. For any $\alpha \in \mathbb{R}^n_{\ge 0}$ with $\sum_{i=1}^n \alpha(i) = n$, if $\lambda = \lambda_{\max}\Big((diag(\alpha)^{-1/2} M\, diag(\alpha)^{-1/2})\Big)$, then $y = \lambda\alpha$ is feasible for the dual SDP and gives a valid upper bound of $n\lambda$ on the objective value for the SDP relaxation to (9). Moreover Algorithm 2 on input M, with parameters δ and $\rho = O(n/\delta)$ after $T = O(n \log n/\delta^3)$ iterations finds a feasible SDP solution $\widehat{X} \succeq 0$ and a feasible dual solution $\widehat{y} \in \mathbb{R}^n$ that both sandwich the optimal SDP value within a $1+\delta$ factor.*

(See Prop C.1 and Theorem C.2 in Appendix C.1 for formal statements along with proofs.) Each iteration only involves a single maximum eigenvalue computation, which can be done up to $(1+\delta)$ accuracy in $T_{eig} = \tilde{O}(m/\delta)$ time where m is the number of non-zeros in M (see e.g., [KL96]). To the best of our knowledge, this gives significant improvements over the prior best bound of $\tilde{O}(n^{1.5}m/\delta^{2.5})$ runtime for solving the above SDP [AHK05, Kal07]. Our algorithm and analysis differs from the general MWU framework [Kal07] by treating the objective differently from the constraints so that the "width parameter" does not depend on the objective. A crucial step in our algorithm and proof is to add a correction term of $O(\delta)$ to the weights in each step that ensures that the potential violation of each constraint in an iteration is bounded. In addition our algorithm is more scalable than existing off-the-shelf methods. See Appendix C.1 and E for details and comparisons.

3.2 Finding Robust Low-rank Representations

We now show how to find a good low-rank robust projections when it exists in the given representation. Given a dataset A, our goal is to find a (low-rank) projection Π that gets small reconstruction error $\|A - \Pi A\|_F^2$, while ensuring that $\|\Pi\|_{\infty\to 2}$ is small. Awasthi et al. [ACCV19] formulate this as an optimization problem that is NP-hard, but show polynomial time algorithms based on the Ellipsoid algorithm that gives constant factor approximations. However, the algorithm is impractical in practice because of the Ellipsoid algorithm, and the separation oracle used by it (that in turn involves solving an SDP). We instead use the connection to sparsity described in Section 3 to devise a much faster heuristic based on sparse PCA to find a good projection Π (see (B.4) in the appendix for a formal justification). We just use an off-the-shelf heuristic for sparse PCA (the scikit-learn sparse PCA implementation [PVG+11] based on [MBPS09]), along with our certification procedure Algorithm 2).

Algorithm 3 Find a Robust Projection

```
1: function RobustProjection(data $A \in R^{m\times n}$, rank $k$, reconstruction error $\delta$)
2:     Set $M := (A^\top A)/\mathrm{tr}(A^\top A)$. Initialize $\widehat{\Pi} \leftarrow \emptyset, \widehat{\kappa} = \infty$.
3:     for different values of $r \leq k$ do
4:         Find $r$-PCA of the $M$ to get a rank $r$ projection $\Pi_1$. $M' \leftarrow M - \Pi_1 M \Pi_1$
5:         Run sparse PCA on $M'$ to find a rank $(k-r)$ projection $\Pi_2$. Set $\Pi = \Pi_1 + \Pi_2$.
6:         Run CertifySDP$(\Pi, \delta = 1/4, \rho)$ to get an upper bound $\kappa$.
7:         if $\kappa < \widehat{\kappa}$ and if $\langle M, I - \Pi\rangle \leq \delta$ then
8:             $\widehat{\Pi} \leftarrow \Pi$, $\widehat{\kappa} = \kappa$.
9:     Output $\widehat{\Pi}$, $\widehat{\kappa}$.
```

4 Training Certified ℓ_∞ Robust Networks in Natural Representations.

Building upon our theoretical results from the previous section we now demonstrate that for natural representations, one can indeed achieve better certified robustness to ℓ_∞ perturbations by translating guarantees from certified ℓ_2 robustness. We focus on image data, and study the representation of images in the DCT basis. Before we describe the details of our training and certification procedure for ℓ_∞ robustness, we provide further empirical evidence that imperceptibility in natural representations such as the DCT basis is a desirable property.

Study of imperceptibility in DCT basis. We argue that for adversarial perturbations to be imperceptible to humans they should be of small magnitude in the DCT representation (perhaps in addition to being small in the pixel basis). We take images from the CIFAR-10 dataset and transform them into the DCT basis. We then add sparse random perturbations to them. In particular, for a sparsity parameter k, we pick k coordinates in the DCT basis at random and add a random perturbation with ℓ_∞ norm of $c_k/\sqrt{k}$ where $c_k \approx \varepsilon\sqrt{n}$ is chosen such that the perturbed images are $\varepsilon \leq 0.09$ away from the unperturbed images in the pixel space. Notice that for small values of k, a perturbation of large ℓ_∞ norm is added. Figure 3 visualizes the perturbed images for different values of k. As seen, large perturbations in the DCT basis lead to visually perceptible changes, even if they

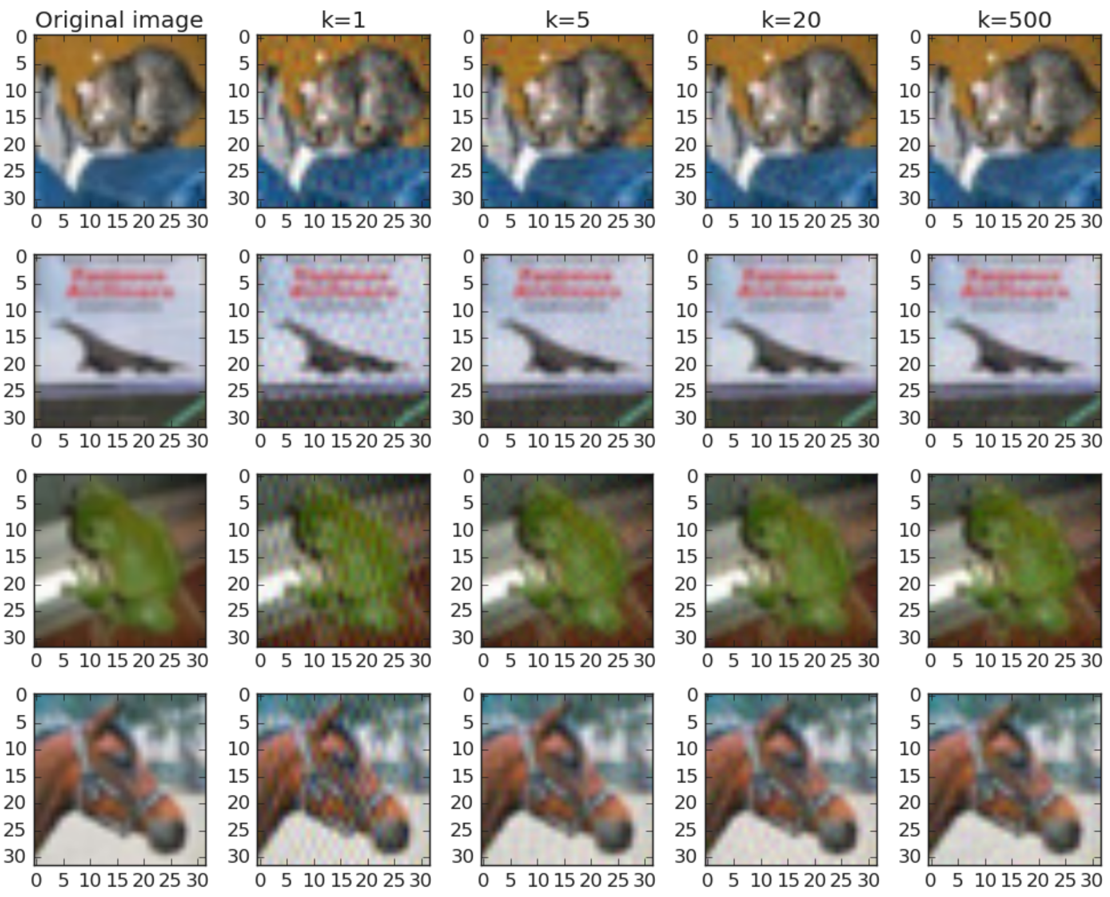

Figure 3: Original images (leftmost) from the CIFAR-10 dataset and their perturbed versions when sparse random perturbations are added in the DCT basis with sparsity k. Large perturbations in the DCT basis (e.g., $k = 1$) lead to perceptible changes in the pixel space though they are ℓ_∞ perturbations of $\varepsilon \leq 0.09$. As k increases the imperceptibility of the perturbed images improves.

are ($\varepsilon \leq 0.09$-)close in the pixel basis. For comparison we also include in Appendix D imperceptible adversarial examples for these images that were generated via the PGD based method of [MMS+17] on a ResNet-32 network trained on the CIFAR-10 dataset for robustness to ℓ_∞ perturbations of magnitude $\varepsilon = 0.09$. This further motivates studying robustness in natural data representations.

Training certified ℓ_∞ robust networks in the DCT basis. The methods developed in Section 3 and 2 together give algorithms for training classifiers with certified ℓ_∞ robustness. However while we want ℓ_∞ robustness in a different representation $\mathcal{U}$ (e.g., in the DCT basis), it may still be more convenient to use off-the-shelf methods for performing the training in the original representation $\mathcal{X} \subseteq \mathbb{R}^n$ (e.g., pixel representation). Let the orthogonal matrix $O \in \mathbb{R}^{n \times n}$ represent the DCT transformation. Consider an input $x \in \mathcal{X}$ and let $u = Ox \in \mathcal{U}$ be its DCT representation, where $\mathcal{U}$ is the space of images in the DCT basis. It is easy to see that functions $f : \mathcal{X} \to \mathcal{Y}$ and $g : \mathcal{U} \to \mathcal{Y}$ given by $g(u) := f(O^{-1}u) = f(x)$ have the same certified accuracy to ℓ_2 perturbations. Moreover if the classifier g satisfies $g(u) = g(\Pi u)$ for some projection Π, the robust accuracy of g to

ℓ_∞ perturbations in the representation $\mathcal{U}$ satisfies (see Proposition B.3 in appendix)

$$\mathrm{acc}_{\varepsilon'}^{(\ell_\infty)}(g) \geq \mathrm{acc}_{\varepsilon}^{(\ell_2)}(f), \text{ for any } \varepsilon > 0, 0 \leq \varepsilon' \leq \varepsilon/\|\Pi\|_{\infty\to 2}. \tag{10}$$

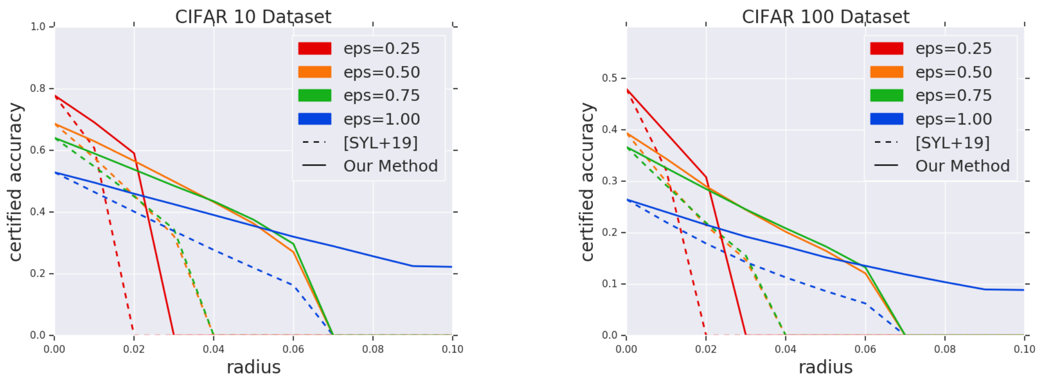

Figure 4: A comparison of certified ℓ_∞ accuracy (y-axis) in the DCT basis of our method to that of [SYL+19], for different values of ε and with different certified radii on the x-axis, for $\lambda = 0.5$. The left and right plots describe results for the CIFAR-10 and the CIFAR-100 datasets respectively.

Experimental Data. We evaluate our approach on the CIFAR-10 and CIFAR-100 datasets. From (10), it is sufficient to train a classifier in the original pixel space with an appropriate projection $\Pi' = O\Pi$. Hence, we train a smoothed classifier as defined in (2) using Algorithm 1. To obtain the required Π, we first use the sparse PCA based heuristic (Algorithm 3) to find a projection matrix of rank 200 for the three image channels separately. We then use Algorithm 2 to compute upper bounds on the $\infty \to 2$ operator norm of the projections matrices. Finally, we combine the obtained projection matrices from each channel to obtain a projection Π. Table 1 shows the values of the operator norms certified by our algorithm for each image channel and for the combined projection matrix. Notice the obtained subspaces have operator norm values significantly smaller than $\sqrt{n} = 55.42$. The reconstruction error in each case, when projected onto Π is at most 0.0345.

After training, on an input x we obtain a certified radius for ℓ_∞ perturbations in the DCT basis by obtaining a certified ℓ_2 radius of ε via randomized smoothing and then dividing the obtained value by $\|\Pi\|_{\infty\to 2}$. We compare with the approach of [SYL+19] for training a smoothed classifier without projections. Since the classifier of [SYL+19] does not involve projections, we translate the resulting ℓ_2 robustness guarantee into an ℓ_∞ guarantee by dividing with $\sqrt{n} = 55.42$ as is done in [SYL+19]. Figure 4 shows that across a range of training parameters, our proposed approach via robust projections leads to significantly higher certified accuracy to ℓ_∞ perturbations in the DCT basis.

Dataset	R	G	B	Π
CIFAR-10	17.45	17.51	17.39	30.22
CIFAR-100	17.22	17.33	17.37	29.97

Table 1: The table shows bounds on $\infty \to 2$ norm for projection matrices obtained by Algorithm 2 on CIFAR-10 and CIFAR-100 training sets.

5 Related Work

Several recent works have aimed to design algorithms that are robust to adversarial perturbations. This is a rapidly growing research area, and here we survey the works most relevant in the context of the current paper.

In practice, first order methods are a popular choice for adversarial training. Algorithms such as the *fast gradient sign method* (FGSM) [GSS14] and *projected gradient descent* (PGD) [MMS+17] fall into this category. These methods are appealing since they are generally applicable to many network architecture and only rely on black box access to gradient information. Recent works have aimed to improve upon the scalability and runtime efficiency of the above approaches [ZYJ+19, SNG+19, WRK20]. While these methods can be used to gather evidences regarding the robustness of a classifier to first order attacks, i.e., those based on gradient information, they do not exclude the possibility of a stronger attack that can significantly degrade their performance.

There have been recent works studying the problem of efficiently certifying the adversarial robustness of classifiers from both empirical and theoretical perspectives [CAH18, WCAJ11, WPW+18, GMDC+18, MGV18, SGM+18, WZC+18, ZWC+18, ADV19]. For the case of certifying robustness to ℓ_∞ perturbations, recent works have explored linear programming (LP) and semi-definite programming (SDP) based approaches to produce lower bounds on the certified robustness of neural networks. Such approaches have seen limited success beyond shallow (depth at most 4) networks [RSL18, WK18]. These works use convex programming (including SDPs) to directly provide an upper bound on the objective that corresponds to finding an adversarial perturbation for a given classifier f on input x. While our multiplicative weights method also approximately solves a convex program (specifically, an SDP), we use the convex program to translate certified ℓ_2 robustness guarantees to certified ℓ_∞ guarantees in a black box manner.

Another line of work has focused on methods for certifying robustness via propagating interval bounds [GDS+18]. While these methods have seen recent success in scaling up to large networks, they typically cannot be applied in a black box manner and require access to the structure of the underlying network.

Randomized smoothing proposed in the works of [LCZH18, LAG+19, CRK19, CG17] is a simple and effective way to certify robustness of neural networks to ℓ_2 perturbations. These methods work by creating a smoothed classifier by adding Gaussian noise to inputs. The improved Lipschitzness of the smoothed classifier can then be translated to bounds on certified ℓ_2 robustness. Other recent works have also proposed developing Lipschitz classifiers in order to get improved robustness [YRZ+20].

The recent work of [SYL+19], that we build upon, shows that by incorporating the smoothed classifier into the training process, one can get state-of-the art results for certified ℓ_2 robustness. The recent works of [LCWC19, DGS+18, YDH+20] extends the work of [SYL+19] by developing smoothing based methods for other ℓ_p norms and noise distribution other than the Gaussian distribution. The authors in [SSY+20] explore smoothing based methods for making a pretrained classifier certifiably robust.

While smoothing based methods have been successful for providing robustness to ℓ_2 perturbations, they have not demonstrated the same benefits for for the more challenging setting of robustness to ℓ_∞ perturbations. The recent works of [BDMZ20, YDH+20, KLGF20] have provided lower bounds demonstrating that smoothing based methods are unlikely to yield benefits for robustness to ℓ_∞ perturbations in the worst case. There have also been recent works exploiting low rank representations to achieve better adversarial robustness [YZKX19] empirically against first order attacks such as the PGD method [MMS+17]. The goal in our work however, is to use low rank representations to achieve *certified* accuracy guarantees.

Fast approximate SDP algorithms. Our certification algorithm is based on the multiplicative weights update (MWU) framework applied to the problem of Quadratic Programming. Klein and Lu [KL96] gave the first MWU-based algorithm for solving particular semidefinite programs (SDP). They used the framework of [PST91] for solving convex programs to give a faster algorithm for solving the maxcut SDP within a $1+\delta$ approximation using $\tilde{O}(n)$ eigenvalue/eigenvector computations, where the $\tilde{O}$ hides polylogarithmic factors in n and polynomial factors in $1/\delta$. But the analysis of Klein and Lu [KL96] is specialized for the Max-Cut problem, which is a special case of (11) where M is a graph Laplacian; it does not directly extend to our more general setting (they use diagonal dominance of a Laplacian to get a small bound on the width). Arora, Hazan and Kale [AHK05, Kal07] introduced a more general framework for solving semidefinite programs. In particular, they showed how when this framework is applied to the problem of Quadratic Programming, it gives a running time bound of $O(\frac{n^{1.5}}{\delta^{2.5}} \cdot \min\{\, m, n^{1.5}/(\delta\alpha^*)\,\})$, where the optimal solution value is $\text{SDP}_{val} = \alpha^* \|M\|_1$, and m is the number of non-zeros of matrix M. For SDPs of the form (12), the width parameter could be reasonably large and could result in $\tilde{\Omega}(n^{3/2})$ iterations (see Section 6.3 of [Kal07]). To the best of our knowledge this represents the prior best running time for approximately solving the Quadratic Programming SDP (9), and is most related to our work.

Arora et al. [AK07] also gave primal-dual based algorithms near-linear time algorithms for several combinatorial problems like max-cut and sparsest cut that use semidefinite programming relaxations. Iyengar et al. [IPS11, JY11, AZLO16] consider positive covering and packing SDPs like max-cut, and give fast approximate algorithms based on the multiplicative weights method. In particular, [JY11, AZLO16, JLL+20] and several other works give parallel algorithms, where the iteration count only has a mild polylogarithmic dependence on width for solving these positive SDPs. This uses the matrix multiplicative weights method that involves computing matrix exponentials. Quadratic Programming does not fall into this class of problems in general, unless $M \succeq 0$. A very recent paper of Lee and Padmanabhan [LP19] gives algorithms that work for problems like quadratic programming with diagonal constraints; however this gives an additive approximation to the objective which does not suffice for our purposes. Moreover, our algorithm is very simple and practical, and each iteration only uses a single computation of the largest eigenvalue. Hence, analyzing this algorithm is interesting in its own right. Finally, interior point methods, cutting plane methods and the Ellipsoid algorithm find solutions to the SDP that have much better dependence on the accuracy δ e.g., a polynomial in $\log(1/\delta)$ dependence, at the expense of significantly higher dependence on n [Ali95, LSW15].

6 Conclusion

In this paper, we have shown significant benefits in leveraging natural structure that exists in real-world data e.g., low-rank or sparse representations, for obtaining certified robustness guarantees under both ℓ_2 perturbations and ℓ_∞ perturbations in natural data representations. Our experiments involving imperceptibility in the DCT basis for images suggest that further study of ℓ_∞ robustness for other natural basis (apart from the co-ordinate basis) would be useful for different data domains like images, audio etc. We also gave faster algorithms for approximately solving semi-definite programs for quadratic programming (with provable guarantees that improve the state-of-the-art), to obtain certified ℓ_∞ robustness guarantees. Such problem-specific fast approximate algorithms for powerful algorithmic techniques like SDPs and other convex relaxations may lead to more scalable certification procedures with better guarantees. Finally it would be interesting to see if our ideas and techniques involving the $\infty \to 2$ operator norm can be adapted into the training phase, in order to achieve better certified ℓ_∞ robustness in any desired basis without compromising much on natural accuracy.

References

[ACCV19] Pranjal Awasthi, Vaggos Chatziafratis, Xue Chen, and Aravindan Vijayaraghavan. Adversarially robust low dimensional representations. *arXiv preprint arXiv:1911.13268*, 2019.

[ADV19] Pranjal Awasthi, Abhratanu Dutta, and Aravindan Vijayaraghavan. On robustness to adversarial examples and polynomial optimization. In *Advances in Neural Information Processing Systems*, pages 13737–13747, 2019.

[AHK05] Sanjeev Arora, Elad Hazan, and Satyen Kale. Fast algorithms for approximate semidefinite programming using the multiplicative weights update method. In *46th Annual IEEE Symposium on Foundations of Computer Science (FOCS'05)*, pages 339–348. IEEE, 2005.

[AK07] Sanjeev Arora and Satyen Kale. A combinatorial, primal-dual approach to semidefinite programs. In *Proceedings of the Thirty-Ninth Annual ACM Symposium on Theory of Computing*, STOC '07, page 227–236, New York, NY, USA, 2007. Association for Computing Machinery.

[Ali95] Farid Alizadeh. Interior point methods in semidefinite programming with applications to combinatorial optimization. *SIAM Journal on Optimization*, 5(1):13–51, 1995.

[AMMN06] Noga Alon, Konstantin Makarychev, Yury Makarychev, and Assaf Naor. Quadratic forms on graphs. *Inventiones mathematicae*, 163(3):499–522, 2006.

[AN04] Noga Alon and Assaf Naor. Approximating the cut-norm via grothendieck's inequality. In *Proceedings of the thirty-sixth annual ACM symposium on Theory of computing*, pages 72–80. ACM, 2004.

[ApS19] MOSEK ApS. *The MOSEK Fusion API for Python manual Version 9.1.13*, 2019.

[AZLO16] Zeyuan Allen-Zhu, Yin Tat Lee, and Lorenzo Orecchia. Using optimization to obtain a width-independent, parallel, simpler, and faster positive sdp solver. In *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '16, page 1824–1831, USA, 2016. Society for Industrial and Applied Mathematics.

[BCM$^+$13] Battista Biggio, Igino Corona, Davide Maiorca, Blaine Nelson, Nedim Šrndić, Pavel Laskov, Giorgio Giacinto, and Fabio Roli. Evasion attacks against machine learning at test time. In *Joint European conference on machine learning and knowledge discovery in databases*, pages 387–402. Springer, 2013.

[BDMZ20] Avrim Blum, Travis Dick, Naren Manoj, and Hongyang Zhang. Random smoothing might be unable to certify ℓ_∞ robustness for high-dimensional images. *arXiv preprint arXiv:2002.03517*, 2020.

[BGG$^+$18] Vijay Bhattiprolu, Mrinalkanti Ghosh, Venkatesan Guruswami, Euiwoong Lee, and Madhur Tulsiani. Inapproximability of matrix p to q norms. *arXiv preprint arXiv:1802.07425*, 2018.

[CAH18] Francesco Croce, Maksym Andriushchenko, and Matthias Hein. Provable robustness of relu networks via maximization of linear regions. *arXiv preprint arXiv:1810.07481*, 2018.

[CG17] Xiaoyu Cao and Neil Zhenqiang Gong. Mitigating evasion attacks to deep neural networks via region-based classification. In *Proceedings of the 33rd Annual Computer Security Applications Conference*, pages 278–287, 2017.

[CRK19] Jeremy M Cohen, Elan Rosenfeld, and J Zico Kolter. Certified adversarial robustness via randomized smoothing. *arXiv preprint arXiv:1902.02918*, 2019.

[CW04] Moses Charikar and Anthony Wirth. Maximizing quadratic programs: Extending grothendieck's inequality. In *Proceedings of the 45th Annual IEEE Symposium on Foundations of Computer Science*, FOCS '04, page 54–60, USA, 2004. IEEE Computer Society.

[CW18] Nicholas Carlini and David Wagner. Audio adversarial examples: Targeted attacks on speech-to-text. In *2018 IEEE Security and Privacy Workshops (SPW)*, pages 1–7. IEEE, 2018.

[DB16] Steven Diamond and Stephen Boyd. CVXPY: A Python-embedded modeling language for convex optimization. *Journal of Machine Learning Research*, 17(83):1–5, 2016.

[DGS$^+$18] Krishnamurthy Dvijotham, Sven Gowal, Robert Stanforth, Relja Arandjelovic, Brendan O'Donoghue, Jonathan Uesato, and Pushmeet Kohli. Training verified learners with learned verifiers. *arXiv preprint arXiv:1805.10265*, 2018.

[ERLD17] Javid Ebrahimi, Anyi Rao, Daniel Lowd, and Dejing Dou. Hotflip: White-box adversarial examples for text classification. *arXiv preprint arXiv:1712.06751*, 2017.

[GDS$^+$18] Sven Gowal, Krishnamurthy Dvijotham, Robert Stanforth, Rudy Bunel, Chongli Qin, Jonathan Uesato, Relja Arandjelovic, Timothy Mann, and Pushmeet Kohli. On the effectiveness of interval bound propagation for training verifiably robust models. *arXiv preprint arXiv:1810.12715*, 2018.

[GMDC$^+$18] Timon Gehr, Matthew Mirman, Dana Drachsler-Cohen, Petar Tsankov, Swarat Chaudhuri, and Martin Vechev. Ai2: Safety and robustness certification of neural networks with abstract interpretation. In *2018 IEEE Symposium on Security and Privacy (SP)*, pages 3–18. IEEE, 2018.

[GSS14] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

[HZRS16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[IPS11] G. Iyengar, D. J. Phillips, and C. Stein. Approximating semidefinite packing programs. *SIAM Journal on Optimization*, 21(1):231–268, 2011.

[JLL+20] Arun Jambulapati, Yin Tat Lee, Jerry Li, Swati Padmanabhan, and Kevin Tian. Positive semidefinite programming: mixed, parallel, and width-independent. In *Proccedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing, STOC 2020, Chicago, IL, USA, June 22-26, 2020*, pages 789–802. ACM, 2020.

[JY11] Rahul Jain and Penghui Yao. A parallel approximation algorithm for positive semidefinite programming. In *Proceedings of the 2011 IEEE 52nd Annual Symposium on Foundations of Computer Science*, FOCS '11, page 463–471, USA, 2011. IEEE Computer Society.

[Kal07] Satyen Kale. *Efficient algorithms using the multiplicative weights update method.* Princeton University, 2007.

[KH+09] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[KL96] Philip Klein and Hsueh-I Lu. Efficient approximation algorithms for semidefinite programs arising from max cut and coloring. In *Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing*, STOC '96, page 338–347, New York, NY, USA, 1996. Association for Computing Machinery.

[KLGF20] Aounon Kumar, Alexander Levine, Tom Goldstein, and Soheil Feizi. Curse of dimensionality on randomized smoothing for certifiable robustness. *arXiv preprint arXiv:2002.03239*, 2020.

[LAG+19] Mathias Lecuyer, Vaggelis Atlidakis, Roxana Geambasu, Daniel Hsu, and Suman Jana. Certified robustness to adversarial examples with differential privacy. In *2019 IEEE Symposium on Security and Privacy (SP)*, pages 656–672. IEEE, 2019.

[LCWC19] Bai Li, Changyou Chen, Wenlin Wang, and Lawrence Carin. Certified adversarial robustness with additive noise. In *Advances in Neural Information Processing Systems*, pages 9459–9469, 2019.

[LCZH18] Xuanqing Liu, Minhao Cheng, Huan Zhang, and Cho-Jui Hsieh. Towards robust neural networks via random self-ensemble. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 369–385, 2018.

[LP19] Yin Tat Lee and Swati Padmanabhan. An $\tilde{o}(m/\varepsilon^{3.5})$-cost algorithm for semidefinite programs with diagonal constraints. *CoRR*, abs/1903.01859, 2019.

[LSW15] Y. T. Lee, A. Sidford, and S. C. Wong. A faster cutting plane method and its implications for combinatorial and convex optimization. In *2015 IEEE 56th Annual Symposium on Foundations of Computer Science*, pages 1049–1065, 2015.

[MBPS09] Julien Mairal, Francis Bach, Jean Ponce, and Guillermo Sapiro. Online dictionary learning for sparse coding. In *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09, page 689–696, New York, NY, USA, 2009. Association for Computing Machinery.

[MGV18] Matthew Mirman, Timon Gehr, and Martin Vechev. Differentiable abstract interpretation for provably robust neural networks. In *International Conference on Machine Learning*, pages 3578–3586, 2018.

[MMS$^+$17] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017.

[Nes98] Yu Nesterov. Semidefinite relaxation and nonconvex quadratic optimization. *Optimization methods and software*, 9(1-3):141–160, 1998.

[PST91] S. A. Plotkin, D. B. Shmoys, and E. Tardos. Fast approximation algorithms for fractional packing and covering problems. In *[1991] Proceedings 32nd Annual Symposium of Foundations of Computer Science*, pages 495–504, 1991.

[PVG$^+$11] 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.

[RSL18] Aditi Raghunathan, Jacob Steinhardt, and Percy S Liang. Semidefinite relaxations for certifying robustness to adversarial examples. In *Advances in Neural Information Processing Systems*, pages 10877–10887, 2018.

[SGM$^+$18] Gagandeep Singh, Timon Gehr, Matthew Mirman, Markus Püschel, and Martin Vechev. Fast and effective robustness certification. In *Advances in Neural Information Processing Systems*, pages 10802–10813, 2018.

[SNG$^+$19] Ali Shafahi, Mahyar Najibi, Mohammad Amin Ghiasi, Zheng Xu, John Dickerson, Christoph Studer, Larry S Davis, Gavin Taylor, and Tom Goldstein. Adversarial training for free! In *Advances in Neural Information Processing Systems*, pages 3353–3364, 2019.

[SSY$^+$20] Hadi Salman, Mingjie Sun, Greg Yang, Ashish Kapoor, and J Zico Kolter. Black-box smoothing: A provable defense for pretrained classifiers. *arXiv preprint arXiv:2003.01908*, 2020.

[SYL$^+$19] Hadi Salman, Greg Yang, Jerry Li, Pengchuan Zhang, Huan Zhang, Ilya Razenshteyn, and Sebastien Bubeck. Provably robust deep learning via adversarially trained smoothed classifiers. *arXiv preprint arXiv:1906.04584*, 2019.

[SZS$^+$13] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

[WCAJ11] Shiqi Wang, Yizheng Chen, Ahmed Abdou, and Suman Jana. Mixtrain: scalable training of formally robust neural networks. corr abs/1811.02625 (2018). *arxiv. org/abs*, 1811.

[WK18] Eric Wong and Zico Kolter. Provable defenses against adversarial examples via the convex outer adversarial polytope. In *International Conference on Machine Learning*, pages 5283–5292, 2018.

[WPW$^+$18] Shiqi Wang, Kexin Pei, Justin Whitehouse, Junfeng Yang, and Suman Jana. Efficient formal safety analysis of neural networks. In *Advances in Neural Information Processing Systems*, pages 6367–6377, 2018.

[WRK20] Eric Wong, Leslie Rice, and J Zico Kolter. Fast is better than free: Revisiting adversarial training. *arXiv preprint arXiv:2001.03994*, 2020.

[WZC$^+$18] Tsui-Wei Weng, Huan Zhang, Hongge Chen, Zhao Song, Cho-Jui Hsieh, Duane Boning, Inderjit S Dhillon, and Luca Daniel. Towards fast computation of certified robustness for relu networks. *arXiv preprint arXiv:1804.09699*, 2018.

[YDH$^+$20] Greg Yang, Tony Duan, Edward Hu, Hadi Salman, Ilya Razenshteyn, and Jerry Li. Randomized smoothing of all shapes and sizes. *arXiv preprint arXiv:2002.08118*, 2020.

[YRZ$^+$20] Yao-Yuan Yang, Cyrus Rashtchian, Hongyang Zhang, Ruslan Salakhutdinov, and Kamalika Chaudhuri. Adversarial robustness through local lipschitzness. *arXiv preprint arXiv:2003.02460*, 2020.

[YZKX19] Yuzhe Yang, Guo Zhang, Dina Katabi, and Zhi Xu. Me-net: Towards effective adversarial robustness with matrix estimation. *arXiv preprint arXiv:1905.11971*, 2019.

[ZWC$^+$18] Huan Zhang, Tsui-Wei Weng, Pin-Yu Chen, Cho-Jui Hsieh, and Luca Daniel. Efficient neural network robustness certification with general activation functions. In *Advances in neural information processing systems*, pages 4939–4948, 2018.

[ZYJ$^+$19] Hongyang Zhang, Yaodong Yu, Jiantao Jiao, Eric P Xing, Laurent El Ghaoui, and Michael I Jordan. Theoretically principled trade-off between robustness and accuracy. *arXiv preprint arXiv:1901.08573*, 2019.

A Additional experiments for certified ℓ_2 robustness

We first discuss the setting of the hyperparameters in our experiments. The experiments in Figure 1, Figure 2, and Figure 4 were obtained by training a ResNet-32 architecture [HZRS16]. In each case we optimize the objective in (7) with a starting learning rate of 0.1 and decaying the learning rate by a factor of 0.1 every 50 epochs. Each model was trained for 150 epochs. We follow the methodology of [SYL+19] and approximate the soft classifier in (5) by drawing 4 noise vectors from the Gaussian distribution $N(0, \sigma^2 I)$ and optimizing the inner maximization in (7) via 10 steps of projected gradient descent. When training the method of [SYL+19] we choose the value of σ to be 0.12 for $\varepsilon = 0.25, 0.5$. For $\varepsilon = 0.75$ and $\varepsilon = 1.0$, we choose σ to be 0.25 and 0.5 respectively. When training our procedure in Algorithm 1 we use a higher noise of $\sigma \cdot \lambda\sqrt{d/r}$, where σ is the corresponding noise used for the method of [SYL+19], $d = 1024$ and $r = 200$. We vary the parameter λ as discussed in Section 2. Empirically, we find that $\lambda = 0.5$ gives the best results.

We first demonstrate that real data such as images have a natural low rank structure. As an illustration, Figure 5 shows the relative reconstruction error for the CIFAR-10 and CIFAR-100 [KH+09] datasets, when each of the three channels is projected onto subspaces of varying dimensions, computed via principal component analysis (PCA). As can be seen, even when projected onto 200 dimensions, the reconstruction error remains small.

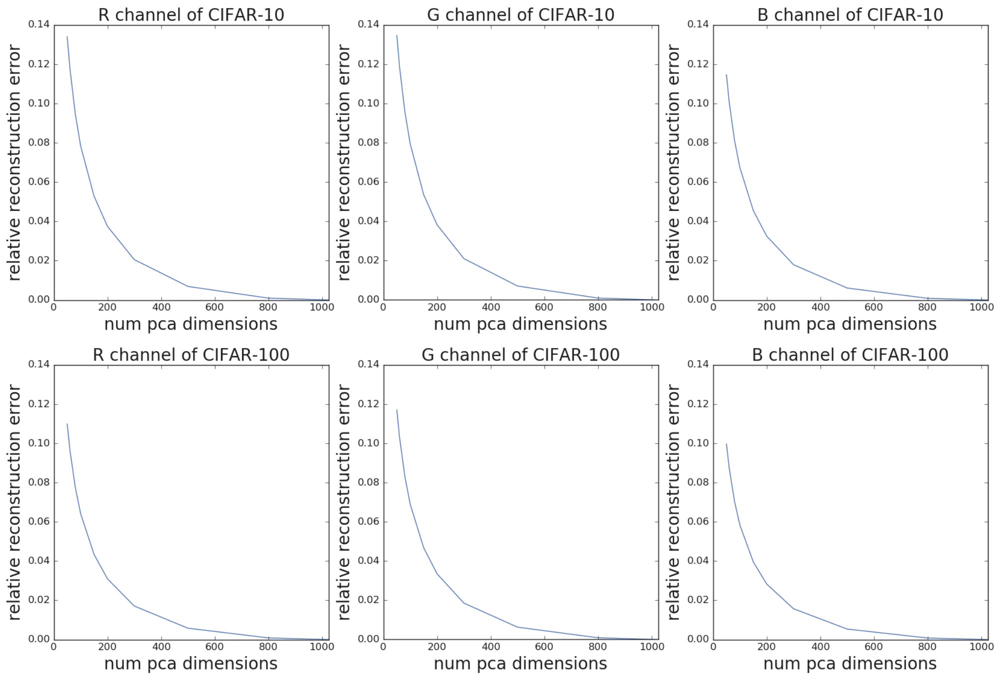

Figure 5

Next we provide in Figure 6 a more fine grained comparison between our procedure in Algorithm 1 and the method of [SYL+19] by separately comparing the performance of the two methods for different values of ε on the CIFAR-10 and CIFAR-100 datasets.

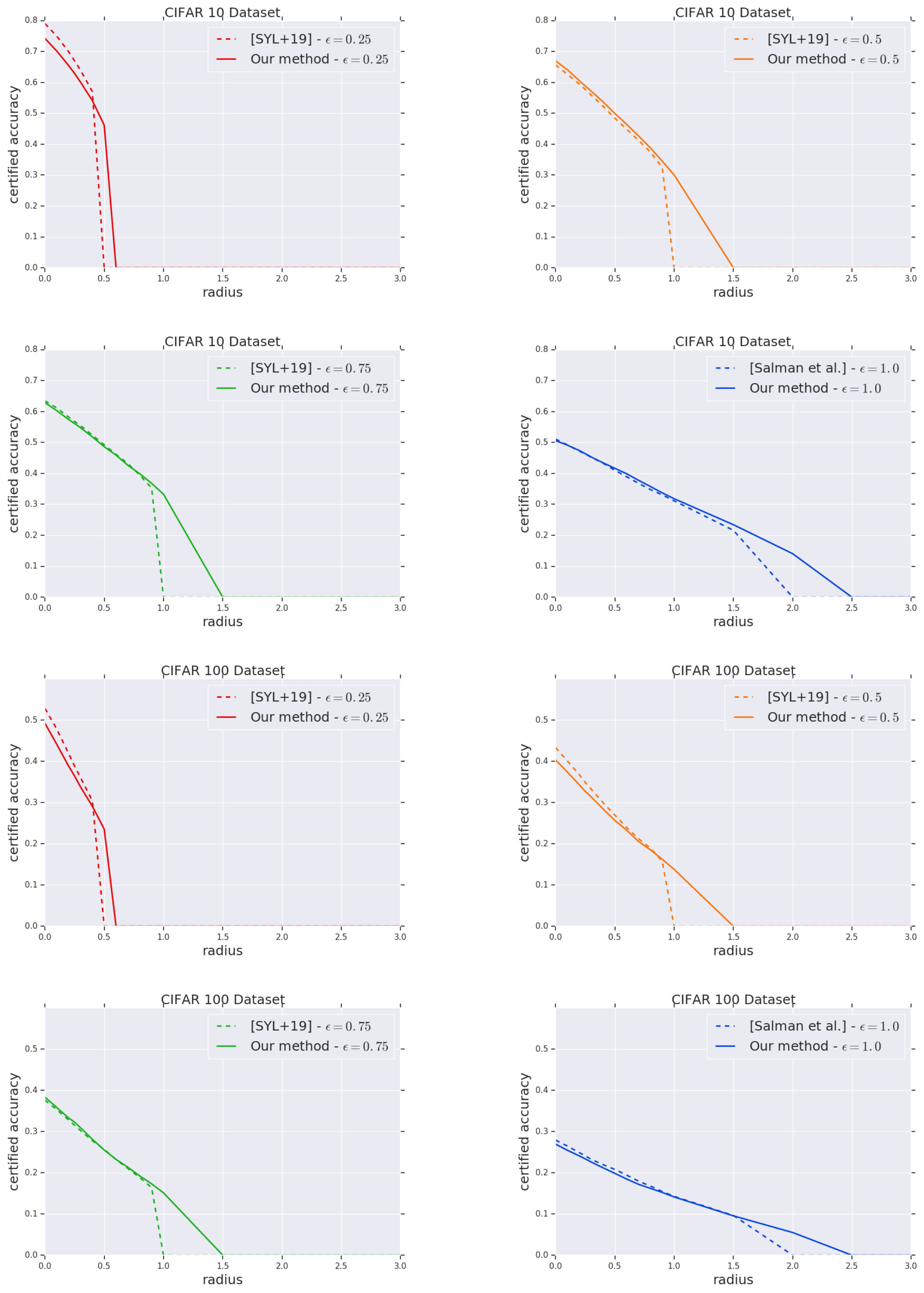

Figure 6: A comparison of certified radius guarantees obtained via Algorithm 1 as compared to the approach of [SYL+19]. The x-axis is the radius, and the y-axis represents the certified accuracy.

B Simple Theoretical Propositions and Proofs

The following proposition shows the equivalence between the smoothed classifiers in (4) and (5).

Proposition B.1. *Given a base classifier $f : \mathbb{R}^n \to \mathcal{Y}$ and a projection matrix Π, on any input x, the smoothed classifier $g_\Pi(x)$ as defined in (4) is equivalent to the classifier given by*

$$\tilde{g}_\Pi(x) = \arg\max_{y \in \mathcal{Y}} \mathbb{P}(f(\Pi(x+\delta)) = y).$$

Here δ is a standard Gaussian of variance σ^2 in every direction.

Proof. Let Π be a projection matrix with the corresponding subspace denoted by $\mathcal{S}$. Let δ be a random variable distributed as $N(0, \sigma^2 I)$ and δ_Π be a standard normal random variable with variance σ^2 within $\mathcal{S}$ and variance 0 outside. From the property of Gaussian distributions we have that projections of a spherical Gaussian random variable with variance σ^2 in each direction are themselves Gaussian random variables with variance σ^2 in each direction within the subspace. Hence, for a fixed input x, the random variables $\Pi x + \delta_\Pi$ and $\Pi x + \Pi\delta$ are identically distributed. From this we conclude that the classifier

$$g_\Pi(x) = \arg\max_{y \in \mathcal{Y}} \mathbb{P}(f(\Pi x + \delta_\Pi) = y).$$

is identical to the classifier $\tilde{g}_\Pi(x)$. □

The following proposition shows how to obtain an ℓ_∞ robustness guarantee from an ℓ_2 robustness guarantee.

Proposition B.2. *Consider any classifier $f : \mathbb{R}^n \to \mathcal{Y}$, and let the classifier $g(x) := f(\Pi x)$, where $\Pi \in \mathbb{R}^{n \times n}$ is any projection matrix. Then if we denote by $r_2(x), r_\infty(x)$ the radius of robustness measured in ℓ_2 and ℓ_∞ norm respectively of g around a point $x \in \mathbb{R}^n$. Then we have*

$$r_\infty(x) \geq \frac{r_2(x)}{\|\Pi\|_{\infty\to 2}}, \text{ where } \|\Pi\|_{\infty\to 2} = \max_{x : \|x\|_\infty \leq 1} \|\Pi x\|_2.$$

Hence for any $\varepsilon \geq 0$, we have $acc^{(\ell_\infty)}_{\varepsilon'}(g) \geq acc^{(\ell_2)}_{\varepsilon}(g)$, for all $0 \leq \varepsilon' \leq \varepsilon / \|P\|_{\infty\to 2}$.

Proof. Consider any perturbation $x' = x + z$ where $\|z\|_\infty \leq \varepsilon'$ such that the predictions given by $g(x') = f(\Pi x')$ and $g(x) = f(\Pi x)$ differs. Consider another perturbation z' such that $z' = \Pi z$. Notice that $g(x + z') = f(\Pi(x + \Pi z)) = g(x')$, since $\Pi^2 = \Pi$ for a projection matrix. Hence, the predictions of the network differ on x and $x + z'$. Moreover $\|z'\|_2 \leq \|\Pi\|_{\infty\to 2}\varepsilon'$. Hence g is not robust at x up to an ℓ_2 radius of $\varepsilon = \varepsilon'\|\Pi\|_{\infty\to 2}$, as required. □

The following simple proposition shows how to obtain certified robustness guarantees in the representation $\mathcal{U}$ e.g., DCT basis by performing appropriate training in another representation $\mathcal{X}$ e.g., the co-ordinate or pixel basis. Let $O \in \mathbb{R}^{n \times n}$ represent an orthogonal matrix that represents the DCT transformation i.e., for an input $x \in \mathcal{X} \subseteq \mathbb{R}^n$ let $\psi(x) = Ox$ represents its DCT representation (hence $\mathcal{U} = \psi(\mathcal{X})$).

Proposition B.3. *Suppose $\mathcal{X}, \mathcal{U} = \psi(\mathcal{X})$ be defined as above. For any classifier $g : \mathcal{U} \to \mathcal{Y}$, consider the classifier $f : \mathcal{X} \to \mathcal{Y}$ obtained as $f = g(Ox)$ where O is an orthogonal matrix. For a point $x \in \mathcal{X}$ if we denote by $r_f^{(\ell_2)}(x)$ the radius of robustness of f at x measured in ℓ_2 norm, then we have that $r_g^{(\ell_2)}(Ox) = r_f^{(\ell_2)}(x)$. Moreover suppose the classifier $g(u) = g(\Pi u)$ $\forall u \in \mathcal{U}$, for some projection matrix Π, then the robust accuracy of g to ℓ_∞ perturbations in the representation $\mathcal{U}$ satisfies $acc_{\varepsilon'}^{(\ell_\infty)}(g) \geq acc_{\varepsilon}^{(\ell_2)}(f)$ for any $\varepsilon > 0$ and $\varepsilon' \leq \varepsilon / \|\Pi\|_{\infty \to 2}$.*

Proof. We will prove by contradiction. Let $u, u' \in \psi(\mathcal{X})$ and let $u' = u + \delta_u$ be an adversarial perturbation of u with $\|\delta_u\|_2 = r$. Let $x = O^{-1}u$, and let $x' = O^{-1}u'$; note that x, x' exist since $u, u' \in \psi(\mathcal{X})$. Moreover $\|x - x'\|_2 = \|u - u'\|_2$ since O is a rotation (orthogonal matrix). Hence x' is an adversarial example for f at x, at a ℓ_2 distance of δ_u. This shows that $r_g^{(\ell_2)}(Ox) \geq r_f^{(\ell_2)}(x)$. A similar argument also shows that $r_g^{(\ell_2)}(Ox) \leq r_f^{(\ell_2)}(x)$. The last part of the proposition follows by applying Proposition B.2. □

The following simple fact relates the $\infty \to 2$ operator norm and the ℓ_1 sparsity of a projection matrix. This justifies the use of ℓ_1 sparsity as an approximate relaxation or proxy for $\|\Pi\|_{\infty \to 2}$.

Fact B.4. *For any (orthogonal) projection matrix $\Pi \in \mathbb{R}^{n \times n}$ of rank r, we have*

$$\frac{\|\Pi\|_1}{r} \leq \|\Pi\|_{\infty \to 2}^2 = \|\Pi\|_{\infty \to 1} \leq \|\Pi\|_1,$$

where $\|\Pi\|_1$ refers to the entry-wise ℓ_1 norm of Π.

Proof. First we show the upper bound

$$\|\Pi\|_{\infty \to 1} = \max_{y,z \in \mathbb{R}^n : \|y\|_\infty \leq 1, \|z\|_\infty \leq 1} \langle \Pi, yz^\top \rangle \leq \sum_{i,j} |\Pi(i,j)| = \|\Pi\|_1.$$

We now show the lower bound. Let $\Pi = \sum_{i=1}^r v_i v_i^\top$, where $\{ v_i : i \in [r] \}$ represents an orthonormal basis for the subspace given by Π. We have from the monotonicity of $\infty \to 1$ operator norm shown in [ACCV19] that

$$\begin{aligned}\|\Pi\|_{\infty \to 1} &= \Big\| \sum_{i=1}^r v_i v_i^\top \Big\|_{\infty \to 1} \geq \max_{i \in [r]} \|v_i v_i^\top\|_{\infty \to 1} = \max_{i \in [r]} \|v_i v_i^\top\|_1 \\ &\geq \frac{1}{r} \sum_{i \in [r]} \|v_i v_i^\top\|_1 \geq \frac{1}{r} \Big\| \sum_{i \in [r]} v_i v_i^\top \Big\|_1 = \frac{\|\Pi\|_1}{r},\end{aligned}$$

where the last inequality uses the triangle inequality for matrix operator norms.

□

C Translating certified adversarial robustness from ℓ_2 to ℓ_∞ perturbations

C.1 Certifying the $\infty \to 2$ operator norm

We now describe our efficient algorithmic procedure that gives a certifies an upper bound on the $\infty \to 2$ norm of the given matrix. Moreover, as we will see in Theorem C.2, our algorithmic procedure comes with provable guarantees: it is guaranteed to output a value that is only a small constant factor off from the global optimum i.e., true value of the $\infty \to 2$ norm.

For any matrix A, we first note that $\|A\|_{\infty\to 2}^2 = \|A^\top A\|_{\infty\to 1}$. Moreover, by the variational characterization of operator norms, we have for $M = AA^\top \succeq 0$

$$\|A\|_{\infty\to 2}^2 = \max_{x:\|x\|_\infty \le 1} \max_{y:\|y\|_\infty \le 1} x^\top M y = \max_{x:\|x\|_\infty \le 1} x^\top M x. \tag{11}$$

As stated in Section 3, our algorithm will apply to the more general problem (9) where $M_{ii} \ge 0$ for all $i \in [n]$.

$$\text{Given a symmetric matrix } M \text{ with } \ \forall i \in [n]: \ M_{ii} \ge 0, \qquad \max_{x:\|x\|_\infty \le 1} x^\top M x.$$

Note that this is certainly satisfied by all $M \succeq 0$. We consider the following standard SDP relaxation (12) for the problem, where $M \succeq 0$. The primal variables are represented by the positive semi-definite (PSD) matrix $X \in \mathbb{R}^{n\times n}$. The SDP dual of this relaxation, given in (14), has variables $y_1, \dots, y_n \ge 0$ corresponding to the n constraints (13). In what follows, for matrices M, X, $\langle M, X\rangle := \mathrm{tr}(M^\top X)$ represents the trace inner product.

Primal SDP:

$$\max_X \ \langle M, X\rangle \tag{12}$$
$$\text{s.t.} \quad X_{ii} \le 1, \quad \forall i \in [n] \tag{13}$$
$$X \succeq 0.$$

Dual SDP:

$$\min_y \ \sum_{i\in[n]} y_i \tag{14}$$
$$\text{s.t.} \quad diag(y) \succeq M \tag{15}$$
$$y \ge 0.$$

Let us denote by SDP_{val} the value of the optimal solution to the primal SDP (12). Recall that our algorithm goal is to output a value that is guaranteed to be a valid upper bound for $\|A\|_{\infty\to 2}^2 = \|M\|_{\infty\to 1}$.[3] The above primal SDP (12) is a valid relaxation, and it is also tight up to a factor of $\pi/2$ i.e., $(2/\pi)\text{SDP}_{val} \le \|M\|_{\infty\to 1} \le \text{SDP}_{val}$. Moreover by weak duality, any feasible solution to the dual SDP (14) gives a valid upper bound for the primal SDP value, and hence $\|M\|_{\infty\to 1}$. However, the above SDP is computationally intensive to solve using off-the-shelf SDP solvers (even for CIFAR-10 images, X is 1024×1024) . Instead we design an algorithm based on the *multiplicative weight update* (MWU) framework for solving SDPs [KL96, AHK05].

Our algorithm is described in Algorithm 2 of Section 3. Our algorithm differs from the standard MWU approach (and analysis) [Kal07] treats the constraints (13) and the objective (12) similarly. Firstly, by treating the objective differently from the constraints, we ensure that the width of the convex program is smaller; in particular, the "width" parameter does not depend on the value of the objective function anymore. This allows us to get a better convergence guarantee. Secondly, we always maintain a dual feasible solution that gives a valid upper bound on the objective value (this also allows us to do an early stop if appropriate). We remark that for *every* choice of α (and the update rule), $\text{val}_{\text{curr}} = n\lambda$ gives a valid upper bound (due to dual feasibility). This is encapsulated in the following simple proposition where SDP_{val} refers to the optimal solution value of the (12).

Proposition C.1. *For any $\alpha \in \mathbb{R}^n_{\ge 0}$ with $\sum_{i=1}^n \alpha(i) = n$, if λ is the maximum eigenvalue*

$$\lambda = \lambda_{\max}\Big(\big(diag(\alpha)^{-1/2} M \, diag(\alpha)^{-1/2}\big)\Big), \text{ then we have } SDP_{val} \le n\lambda,$$

and $y = \lambda\alpha$ is feasible for the dual SDP (14) and attains an objective value of $n\lambda$.

[3] Hence a fast algorithm that potentially finds a local optimum for the problem will not suffice for our purposes; we need an upper bound on the global optimum of (11).

Proof. Consider $y = \lambda\alpha$. Firstly, the dual objective value at y is $\sum_{i=1}^n y_i = \lambda \sum_i \alpha(i) = n\lambda$, as required. Moreover $y \geq 0$, since the $\alpha \geq 0$. Finally, (15) is satisfied since by definition of λ,

$$\mathrm{diag}(\alpha)^{-1/2} M \mathrm{diag}(\alpha)^{-1/2} \preceq \lambda \cdot I \quad \Longrightarrow \quad M \preceq \lambda \cdot \mathrm{diag}(\alpha) = \mathrm{diag}(y),$$

by pre-multiplying and post-multiplying by $\mathrm{diag}(\alpha)^{1/2}$. Finally, from weak duality $\mathrm{SDP}_{val} \leq n\lambda$. $\square$

Analysis of the algorithm We show the following guarantee for our algorithm.

Theorem C.2. *For any $\delta > 0$, any symmetric matrix M with $M_{ii} \geq 0\ \forall i \in [n]$, Algorithm 2 on input M, with parameters δ and $\rho = O(n/\delta)$ after $T = O(n \log n/\delta^3)$ iterations finds a solution $\widehat{X} \in \mathbb{R}^{n\times n}$ and $\widehat{y} \in \mathbb{R}^n$ such that*

(a) $\widehat{y}$ is feasible for the dual SDP (14) such that $\langle M, \widehat{X}\rangle = \sum_{i=1}^n y_i$.
(b) $\widehat{X} \succeq 0$, $\widehat{X}_{ii} \leq 1 + \delta$ for all $i \in [n]$, and $\langle M, \widehat{X}\rangle \geq SDP_{val}$, the primal SDP value.

The above Theorem C.2 and Proposition C.1 together imply Theorem 3.1. We remark that the $\tilde{O}(n \log n) \cdot T_{eig}$ running time guarantee almost matches the guarantees for Klein and Lu for Max-Cut SDP [KL96] (up to a $O(1/\delta)$ factor), but our guarantees apply to the SDP for the more general Quadratic Programming problem. Note that the maximum eigenvalue can be computed within δ accuracy in $O(m/\delta)$ time where m is the number of non-zeros of M, (see e.g., [KL96] for a proof and use in MWU framework). To the best of our knowledge, the best known algorithm prior to our work for solving the SDP to this general Quadratic Programming problem (or even problem (11)) is by Arora, Hazan and Kale [AHK05, Kal07] who give a $O(\frac{n^{1.5}}{\delta^{2.5}} \cdot \min\{m, n^{1.5}/(\delta\alpha^*)\})$, where the optimal solution value is $\mathrm{SDP}_{val} = \alpha^* \|M\|_1$. Compared to our algorithm's running time of $\tilde{O}(nm \log n)$, even when $\alpha^* = \Omega(1)$ the previous best requires a running time of $\tilde{O}(n^{1.5} \min\{m, n^{1.5})\})$; but it can even be the case that $\alpha^* = O(1/n)$ [CW04]. Finally recall that upper bound given by the SDP is only off by a factor of $\pi/2$ for PSD matrices, that is the best possible assuming $P \neq NP$ (for general matrices the approximation factor could be $O(\log n)$) [Nes98, CW04, AMMN06, BGG$^+$18].

The analysis closely mirrors the analysis of the Multiplicative Weights algorithm for solving SDPs due to Klein and Lu [KL96], and Arora, Hazan and Kale [AHK05, Kal07]. The main parameter that affects the running time of the multiplicative weights update method is the width parameter ρ' of the SDP constraints. The analysis of Klein and Lu [KL96] is specialized for the Max-Cut problem (which is a special case of (11) where M is a graph Laplacian), where they achieve a bound of $O(n \log n/\delta^2)$ eigenvalue computations. However their analysis does not directly extend to our more general setting (they crucially use the fact that the optimum solution value is at least $\Omega(\mathrm{tr}(M))$ to get a small bound on the width). In the more general framework of [AHK05], the optimization problem is first converted into a feasibility problem (hence the objective also becomes a constraint). For SDPs of the form (12), the width parameter could be reasonably large and could result in $\Omega(n^{3/2})$ iterations (see Section 6.3 of [Kal07]). In our analysis, we ensure that the width is small by treating the objective separately from the constraints. A crucial step of our algorithm is introducing an amount δ to each of the weights which solving the eigenvalue maximization in each step. This correction term of δ is important in getting an upper bound on the violation of each constraint, and hence the width of the program. Finally, we present a slightly different analysis using the SDP dual, that has the additional benefit of providing a certificate of optimality.

Proof of Theorem C.2. Let at step t of the iteration, $\alpha^{(t)}$ be the weight vector, $X^{(t)}$ be the candidate SDP solution and $y^{(t)}$ be the candidate dual solution maintained by the Algorithm 2 in step 6. Also $\tilde{\alpha}^{(t)} := (1-\delta)\alpha^{(t)} + \delta\mathbf{1}$, where $\mathbf{1} = (1, 1, \ldots, 1)$. It is easy to see that

$$\forall t \le T, \quad X^{(t)} = \frac{1}{t}\sum_{\ell=1}^{t} v^{(\ell)}(v^{(\ell)})^\top, \quad \text{and} \qquad y^{(t)} = \frac{1}{t}\sum_{\ell=1}^{t} \lambda^{(\ell)}\tilde{\alpha}^{(\ell)}.$$

We first establish the following lemma, which immediately implies part (a) of the theorem statement.

Lemma C.3. *For all $t \le T$, $y^{(t)}$ is feasible for the dual SDP (14), and $\langle M, X^{(t)}\rangle = \sum_i y_i^{(t)}$.*

Proof. We first check feasibility of the dual solution. First note that in every iteration $t \in [T]$, $\tilde{\alpha}^{(t)} \ge 0$ and $\lambda^{(t)} \ge 0$. The latter is because $\mathrm{tr}(\mathrm{diag}(\tilde{\alpha}^{(t)})^{-1/2} M \mathrm{diag}(\tilde{\alpha}^{(t)})^{-1/2}) \ge 0$ as all the diagonal entries of M are non-negative, and hence $\lambda_{\max}(\mathrm{diag}(\tilde{\alpha}^{(t)})^{-1/2} M \mathrm{diag}(\tilde{\alpha}^{(t)})^{-1/2}) \ge 0$. Hence $y^{(t)} \ge 0$. Moreover, by definition of $\lambda^{(t)}$

$$M \preceq \lambda^{(\ell)} \cdot \mathrm{diag}(\tilde{\alpha}^{(\ell)}) \;\; \forall \ell \le T, \;\; (\text{ since } \mathrm{diag}(\tilde{\alpha}^{(t)})^{-1/2} M \mathrm{diag}(\tilde{\alpha}^{(t)})^{-1/2} \preceq \lambda^{(t)} I),$$
$$M \preceq \frac{1}{t}\sum_{\ell=1}^{t} \lambda^{(\ell)} \cdot \mathrm{diag}(\tilde{\alpha}^{(\ell)}) = \frac{1}{t}\sum_{\ell=1}^{t} \mathrm{diag}(y^{(\ell)}).$$

Finally, we establish $\langle M, X^{(t)}\rangle = \sum_i y_i^{(t)}$. Note that by definition $v^{(\ell)} = \sqrt{n}\mathrm{diag}(\tilde{\alpha}^{(\ell)})^{-1/2} u^{(\ell)}$, where $u^{(\ell)}$ is the top eigenvector of $(\mathrm{diag}(\tilde{\alpha})^{-1/2} M \mathrm{diag}(\tilde{\alpha})^{-1/2})$, and $\lambda^{(\ell)}$ is its eigenvalue. Hence

$$\begin{aligned}\langle M, X^{(t)}\rangle &= \frac{1}{t}\sum_{\ell=1}^{t} \langle M, v^{(\ell)}(v^{(\ell)})^\top\rangle = \frac{1}{t}\sum_{\ell=1}^{t} (v^{(\ell)})^\top M v^{(\ell)} \\ &= n\frac{1}{t}\sum_{\ell=1}^{t}\left((u^{(\ell)})^\top \mathrm{diag}(\tilde{\alpha}^{(\ell)})^{-1/2} M \mathrm{diag}(\tilde{\alpha}^{(\ell)})^{-1/2} u^{(\ell)}\right) = \frac{1}{t}\sum_{\ell=1}^{t} \lambda^{(\ell)} \cdot n\end{aligned}$$

On the other hand, $\displaystyle\sum_{i=1}^{n} y_i^{(t)} = \sum_{i=1}^{n}\frac{1}{t}\sum_{\ell=1}^{t} \lambda^{(\ell)}\tilde{\alpha}^{(\ell)}(i) = \frac{1}{t}\sum_{\ell=1}^{t}\lambda^{(\ell)} \cdot \sum_{i=1}^{n}\tilde{\alpha}^{(\ell)}(i) = \frac{1}{t}\sum_{\ell=1}^{t}\lambda^{(\ell)} \cdot n.$

This proves the lemma. □

We now complete the proof of Theorem C.2.

Proof. Recall that $\widehat{X} = X^{(T)}, \widehat{y} = y^{(T)}$. Consider the SDP solution $X' = \frac{1}{1+\delta}\widehat{X}$. From Lemma C.3, we see that $y^{(T)}$ is feasible and

$$\langle M, X'\rangle = \frac{1}{1+\delta}\langle M, \widehat{X}\rangle = \frac{1}{1+\delta}\sum_{i=1}^{n}\widehat{y}(i) \ge \frac{1}{1+\delta} \cdot \mathrm{SDP}_{val},$$

where the last inequality follows from weak duality of the SDP. Hence it suffices to show that X' is feasible i.e., $\widehat{X}_{ii} \le 1+\delta$ for all $i \in [n]$.

We prove $\max_{i\in[n]} \widehat{X}_{ii} \le 1+\delta$ by following the same multiplicative weight analysis in [Kal07] (see Section 2, Theorem 5), but only restricted to the n constraints of the form $X_{ii} \le 1$.

There is one expert for each $i \in [n]$ corresponding to the constraint $X_{ii} \le 1$. At each iteration t, we consider a probability distribution $p^{(t)} = \frac{1}{n}\alpha^{(t)}$. The loss of expert i at time t is given by $m^{(t)}(i) = \frac{1}{\rho}(1 - (v_i^{(t)})^2)$ for $\rho = n/\delta$. We note that in each iteration ℓ,

$$\sum_i (\alpha_i^{(t)} + \delta)(v_i^{(t)})^2 \le n \implies \forall i \in [n], \quad (v_i^{(t)})^2 \le \frac{n}{\delta} \le \rho$$

$$\text{Hence } \frac{1}{\rho} \ge m_i^{(t)} = \frac{1}{\rho}\Big(1 - (v_i^{(t)})^2\Big) \ge -\frac{(n/\delta - 1)}{\rho} \ge -1.$$

From the guarantees of the multiplicative weight update method (see Section 2, Theorem 2 of [Kal07]), we have for each $i \in [n]$

$$\begin{aligned}
\sum_{t=1}^{T}\sum_{i=1}^{n} m_i^{(t)} p_i^{(t)} &\le \sum_{t=1}^{T} m_i^{(t)} + \delta \sum_{t=1}^{T} |m_i^{(t)}| + \frac{\ln n}{\delta} \\
&= (1-\delta)\sum_{t=1}^{T} m_i^{(t)} + 2\delta \sum_{\substack{t \in [T] \\ m_i^{(t)} > 0}} |m_i^{(t)}| + \frac{\ln n}{\delta} \\
&\le \frac{(1+\delta)}{\rho}\sum_{t=1}^{T}(1 - (v_i^{(t)})^2) + \frac{2\delta}{\rho} + \frac{\ln n}{\delta} \\
\frac{\rho}{T}\sum_{t=1}^{T}\sum_{i=1}^{n} m_i^{(t)} p_i^{(t)} &\le (1-\delta)\cdot(1 - \widehat{X}_{ii}) + 2\delta + \frac{\rho \ln n}{T\delta}. \qquad (16)
\end{aligned}$$

On the other hand, using the fact that $\sum_i \alpha_i^{(t)} = n$, and $\sum_i \tilde{\alpha}_i^{(t)}(v_i^{(t)})^2 = n$ we have

$$\begin{aligned}
\frac{\rho}{T}\sum_{t=1}^{T}\sum_{i=1}^{n} m_i^{(t)} p_i^{(t)} &= \frac{1}{T}\sum_{t=1}^{T}\frac{1}{n}\sum_{i=1}^{n}(1 - (v_i^{(t)})^2)\alpha_i^{(t)} \\
&= \frac{1}{T}\sum_{t=1}^{T}\sum_{i=1}^{n}\alpha_i^{(t)} - \frac{1}{T}\sum_{t=1}^{T}\frac{1}{n}\sum_{i=1}^{n}\alpha_i^{(t)}(v_i^{(t)})^2 = -\frac{1}{T}\sum_{t=1}^{T}\frac{1}{n}\sum_{i=1}^{n}\Big(\frac{\tilde{\alpha}_i{}^{(t)}}{1-\delta} + \frac{\delta}{1-\delta}\Big)(v_i^{(t)})^2 + 1 \\
&= -\frac{1}{T}\sum_{t=1}^{T}\frac{1}{(1-\delta)n}\sum_{i=1}^{n}\tilde{\alpha}_i{}^{(t)}(v_i^{(t)})^2 + \frac{1}{T}\sum_{t=1}^{T}\frac{1}{(1-\delta)n}\sum_{i=1}^{n}\delta(v_i^{(t)})^2) + 1 \\
&= -\frac{1}{1-\delta} + 1 + \frac{1}{T}\sum_{t=1}^{T}\frac{1}{(1-\delta)n}\sum_{i=1}^{n}\delta(v_i^{(t)})^2 \\
&\ge -\frac{\delta}{1-\delta} \qquad (17)
\end{aligned}$$

Combining (16) and (17), we get that for $\delta \in (0, 1/2)$, and $T = \rho \log n/((1-\delta)\delta^2) = O(n \log n/\delta^3)$

$$\forall i \in [n], \quad (1 - \widehat{X}_{ii}) \ge -\frac{\delta}{(1-\delta)^2} - \frac{2\delta}{1-\delta} - \frac{\rho \ln n}{T\delta(1-\delta)}$$

$$\widehat{X}_{ii} \le 1 + \delta(1 + 4\delta) + 2\delta(1 + 2\delta) + \delta \le 1 + 8\delta.$$

This completes the proof of part (b). It is also straightforward to see that in above analysis, a $(1+\delta)$ approximate eigenvalue method can also be used to get a similar guarantee with an extra $(1+\delta)$ factor loss in the objective value. □

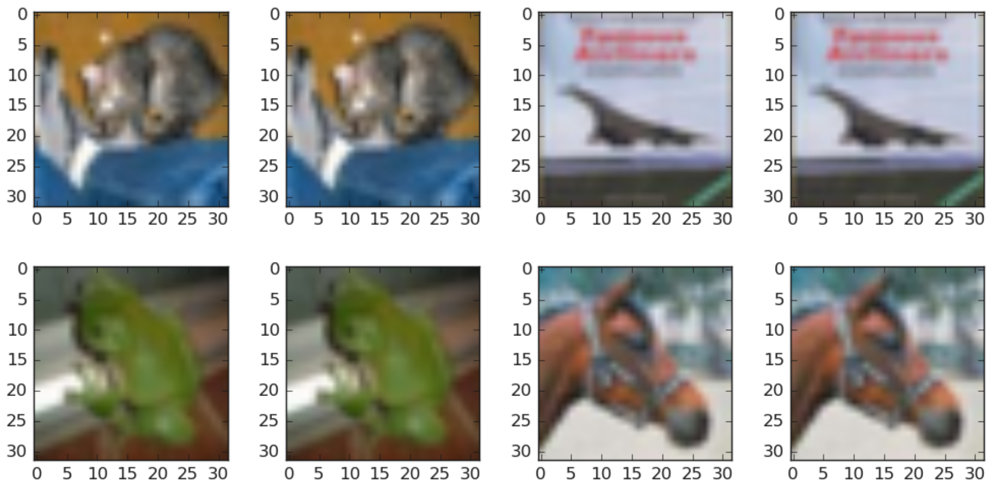

Figure 7: The images on the left correspond to the original images and the images on the right correspond to imperceptible adversarial examples within an ℓ_∞ radius of $\varepsilon \le 0.09$.

We remark that the larger dependence of $\rho = O(n/\delta)$ is needed to ensure that in each iteration $(v_i^{(t)})^2/\rho \le 1$. If this condition is satisfied for $\rho \ll n/\delta$, then the iteration bound is $O(\rho \log n)/\delta^2$. This also justifies the use of a more aggressive i.e., smaller choice of ρ in practice.

D Imperceptibility in the DCT basis and Training Certified ℓ_∞ Robust Networks

Adversarial Examples for CIFAR-10 images. We take a ResNet-32 network that has been trained on the CIFAR-10 datasets via the PGD based method of [MMS+17] for robustness to ℓ_∞ perturbations. We then generate imperceptible adversarial examples for the test images via projected gradient ascent and an ℓ_∞ perturbation radius of $\varepsilon = 0.09$. See Figure 7 for a few of the original images and the corresponding adversarial perturbations.

E Experimental Evaluation of MWU-based SDP Certification Algorithm 2

In this section we empirically evaluate the effectiveness of our multiplicative weights based procedure from Algorithm 2 for solving the general Quadratic Programming (QP) problem as defined in (9). Recall that this corresponds to the following optimization problem:

$$\text{Given a symmetric matrix } M \text{ with } \forall i \in [n]: \; M_{ii} \ge 0, \qquad \max_{x : \|x\|_\infty \le 1} x^\top M x. \tag{18}$$

Recall that in our Algorithm 2 every iterations involves a single maximum eigenvalue computation, for which we use an off the shelf subroutine from Python's scipy package. In our implementation of Algorithm 2 we add an early stopping condition if the dual value does not improve noticeably in successive rounds. Recall that Proposition C.1 proves that this also gives a valid upper bound on the primal SDP value.

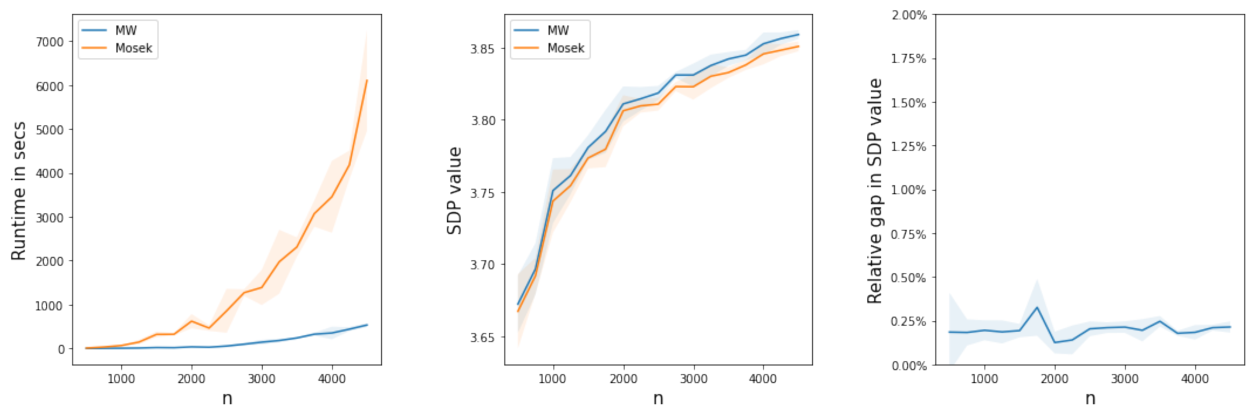

Figure 8: Comparison of the running time of Algorithm 2(left plot) with the MOSEK solver for PSD matrices of varying sizes (n from 500 to 4500). The middle plot shows the SDP values output by the two procedures. The right plot shows the relative error in the SDP value output by our procedure as compared to the value output by the MOSEK solver.

We consider two scenarios, one where M is a positive semi-definite matrix (PSD) and the other when M is a symmetric matrix with non-negative diagonal entries. In each case we compare the running time of our algorithm and the dual SDP value that it outputs with the corresponding values obtained by using a state of the art SDP solver on the same instance. As a comparison for our experiments we choose the SDP solver from the commercial optimization software MOSEK [ApS19]. For consistency, the comparison experiments were run on a single core of a Microsoft Surface Pro 3 tablet/computer with Intel Core i7-8650U CPU 1.90 GHz with 16GB RAM.

For the case when M is a PSD matrix, we generate a random $n \times n$ matrix A that contains entries drawn from a standard normal distribution and set $M = AA^\top$, and renormalize the matrix so that the trace is 1. We vary n from 500 to 4500 in increments of 250 (we used 5 random trials for each value of n). Figure 8 shows the comparison of our algorithm with the SDP solver from MOSEK for different values of n, averaged over the trials along with error bars. Notice that the running time of our algorithm is an order of magnitude faster than the MOSEK solver particularly for larger values of n, and furthermore the SDP values output are within 0.5% of the values output by exactly optimizing the SDP via the MOSEK solver. We also include a table of the average run times for a few different values of n in Table 2. We remark that other non-commercial SDP solvers like CVXPY [DB16] timed out even for for $n > 800$.

A similar trend holds in Figure 9 where we consider randomly generated instances of the more general QP problem (18). Here M is chosen to be a random symmetric $n \times n$ matrix with entries drawn from the standard normal distribution and replace the diagonal entries of M with their absolute values (so that the diagonals are non-negative).

Furthermore, unlike MOSEK, our procedure can scale to much larger values of n. As an example, Table 3 shows the running time needed for our procedure to perform 200 iterations on PSD matrices M of sizes ranging from 5000 to 2000. This demonstrates the scalability of our method for larger datasets with higher dimensions. For scalability purposes, the experiments below were run on a machine with access to a single GPU. Hence, the runtimes reported in Table 3 are not directly comparable to those in Table 2.

n	Running time (in s) of our Algorithm 2	Running time (in s) of MOSEK
500	1.384 ± 0.324	8.444 ± 0.389
1000	5.631 ± 1.190	66.578 ± 5.432
2000	35.747 ± 8.960	620.615 ± 161.96
3000	143.07 ± 17.29	1387.69 ± 398.26
4000	351.81 ± 142.97	3453.95 ± 818.50
4500	518.56 ± 27.81	5579.93 ± 523.53

Table 2: The average time (in seconds) taken by Algorithm 2 and the MOSEK solver on random PSD matrices for a few different sizes (n). The mean and standard deviation are computed over 5 independent runs for each value of n. See also Figure 8 for the plot based on more values of n taken in increments of 250.

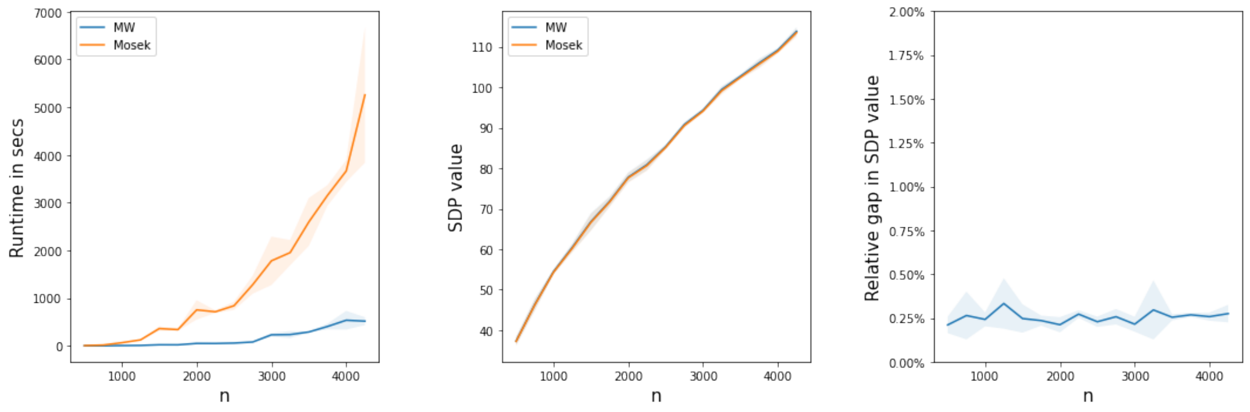

Figure 9: Comparison of the running time of Algorithm 2(left plot) with the MOSEK solver for general symmetric matrices with non-negative diagonal entries of varying sizes (n). The middle plot shows the SDP values output by the two procedures. The right plot shows the relative error in the SDP value output by our procedure as compared to the value output by the MOSEK solver.

n	Running time (in s) of our Algorithm 2
5000	104.92 ± 4.04
6000	156.29 ± 4.99
7000	213.80 ± 7.86
8000	272.83 ± 17.29
9000	346 ± 8.99
10000	378.70 ± 17.4
15000	1092.14 ± 103.79
20000	1801.08 ± 63.95

Table 3: The time (in seconds) taken by Algorithm 2 to perform 200 iterations on random PSD matrices of varying sizes (n). The mean and standard deviation are computed over 5 independent runs for each value of n.

We remark that our algorithm is specific to the Quadratic Programming SDP (which is a large class of problems in itself), while MOSEK SDP solver is a general purpose commercial SDP solver based on interior-point methods that is more accurate. Our MWU-based algorithm (Algorithm 2) for the Quadratic Programming SDP (12) gives much faster algorithms for approximately solving the SDP (along with dual certificate of the upper bounds), while not compromising much on approximation loss in the value. Hence the experiments are consistent with the running time improvements suggested by the theoretical results in Section 3.1.

F Robust Projections for Audio Data in DCT basis

We now perform an empirical evaluation on audio data to demonstrate the existence of good low-dimensional robust representations in the DCT basis. We consider the Mozilla CommonVoice speech-to-text audio dataset (english en_1488h_2019-12-10)[4] that was also considered by the work of Carlini et al. [CW18] on audio adversarial examples. We use a standard approach (that is also employed by [CW18]) to convert each audio file that may be of variable duration into a representation in high dimensional space. Each element x_i is a signed 16-bit value. First each audio file is converted into a sequence of overlapping frames corresponding to time windows; and each frame is represented as an n-dimensional vector of 16 bit values where n is given by the product of the window size and the sampling rate (for the Commonvoice dataset $n = 1200$). Hence each audio file corresponds to a sequence of frames, each of which is represented by an n dimensional vector.

Trial number	Number of frames N	$\infty \to 2$ norm	Projection error %
1	421469	21.285	4.974
2	480773	22.899	5.056
3	447353	21.962	5.578
4	461034	24.439	5.247
5	438954	23.840	5.392
6	452581	22.902	5.025

Table 4: The table shows SDP upper bounds on $\infty \to 2$ norm for projection matrices of rank $r = 200$ obtained by Algorithm 3 for 6 trials each with random 1000 audio samples from the Mozilla Common Voice dataset. Each random sample of 1000 audio samples corresponds to roughly $N \approx 10000$ frames each of $n = 1200$ dimension in the DCT basis.

For our experiments, in each random trial we consider 1000 randomly chosen audio files, and consider the data matrix (with n-dimensional columns) consisting of all the frames corresponding to these audio files (and we consider 6 such random trials). We first use the sparse PCA based heuristic (Algorithm 3 in Appendix C.1) to find a projection matrix of rank $r = 200$. We then use Algorithm 2 to compute upper bounds on the $\infty \to 2$ operator norm of the projections matrices. Table 4 shows the values of the operator norms certified by our algorithm for the projection matrices that are found, along with the reconstruction/ projection error, expressed as a percentage. Notice the obtained subspaces have operator norm values significantly smaller than $\sqrt{n} \approx 34.641$.

[4] https://voice.mozilla.org/en/datasets