Hollow Risk Minimization for Robustness: there is more info in your data

Yi Sun MIT
Ivan Ramirez MIT
Alfredo Cuesta-Infante Univ. Rey Juan Carlos
Kalyan Veeramachaneni MIT

Abstract

Recently, deep neural networks have been shown to be susceptible to disruption by adversarial examples. The problem of adversarial examples can be viewed as an over-fitting or false–structure problem, which can only be solved with prior knowledge. Adversarial training seems to be the most effective technique for overcoming the issue, but it often results in over-fitting to a specific adversarial perturbation (which constitutes a weak prior), leading to lower accuracy on clean inputs or other types of perturbations. In this work, we try to improve robustness not by relying on additional adversarial examples to solve the problem, but instead by exploiting more information that exists in the original data set. We propose a different learning scheme: Hollow Risk Minimization (HRM). In contrast to the standard Empirical Risk Minimization (ERM), HRM is able to exploit the geometry of the data, as well as the inner correlated errors, by training with *delocalized* gradients (hollow matrix). Our results show that this approach can naturally improve the robustness of neural networks without sacrificing accuracy, which is aligned with the over-fitting explanation of adversarial examples.

1 Introduction

Neural networks have proven to be excellent at making predictions in a wide field of applications, ranging from speech recognition [1, 2] to image classification [3] and image generation [4]. With the increasing availability of computational power, more advanced techniques like deep learning have boosted the performance and accuracy of such systems to unprecedented levels. As defined in [5], *"supervised learning is a high-dimensional interpolation problem"* where neural networks have been able to escape, to some extent, the curse of dimensionality. However, how they are able to do that is entangled with their behaviour in high dimensions which is often unclear and counter-intuitive [6].

Although neural networks have been proven to be *Universal Approximation Functions* [7], i.e., functions that can approximate almost any continuous function, the best way to train such functions is still an open question. Noisy measures, unbalanced data or a lack of samples usually lead to an over-fitting of the training data, undermining the quality of the learned model.

Moreover, recently, the reliability of these neural networks has been undercut by the discovery of adversarial examples – based a tiny modification of the input – can drastically change the output/prediction of the model [8]. Interestingly, adversarial examples have transferable properties and can successfully fool independently-trained neural networks. For this reason, they can be used to attack a model without direct access to that model (*black* box attacks), in contrast to those that have full access to it (*white* box attacks). See [9] for a detailed survey. Adversarial examples also make us reconsider the quality of those models – that is, their ability to generalize, measured as the different between train and test accuracy. If both are similarly high, we often conclude the model generalizes well. However, in practice, train and test sets come from the same dataset, and thus usually represent the same part of the actual distribution. If, for example, a CNN seems to generalize well (similar train and test accuracy), the model might actually be over-fitting "clean" image patterns, instead of images.

Preprint. Under review.

Since those patterns are present in both datasets, train and test accuracy are expected to be similar; meanwhile, "new" adversarial patterns are able to fool the model. In other words, a small difference between train and test accuracy is a necessary but not sufficient condition to claim generalization, as evidenced by adversarial examples.

The problem of adversarial examples has given rise to a growing body of work on robustness in machine learning. Numerous methods have been proposed for attacking neural networks with adversarial examples, as well as defending from them. In most of these approaches, a lack of robustness is considered to be a defection or imperfection of neural networks. The models are assumed to be fundamentally correct, and adversarial examples are just spurious errors that remain to be fixed with methods like adversarial training [10] where they are included in the training set. However, a simpler explanation would be that the models we train are false models (false–structure) which explains the lack of robustness. We can summarize this with the following question:

> *Are adversarial examples simply a side effect of neural networks or are they the proof of a lack of generalization?*

In this work, we aim to improve neural network robustness without using adversarial examples, obtaining models that are naturally robust. We will consider adversarial examples as an over-fitting [1] problem that weakens the ability of a model to generalize. Over-fitting requires regularization, which means the use of some prior knowledge to restrict the space of possible solutions. L2 and L1 regularization of neural networks aim at minimizing the energy and number of parameters, respectively. Dropout [11] does this by averaging models, which can also be seen as an approximate variational inference technique [12]. CNNs, for their part, introduce inductive bias with convolutions (infinitely strong priors). In this work, we propose to obtain this prior knowledge by better exploiting the information present in data – more precisely, by making use of cross-errors and the intrinsic geometry of the data.

The main contributions of this paper can be summarized as follows:

First, we propose a new learning paradigm to obtain higher levels of robustness without making use of adversarial examples, for two reasons: 1) adversarial training is computationally intense and may over-fit specific adversarial attacks and 2) the lack of robustness is an over-fitting/false–structure problem which must be addressed through regularization or prior knowledge. Second, we obtain such prior knowledge from the data itself, exploiting cross-errors (product of errors) that can capture the relationships of pairs of samples. To this end, we propose Hollow Risk Minimization (HRM), which makes use of *delocalized* gradients (gradients that depend on a pair of separated examples) by ignoring marginal errors. In contrast to the standard Empirical Risk Minimization (ERM), HRM allows the model to ignore outliers with high losses in the data, leading to a more robust model. Third, we analyse and provide a plausible explanation on the properties of HRM, in particular, how this learning paradigm is able to ignore samples through coupled gradients. Finally, our experiments show HRM leads to models with higher levels of robustness without sacrificing test accuracy, which further motivates the study of alternative learning methods to reduce adversarial examples.

2 Related Work

The existence of adversarial examples It is still unclear why adversarial examples exist in neural networks. Some argue that the set of adversarial examples lies in an extremely rare, high-dimensional "pocket" that can be found near each test sample [8]. This hypothesis was refuted in [10], where the authors argue that adversarial examples are instead a result of the linear nature of neural networks in high dimensions. In contrast, in [13], authors claim the linearity hypothesis is limited since linear classifiers do not usually suffer from adversarial examples. Instead they propose that adversarial examples exist because classification boundaries are close to the sub-manifold of data but tilted with respect to it, and that this phenomenon can be mitigated by regularization. For an exhaustive survey of proposed explanations, we refer the reader to [14].

Defenses against adversarial examples One line of research in designing robust models focuses on augmenting the data set with new examples for both empirical and certified defenses. One of the

[1] *'The so-called over-fitting phenomenon observed in experiments is actually a phenomenon of "false structure" known in the theory for solving ill-posed problems.'*

most successful empirical defenses is *adversarial training* ([10, 15–17]) with min-max formulation, where the inner maximization problem augments the training process with adversarial examples. Although effective in practice, this method often comes at the cost of decreased accuracy on clean inputs. On the other hand, [18] proposes a certifiable approach called *randomized smoothing*, where the input to a classifier is augmented with a Gaussian distribution. Some methods propose augmenting the data sets with linear interpolations between input points [19] or learned manifolds [20] in order to smooth decision boundaries. There are also techniques that consist of preprocessing the input. [21] proposes a Defensive Quantization method for improving accuracy under adversarial attacks. Others tackle the problem through gradient regularization [22].

Over-fitting to adversarial examples Almost all of the defenses mentioned above are tailored to one specific type of adversarial example, yet the question of finding an appropriate input perturbation set still remains. Despite the success of *adversarial training*, empirical research has found that *adversarial training* over-fits to one type of perturbation, which can lead the model to be even more susceptible to other types [23, 24]. Moreover, a model trained to be robust against ℓ_p-bounded attacks can still be fooled by simple unbounded attacks such as rotations and translations [25]. [26] shows that over-fitting to the adversarial training set can harm adversarial robustness in the test set. These results raise the question of whether *adversarial training* is over-fitting to the adversarial perturbation itself. In this paper, our goal is to address the adversarial robustness through a new learning paradigm, which does not over-fit to any perturbation sets.

3 Background

Empirical risk minimization (ERM), with its goal of driving individual training errors to zero, is prone to over-fitting and it may result in highly non-smooth decision boundaries. Vapnik and Izmailov [27] propose a new learning paradigm, V-matrix, to capture the geometric properties of training data, which leads to a smoother approximation of the function one is attempting to learn compared to ERM. Here we describe the link between Empirical Risk Minimization and the V-Matrix method, which inspires our framework of Hollow Risk Minimization.

3.1 Empirical Risk Minimization

First, we recall the standard problem of risk minimization widely used in supervised learning. Let $\{x_1, \cdots, x_i\},\ x_i \in \mathbb{R}^m$ be a set of given *i.i.d.* samples drawn from an unknown distribution, and let $\{y_1, \cdots, y_i\},\ y_i \in \mathbb{R}^n$ be their associated labels. The objective functional[2]to minimize is then defined by

$$R(f) = \int \mathcal{L}(y, f(x)) dP(x, y) \tag{1}$$

where $\mathcal{L}$ is a selected loss function, f represents the model, and P is the Cumulative Distribution Function (CDF) of the joint Probability Density Function $p(x, y)$. Recall that $dP(x, y) = p(x, y)dxdy$. In practice, since we do not have access to the true CDF P, Eq. 1 is replaced by the Empirical Risk Minimization (ERM) problem, where the objective functional is computed using the given set of samples, N. For instance, if we aim at minimizing the euclidean distance between target and prediction, $\mathcal{L}$ is set to match the Mean Square Error function, i.e.,

$$R_{emp}(f) = \frac{1}{N} \sum_{i}^{N} (y_i - f(x_i))^2. \tag{2}$$

In the rest of the paper, we shall focus on classification problems.

3.2 V-Matrix method

Second, we describe the V-Matrix method proposed in [27]. The goal when learning a classifier is to find a function $f(x)$ that approximates the conditional probability of a class label $p(y = c|x)$. To this

[2]We refer to a functional as a function of functions. It can also be referred as a Cost function.

end, [27] proposes to solve the following objective functional,

$$R_V(f) = \frac{1}{N^2} \int \left(\sum_i^N G(x - x_i)(y_i - f(x_i)) \right)^2 dP(x) \tag{3}$$

where $G(x - x_i)$ is any function in L_2, and $dP(x)$ is the probability density of x. In fact, this solution must be held for the whole set of G functions, and it is said that each selection constitutes a *statistical invariant*. This expression can be easily re-written as

$$R_V(f) = \frac{1}{N^2} \sum_{i,j} \underbrace{(y_i - f(x_i))(y_j - f(x_j))}_{\text{Cross Error Matrix (CEM)}} \underbrace{v(x_i, x_j)}_{V\text{-Matrix}} \tag{4}$$

where $v(x_i, x_j)$ are the entries of the V-Matrix defined as

$$v(x_i, x_j) = \int G(x - x_i) G(x - x_j) dP(x). \tag{5}$$

In contrast to ERM, this setting is able to: 1) incorporate cross–errors $(y_i - f(x_i))(y_j - f(x_j))$, which we will refer to as Cross Error Matrix (CEM), and 2) weight each term based on pairs of inputs (V-Matrix). A more detailed explanation of this V-Matrix method can be found in the appendix.

3.3 General loss functions lead to different upper-bounds

From Eq. 3, one can replace the term $(y_i - f(x_i))$, which computes the difference between target and prediction, with a different loss function. Among others and since we aim to solve a classification problem, we will make use of the Cross Entropy (CE) loss for its ability to reduce the vanishing gradient effect. This is,

$$R_{CE}(f) = \sum_{i,j}^N \left(y_i \log\left(f(x_i)\right) \cdot y_j \log\left(f(x_j)\right) \right)^{1/2} v(x_i, x_j) \tag{6}$$

where, $v(x_i, x_j)$ is defined as in Eq 5. Interestingly, if instead of the CE function, we use other statistical metrics as the Total Variation (TV) or the Kullback-Leibler divergence, it can be proven that: each resulting objective functionals for TV, KL and CE, constitute a set of ordered upper-bounds for the original functional $R_V(f)$. Concretely, we can obtain: $R_V(f) \leq R_{TV}(f) \leq R_{KL}(f) \leq R_{CE}(f)$. Further details can be found in the appendix. Finally, a general form of the target function can be written as

$$R_{\mathcal{L}}(f) = \sum_{i,j}^N \left(\mathcal{L}_i \cdot \mathcal{L}_j \right)^{1/2} v(x_i, x_j), \tag{7}$$

where $\mathcal{L}_i(y_i, f(x_i))$ is an arbitrary loss function.

4 Hollow Risk Minimization

The aim of this work is to train neural networks that are naturally robust for classification problems. The fundamental change that the V-Matrix method brings in practice is a modification of the objective functional (Eq. 4): first, cross-errors are considered, and second, each of them is weighted based on distances between inputs. However, the construction of the V-Matrix requires a priori definition of the set of $\{G\}$ functions, which is a difficult and application-dependent task. To illustrate this, let us consider the following two classification problems: 1) Dog detection problem and 2) Star moving problem. In the former, a model f_1 should determine whether a given picture shows a dog or not. Suppose f_1 is given a picture of a dog, and f_1 says it is a dog – we would expect that the same picture only modified by a few pixels should lead to the same result. In the latter problem, a model f_2 should determine if a star in a constellation image is approaching (blue) or moving away (red) because of the Doppler effect. Then, a tiny change in pixels color should effectively change the output of f_2, while changes on other pixels may not. For these reasons, the distances between pairs of samples should be measured according to their particular domains, which is often difficult to translate into G.

Because our aim is to remain as general as possible, we explore how each entry in the CEM should be treated so that robustness is improved, no matter the specific nature of the data (V-Matrix). In other

words, we will manually set the V-Matrix. Moreover, we aim to study the effect of using cross-errors exclusively in contrast to those in the diagonal.

To this end, we propose a hollow-type matrix to substitute for the V-Matrix. A hollow matrix is a square matrix with its diagonal entries set to zero. When we point-wise multiply the CEM by a hollow matrix, the result is also a hollow matrix. Removing the diagonal of a CEM matrix prevents the model from over-fitting on individual samples. Thus, we substitute the V-Matrix by $V = H = 1 - I$, where I is the identity matrix, resulting in:

$$R_H(f) = \frac{1}{N(N-1)} \sum_{i,j} \left(\mathcal{L}_i \cdot \mathcal{L}_j\right)^{1/2} h_{i,j}$$

$$= \frac{1}{N(N-1)} \sum \underbrace{\begin{bmatrix} \mathcal{L}_1 & \cdots & \sqrt{\mathcal{L}_1 \cdot \mathcal{L}_N} \\ \vdots & \ddots & \vdots \\ \sqrt{\mathcal{L}_j \cdot \mathcal{L}_1} & \cdots & \mathcal{L}_N \end{bmatrix}}_{CEM} \cdot \underbrace{\begin{bmatrix} 0 & \cdots & 1 \\ \vdots & 0 & \vdots \\ 1 & \cdots & 0 \end{bmatrix}}_{H}$$

where $h_{i,i}$ are the entries of the hollow matrix H. We here decide to divide by $1/(N(N-1))$ to keep a per-sample scale of the energy functional. Consequently, ERM shall be normalized by N and CEM by N^2.

4.1 HRM, ERM and CEM

HRM consists of the complementary cross–error terms of ERM, leading to CEM. That is, if $H = 1 - I$, then minimizing both HRM- and ERM-associated matrices is equivalent to minimizing the CEM. The fundamental difference between elements in diagonal and off-diagonal is that, while training, the associated gradients of the cross–errors terms are *delocalized* and coupled. That means the gradients depend on pair of samples $\nabla_\theta(x_i, x_j, y_i, y_j)$[3] that are also entangled. While ERM minimizes local errors, HRM combines them.

To better understand this *delocalized* phenomenon, let us consider a simple example where the set of points is $\{x_1, x_2\}$. We first consider HRM, ERM and CEM target functionals and their respective gradients. We leave a table for the general case (N examples) in appendix. Then, we have

$$R_{ERM} = \frac{1}{2} \sum \begin{bmatrix} \mathcal{L}_1 & 0 \\ 0 & \mathcal{L}_2 \end{bmatrix}, \; R_{HRM} = \frac{1}{2} \sum \begin{bmatrix} 0 & \sqrt{\mathcal{L}_1 \mathcal{L}_2} \\ \sqrt{\mathcal{L}_2 \mathcal{L}_1} & 0 \end{bmatrix},$$

$$R_{CEM} = \frac{1}{4} \sum \begin{bmatrix} \mathcal{L}_1 & \sqrt{\mathcal{L}_1 \mathcal{L}_2} \\ \sqrt{\mathcal{L}_2 \mathcal{L}_1} & \mathcal{L}_2 \end{bmatrix}$$

so

$$\nabla_{ERM} = \frac{1}{2}(\nabla_\theta \mathcal{L}_1 + \nabla_\theta \mathcal{L}_2), \; \nabla_{HRM} = \sqrt{\frac{\mathcal{L}_2}{\mathcal{L}_1}} \cdot \nabla_\theta \mathcal{L}_1 + \sqrt{\frac{\mathcal{L}_1}{\mathcal{L}_2}} \cdot \nabla_\theta \mathcal{L}_2,$$

$$\nabla_{CEM} = \left(\frac{1}{2} + \sqrt{\frac{\mathcal{L}_2}{\mathcal{L}_1}}\right) \cdot \nabla_\theta \mathcal{L}_1 + \left(\frac{1}{2} + \sqrt{\frac{\mathcal{L}_1}{\mathcal{L}_2}}\right) \cdot \nabla_\theta \mathcal{L}_2.$$

Furthermore, we will assume without loss of generality that $\mathcal{L}_2 > \mathcal{L}_1$. Figure 1 depicts the relative directions of each resulting gradient in the plane formed by $\nabla_\theta \mathcal{L}_1 \times \nabla_\theta \mathcal{L}_2$. Note that, $\nabla_\theta \mathcal{L}_1$ and $\nabla_\theta \mathcal{L}_2$ may not be orthogonal nor have the same magnitude, i.e., differences are relative. We highlight the following properties describing the differences in gradients illustrated in the aforementioned figure:

1. ERM treats gradients equally. Moreover, gradients of different samples are decoupled.
2. HRM instead couples gradients with losses of other samples. The resulting magnitude of the gradient of a given sample x_1, $\nabla_\theta \mathcal{L}_1$, depends on the aggregated losses of the remaining samples and, additionally, is inversely proportional to its own loss ($1/\sqrt{\mathcal{L}_i} \cdot \nabla_\theta \mathcal{L}_i$). These two mechanisms can ignore a sample if the majority has low loss and the sample itself has high loss. In other words, the model is not prone to fit samples with a higher loss than the majority. Those samples may be outliers, and ignoring them prevents the training procedure from finding a false model that over-fits all samples.

[3] θ denotes the set of parameters of a neural network.

3. CEM provides a middle ground between training accuracy (as promoted by ERM) and potential generalization ability (as promoted by HRM).

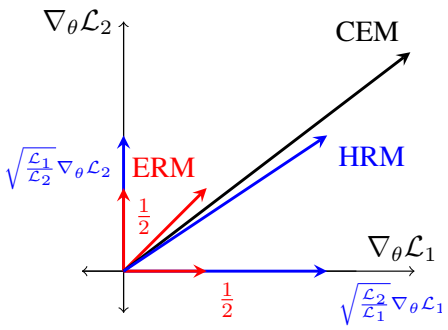


Figure 1: Each axis depicts the gradient direction for two given samples $\{x_1, x_2\}$. Note that both axes are orthogonal and of equal magnitude for the purpose of representation only. We assume $\mathcal{L}_2 > \mathcal{L}_1$. In this example, we graphically show how ERM promotes gradients proportionally to their associated losses on each sample. HRM on the contrary prevents movement in directions where the associated loss is high. This results in the ability to ignore samples with high error loss that may be outliers. CEM lies between both.

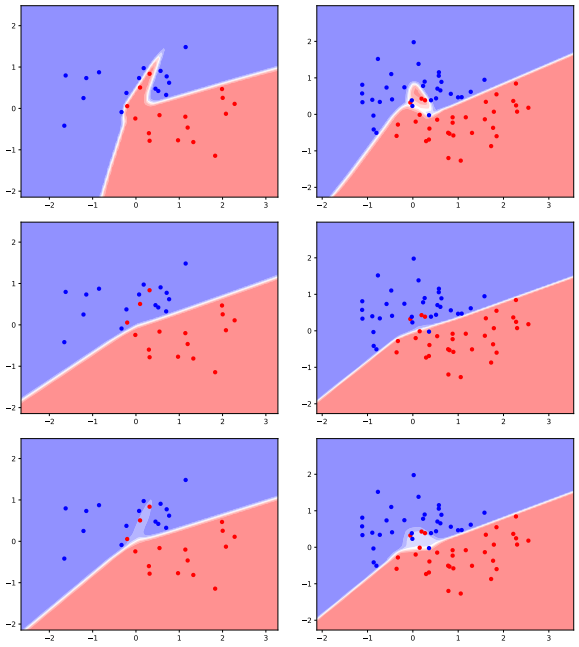


Figure 2: Each row depicts results for ERM, HRM and CEM, respectively. The results show HRM is able to ignore some training points in contrast to ERM and CEM.

Toy Example To illustrate the findings presented above, let us consider a practical toy example. We compare the resulting boundary decisions for these 3 target functions using a dataset with very few examples. In Figure 1 qualitative differences in the final boundary are revealed. HRM is clearly able to ignore training points, while ERM adapts to fit the whole training set. CEM gives a middle ground between both. In addition, in Figure 3, we depict the variety of gradient directions during the training step. The contribution of cross-error gradients is clearly distinct. The gradients of HRM and CEM are more consistent, indicating that the training process is more stable, which is a known property for improving the generalization of models [28]. A measure of these models' differences is reported in Table 4.1, where we compute the euclidean distance between parameters of trained models $\{\theta_{ERM}, \theta_{HRM}, \theta_{CEM}\}$. The results show that θ_{HRM} is significantly different from θ_{ERM} as the matrix increases, getting closer to CEM when the cross-error terms tend to dominate and CEM and HRM get closer.

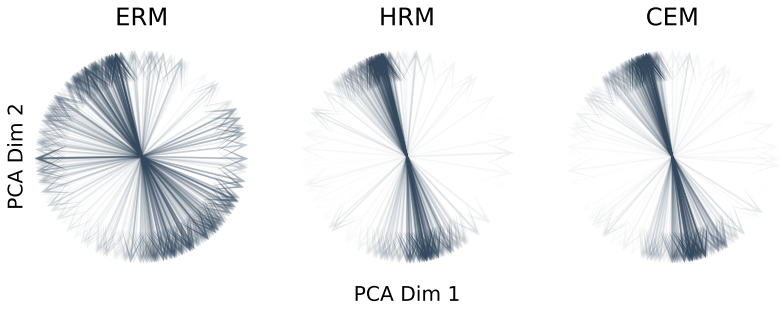


Figure 3: PCA analysis of training gradients. All gradients are normalized, and the intensity represents their magnitude. ERM and CEM present a high-correlated direction distribution with several modes where the gradients are concentrated. HRM shows a significantly different distribution of gradients.

Batch size	$\|\|\theta_{ERM} - \theta_{HRM}\|\|_2$	$\|\|\theta_{ERM} - \theta_{CEM}\|\|_2$	$\|\|\theta_{CEM} - \theta_{HRM}\|\|_2$
8	2.391	2.385	0.927
16	2.807	2.274	0.780
32	2.572	2.473	0.242

Table 1: Results show the euclidean distance between trained models. We highlight the last column: when the batch size increases, CEM and HRM get closer because the diagonal is less dominant. The euclidean distance is computed for a vector of 898 parameters in a 5-layer neural network.

5 Experiments

In this section, we evaluate the robustness of CEM and HRM method. To do so we train Lenet5 for MNIST and Resnet-50 for CIFAR10 datasets using one of the three objective functions: CEM, HRM and ERM (baseline). We then attack these models using norm-bounded attacks and report their performance on the adversarial examples. We also report performance of a given trained model, say trained using CEM, on the adversarial examples generated for another trained model, say trained using ERM.

Experiment settings All experiments reported here are trained for 100 epochs with batch size 128, and using Adam optimizer with base learning rate 0.1. For the norm bounded attack methods, we use Fast Gradient Signed Method (FGSM) and the Projected Gradient Descent (PGD).

Results We present the results of ERM, CEM and HRM under adversarial attack for MNIST (Table 2) and CIFAR10 (Table 3). With HRM, the model is able to improve robustness on MNIST to 56.22% and robustness on CIFAR10 to 36.05% without sacrificing any of the clean inputs' test accuracy (for PGD). This is aligned with the assumption that HRM avoids over-fitting on outliers. The gains in robustness can be achieved just by improving the generalization of the model, which is not at odds with accuracy on clean inputs, as is known in *adversarial training* [29].

	Test Acc	FGSM (ϵ = 0.3)	PGD (ϵ = 0.3)
ERM (Baseline)	98.71%	24.55%	3.25%
CEM	98.69%	72.98%	41.20%
HRM	98.54%	83.80%	56.22%

Table 2: MNIST: FGSM and PGD (40 steps) attack with $\epsilon = 0.3$. All number represent the resulting accuracy. Results show a significant improvement under adversarial attacks created with the given model. The accuracy remains similarly high.

	Test Acc	FGSM (ϵ =16)	PGD (ϵ = 4)	PGD (ϵ = 8)
ERM (Baseline)	90.58%	21.28%	16.68%	12.49%
CEM	90.93%	46.43%	42.35%	35.56%
HRM	90.21%	46.70%	43.24%	36.05%

Table 3: CIFAR10: FGSM ($\epsilon = 16/255$) and PGD ($\epsilon = 4/255, 8/255$, 20 steps) attack. All number represent the resulting accuracy. Although the accuracy is lower, HRM remains to be the most robust one.

Different Objective Function Attacks We also investigate a setting where the model is trained with either CEM or HRM, but tested with attacks that are based on a different objective functional, in particular ERM. This is to make sure the gains from HRM are not simply due to gradient masking. As shown in Table 4, HRM remains robust under a different loss function at attack time. We leave more experiments with different batch sizes in appendix.

Transfer Attack Adversarial examples are known to have properties of transferability, where adversarial examples generated from one model are able to effectively attack a different model. In

	FGSM (ϵ =16)	PGD (ϵ = 4)	PGD (ϵ = 8)
Baseline ERM	21.28%	16.68%	12.49%
CEM	46.43%	43.57%	35.04%
HRM	46.70%	43.25%	35.53%

Table 4: CIFAR10-Attack using cross entropy loss. FGSM ($\epsilon = 16/255$) and PGD ($\epsilon = 4/255, 8/255$) attack. All number represent the resulting accuracy. Here the adversarial attacks are created with a different loss function (CE) than the one used for training. This allows us to check whether a masking gradient effect may be falsifying the results. Accuracies remain similar concluding there is no such effect.

this section, we address the question of: how successful are the adversarial examples generated for models using ERM are on models generated using CEM or HRM. If CEM or HRM are robust they should be robust to these examples. In table-5, models trained with CEM and HRM are slightly more robust under transferred attacks than under their own. We can draw two conclusions from these results: 1) no masking gradient effect is taking place and 2) the ineffectiveness of attacks from ERM indicates that the final models obtained are significantly different. Interestingly, neither CEM nor HRM are able to create attacks as effective as those that ERM creates. We expect an ideal model will not suffer from adversarial examples, meaning that the adversarial perturbation required to change the output of the model is so strong that the adversarial example is no longer a valid sample.

Source \ Target	ERM	CEM	HRM
ERM	12.40%	42.22%	38.14%
CEM	48.51%	35.05%	47.71%
HRM	47.73%	48.73%	35.40%

Table 5: CIFAR10: Black-box PGD attack. Adversarial examples are created with PGD with 7 steps and $\epsilon = 8$ on the source network, and evaluated on an independently-trained target network.

6 Conclusions

In this work, we presented HRM, a new learning paradigm for obtaining natural robustness in neural networks. Unlike the standard ERM, which minimizes individual errors, HRM minimizes cross-errors, which constitutes a *delocalized* or global prior. This promotes a mechanism by which potential outliers are ignored, reducing over-fitting and increasing generalization and robustness.

The results show a significant improvement in robustness (adversarial test) and preservation of test accuracy, all without introducing any adversarial example during training. This shows that data contains useful information that is not exploited in standard methods, such as ERM.

To conclude, we have claimed that adversarial examples place into doubt the generalization ability of a neural network. We have identified the lack of robustness as an over-fitting or false-structure problem, and have addressed the problem through regularization, which requires prior knowledge. We have proposed a learning paradigm to obtain such prior knowledge directly from data, and we have showed that it is possible to achieve more robust models in this way.

7 Broader Impact

Questions about robustness and adversarial examples are important to more than just field experts. Classification systems can be found in many critical decision-making processes; so even non-experts can understand the risk involved in adversarial examples and attacks. Currently, the popular way to address this threat is adversarial training. We claim that this strategy may not be sufficient. Adversarial training is an ad-hoc form of data augmentation that extends the training set towards regions that otherwise could be misclassified; so, in the end, the model is over-fitted to this extended space rather than enabled to generalize.

We hope this work motives new learning approaches which, ideally, do not suffer from adversarial examples. Our major contribution in this paper is to show this is possible by rethinking our learning paradigms. However, we are not free of limitations, and further theoretical analysis and practical experiments must be carried out on our proposed HRM. For example, why the combined errors should be multiplied? Is there a more general function to couple them? Why not consider more than two examples at a time? We envisage this as future work.

References

[1] Geoffrey Hinton, Li Deng, Dong Yu, George E Dahl, Abdel-rahman Mohamed, Navdeep Jaitly, Andrew Senior, Vincent Vanhoucke, Patrick Nguyen, Tara N Sainath, et al. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal processing magazine*, 29(6):82–97, 2012.

[2] Alex Graves, Abdel-rahman Mohamed, and Geoffrey Hinton. Speech recognition with deep recurrent neural networks. In *2013 IEEE international conference on acoustics, speech and signal processing*, pages 6645–6649. IEEE, 2013.

[3] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[4] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

[5] Stéphane Mallat. Understanding deep convolutional networks. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 374(2065):20150203, 2016.

[6] Michel Verleysen, Damien Francois, Geoffroy Simon, and Vincent Wertz. On the effects of dimensionality on data analysis with neural networks. In *International Work-Conference on Artificial Neural Networks*, pages 105–112. Springer, 2003.

[7] George Cybenko. Approximation by superpositions of a sigmoidal function. *Mathematics of control, signals and systems*, 2(4):303–314, 1989.

[8] 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.

[9] Alexey Kurakin, Ian Goodfellow, Samy Bengio, Yinpeng Dong, Fangzhou Liao, Ming Liang, Tianyu Pang, Jun Zhu, Xiaolin Hu, Cihang Xie, et al. Adversarial attacks and defences competition. In *The NIPS'17 Competition: Building Intelligent Systems*, pages 195–231. Springer, 2018.

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

[11] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958, 2014.

[12] Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, pages 1050–1059, 2016.

[13] Thomas Tanay and Lewis Griffin. A boundary tilting persepective on the phenomenon of adversarial examples. *arXiv preprint arXiv:1608.07690*, 2016.

[14] Alexandru Constantin Serban, Erik Poll, and Joost Visser. Adversarial examples-a complete characterisation of the phenomenon. *arXiv preprint arXiv:1810.01185*, 2018.

[15] Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial machine learning at scale. *arXiv preprint arXiv:1611.01236*, 2016.

[16] Florian Tramèr, Alexey Kurakin, Nicolas Papernot, Ian Goodfellow, Dan Boneh, and Patrick McDaniel. Ensemble adversarial training: Attacks and defenses. *arXiv preprint arXiv:1705.07204*, 2017.

[17] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *ArXiv*, abs/1706.06083, 2017.

[18] Jeremy M. Cohen, Elan Rosenfeld, and J. Zico Kolter. Certified adversarial robustness via randomized smoothing. In *ICML*, 2019.

[19] Hongyi Zhang, Moustapha Cissé, Yann Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. *ArXiv*, abs/1710.09412, 2018.

[20] Vikas Verma, Alex Lamb, Christopher Beckham, Aaron C. Courville, Ioannis Mitliagkas, and Yoshua Bengio. Manifold mixup: Encouraging meaningful on-manifold interpolation as a regularizer. *ArXiv*, abs/1806.05236, 2018.

[21] Ji Lin, Chuang Gan, and Song Han. Defensive quantization: When efficiency meets robustness. *arXiv preprint arXiv:1904.08444*, 2019.

[22] Andrew Slavin Ross and Finale Doshi-Velez. Improving the adversarial robustness and interpretability of deep neural networks by regularizing their input gradients. In *Thirty-second AAAI conference on artificial intelligence*, 2018.

[23] Lukas Schott, Jonas Rauber, Matthias Bethge, and Wieland Brendel. Towards the first adversarially robust neural network model on mnist. *arXiv: Computer Vision and Pattern Recognition*, pages 1–17, 2019.

[24] Yash Sharma and Pin-Yu Chen. Attacking the madry defense model with l1-based adversarial examples. *ArXiv*, abs/1710.10733, 2017.

[25] Logan Engstrom, Dimitris Tsipras, Ludwig Schmidt, and Aleksander Madry. A rotation and a translation suffice: Fooling cnns with simple transformations. *ArXiv*, abs/1712.02779, 2017.

[26] Leslie Rice, Eric Wong, and J. Zico Kolter. Overfitting in adversarially robust deep learning. *ArXiv*, abs/2002.11569, 2020.

[27] Vladimir Vapnik and Rauf Izmailov. V-matrix method of solving statistical inference problems. *Journal of Machine Learning Research*, 16(51):1683–1730, 2015.

[28] Olivier Bousquet and André Elisseeff. Stability and generalization. *J. Mach. Learn. Res.*, 2: 499–526, 2002.

[29] Dimitris Tsipras, Shibani Santurkar, Logan Engstrom, Alexander Turner, and Aleksander Madry. Robustness may be at odds with accuracy. *arXiv: Machine Learning*, 2018.

8 Appendix

8.1 V-Matrix method derivation

In the following, we describe the *Direct Constructive Setting* for conditional probability estimation that leads to the V-Matrix method proposed by Vapnik and Izmailov [27]. Thus, we consider to estimate the conditional probability $p(y = c|x)$, which is to solve a classification problem where the output is a categorical variable. Starting from Bayes equation we have

$$p(y = c|x)p(x) = p(y = c, x). \tag{8}$$

We argue as follow: both sides of the equation can be multiplied by some function $G(x - x') \in L_2$ and integrated leading to

$$\int G(x - x')f(x')dP(x') = \int G(x - x')dP(y = c, x') \tag{9}$$

where $f(x) = p(y = c|x)$ is the solution of the integral equation. Again, we only have access to samples (training set) from the real distribution, thus, the equation above leads to

$$\frac{1}{N}\sum_{i}^{N} G(x - x_i)f(x_i) = \frac{1}{N}\sum_{i}^{N} \mathbf{1}\{y_i = c\}G(x - x_i). \tag{10}$$

Here $\mathbf{1}\{\cdot\}$ is the indicator function, which can be replaced by y_i if a one hot codification is considered, i.e., each component of vector y_i, takes two values: $y_i^d \in \{0, 1\}$. Eq. 9 and it's empirical version Eq. 10 hold true for any function $G \in L_2$ leading to a set of equalities called *statistical invariants* for f. As a consequence and for the sake of feasibility, it is necessary to select a subset of functions $\{G_1, ..., G_m\}$ which are referred to as *predicates*. This constitutes an arbitrary election which need to be motivated by some prior knowledge, which in turn depends on the application or inference task.

To solve this problem means to find a function $f \in [0, 1]$ such that Eq. 10 holds true. To this end, one can choose to minimize the L_2 distance of both terms obtaining the following objective functional:

$$R(f) = \frac{1}{N^2}\int \left(\sum_{i}^{N} G(x - x_i)f(x_i) - \sum_{j}^{N} y_i G(x - x_j)\right)^2 dP(x) \tag{11}$$

i.e.

$$R(f) = \frac{1}{N^2}\int \left(\sum_{i}^{N} G(x - x_i)(y_i - f(x_i))\right)^2 dP(x). \tag{12}$$

Finally, this equation can be re-written as

$$R_V(f) = \frac{1}{N^2}\sum_{i,j} \underbrace{(y_i - f(x_i))(y_j - f(x_j))}_{\text{Cross Error Matrix CEM}} \underbrace{v(x_i, x_j)}_{V\text{-Matrix}} \tag{13}$$

where $v(x_i, x_j)$ are the entries of the V-Matrix defined as

$$v(x_i, x_j) = \int G(x - x_i)G(x - x_j)dP(x). \tag{14}$$

It is easy to see that, if the V-Matrix is set to the identity function, we recover the ERM Eq. 2. Roughly speaking, the V-Matrix allows us to weight every cross–term $(y_i - f(x_i))(y_j - f(x_j))$ in the Cross Error Matrix (CEM) and minimize it, which, in turn, provides a co-related loss between samples. These cross–errors weighted by their associated V-Matrix entries are actually able to capture the geometry of the given distribution, and train a model accordingly.

Two conclusions can be drawn from the V-Matrix method:

- First, in contrast to the Empirical Risk minimization,the cross–errors are taken into consideration, which is already a conceptual improvement. Not only individual errors must be reduced, but also their *delocalized* errors.

- Second, the relative importance of such cross terms is driven by the V-Matrix. This, however, must be defined through $\{G_m\}$ a priori.

In this sense, the solution of the problem belongs to a set of admissible functions $\{f_k\}$ which is restricted by information coming from the data itself, but unused in classic approaches to machine learning.

8.2 General loss functions lead to different upper-bounds - Extended

When it comes to solve classification problems, it is well known in machine learning literature how minimizing the Cross Entropy against MSE improves the optimization step. This is achieved by reducing the vanishing gradient effect that arises when using a MSE loss function. For such a reason we propose to modify the objective function 3, to match a different loss function. For example, we can replace the difference term $(y_i - f(x_i))$ by the Total Variation [4],

$$R_{TV}(f) = \int \left(\sum_i^N G(x - x_i) \frac{1}{2} ||y_i - f(x_i)||_1 \right)^2 dP(x) \tag{15}$$

which can be upper-bounded using Pinsker's inequality[5] with the Kullback-Leibler, which is a particular case of a broader family of measures (Rènyi α-divergences).

$$R_{KL}(f) = \int \left(\sum_i^N G(x - x_i) \left(-y_i \log \left(\frac{f(x_i)}{y_i} \right) \right)^{1/2} \right)^2 dP(x) \tag{16}$$

Similarly, using that $CE(q, p) = \mathrm{H}(p) + KL(q||p)$, the extensively used Cross Entropy can be obtained

$$R_{CE}(f) = \int \left(\sum_i^N G(x - x_i) \left(-y_i \log \left(f(x_i) \right) \right)^{1/2} \right)^2 dP(x) \tag{17}$$

$$= \sum_{i,j}^N \left(y_i \log \left(f(x_i) \right) \cdot y_j \log \left(f(x_j) \right) \right)^{1/2} v(x_i, x_j) \tag{18}$$

where, $v(x_i, x_j)$ is defined as in Eq 5 and H is the Entropy.

In fact, this 3 options constitute an ordered set of upper-bounds of the original V-Matrix (Eq. 19):

$$R_V(f) \leq R_{TV}(f) \leq R_{KL}(f) \leq R_{CE}(f).$$

Those are just a few examples showing how the V-Matrix method can be generalized to obtain different loss functions while keeping the same structured functional. Thus, a general expression of the target functional we aim to minimize is

$$R_{\mathcal{L}}(f) = \sum_{i,j}^N \left(\mathcal{L}_i \cdot \mathcal{L}_j \right)^{1/2} v(x_i, x_j), \tag{19}$$

where $\mathcal{L}_i(y_i, f(x_i))$ is an arbitrary loss function.

8.3 Additional experiments results with different toy-datasets

Figure 4 depicts the resulting boundaries for a set of four datasets trained with ERM and HRM. Boundaries in HRM are less complex and smoother than those in ERM. HRM shows, as we have said repeatedly, the ability to ignore points that may be outliers of very noisy samples.

[4] R_{TV} upper bounds R_V using $(y_i - f(x_i)) \leq |y_i - f(x_i)|$

[5] Pinsker's Inequality: $\frac{1}{2}||p - q||_1 \leq \sqrt{2\mathrm{KL}(p||q)}$

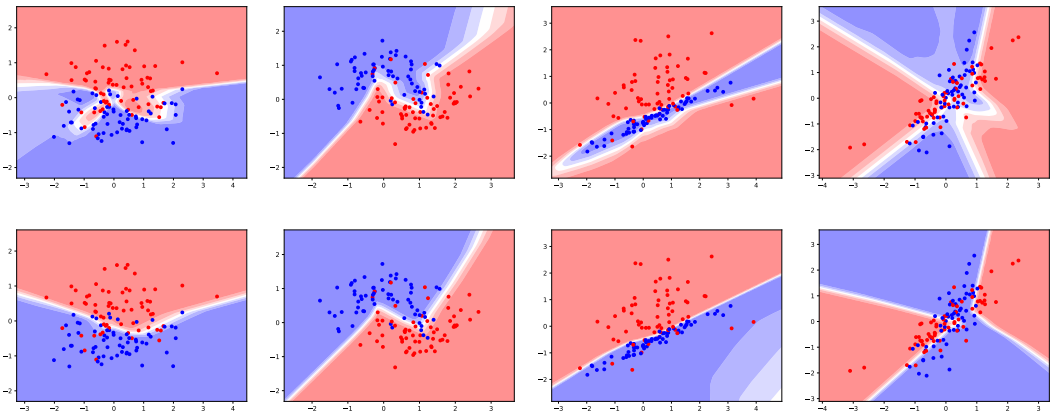


Figure 4: ERM (first row) and HRM (second row) models comparison. Under the same training conditions, HRM results in a smoother an simpler model due to its ability to ignore samples that are potential outliers.

8.4 Intuition behind V-Matrix and G-functions

The correct selection of a G function that leads to a V-Matrix is challenging because, first, it requires an understanding of the nature and relationships in data points and second, such prior knowledge must be translate to G. Let us consider that G is a kernel function that measures a distance between to points x_1, x_2, and let the whole set of points be defined as $X = \{x_1, x_2, x_3, x_4, x_5\}$. Given the definition of $v(x_i, x_j)$ as

$$v(x_i, x_j) = \int G(x - x_i)G(x - x_j)dP(x) \tag{20}$$

and its empirical version

$$v_{emp}(x_i, x_j) = \sum_k G(x_k - x_i)G(x_k - x_j). \tag{21}$$

Let us also consider G a Radial Basis Function (RBF)

$$G(x_i, x_j) = \exp(-\gamma||x_i - x_j||^2)$$

then, the G value of a pair of samples x_i, x_j decreases when the euclidean distance increases. Figure 5 depicts all RBF values for the considered pair of samples x_1, x_2. Eq. 21 can be seen as the inner product of two vectors, each of them, where each component contains the value of G for each sample of the distribution. This product gives a notion of distance between: 1) the distance of a sample against a distribution and 2) the distance of another sample against the same distribution. If both samples are equally distant to the distribution, the inner product will be maximal. In other words, $G(x_1, x_2)$ controls the euclidean-based distance between x_1, x_2, while $v(x_1, x_2)$ controls the relative distance of the two points against the whole distribution.

8.5 Additional experiments results with different G-functions

G functions We also explore with more sophisticated v-matrix to better capture the geometry of the training data. Proposed G functions:

- Identity function (CEM): $G(x_i, x_j) = 1$
- Radial basis function (RBF) Kernel: $G(x_i, x_j) = exp(-\gamma||x_i - x_j||^2)$
- Hyper-Laplacian Kernel: $G(x_i, x_j) = exp(-\alpha|x_i - x_j|^p), \quad 0 < p \leq 1$
- Modified Cosine Distance: $G(x_i, x_j) = 0.5 * (1 + < x_i, x_j > / (||x_i|| \cdot ||x_j||))$

When using RBF and $\gamma \to 0$, G approximates to a matrix with all ones, where each pair of cross-errors are weighted equally as in CEM. When $\gamma \to \infty$, G tends to be the identity matrix, and the

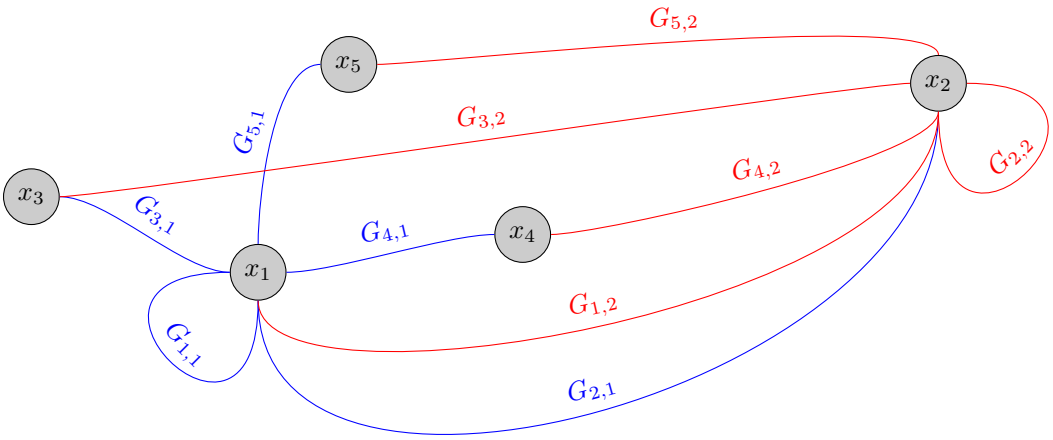


Figure 5: Samples x_1, x_2 are considered. Each $G_{i,j}$ represents the value for $G(x_i - x_j) = \exp(-\gamma||x_i - x_j||^2)$ for $i, j = \{1, 2, 3, 4, 5\}$. The associated V-Matrix entry is then $v(x_1, x_2) = \sum_{k=1}^{5} G_{k,1}G_{k,2}$.

objective function goes back to the regular ERM. We also test with other G to show how different can result the final models which reminds us of the importance of selecting an appropriate distance depending on the nature of the data.

	Test Acc	FGSM (ϵ =0.3)	PGD (ϵ = 0.3)
ERM (Baseline)	98.71%	24.55%	3.25%
Hyper Laplacian (p=0.5) with HRM	98.77%	45.87%	12.93%
RBF with HRM	98.66%	82.64%	58.11%

Table 6: MNIST: FGSM ($\epsilon = 0.3$) and PGD ($\epsilon = 0.3$, 40 steps) attack. Batchsize: 128.

Tables 6 and 7 show the Hyper-Laplacian election for G implies a severe drop in accurary compared to RBF. The Hyper-Laplacian function, in contrast to RBF, penalizes small differences between samples while big differences are treated more equally. This can promote the model to strongly discriminate between samples that are similar, and thus, to over-fit the distribution. Cosine Distance, however, discards the magnitude of samples (treated as vectors) and focus of their alignment. This seems to better capture the inner geometry of the data, increasing the robustness while keeping similar test accuracy to ERM.

	Test Acc	FGSM (ϵ =16)	PGD (ϵ = 4)	PGD (ϵ = 8)
ERM (Baseline)	90.58%	21.28%	16.68%	12.49%
Hyper Laplacian (p=0.5) with HRM	80.37%	9.10%	0.00%	0.00%
RBF with HRM	90.00%	43.32%	40.41%	33.12%
Mod. Cosine Dis with HRM	90.51%	46.62%	42.34%	34.32%

Table 7: CIFAR10: FGSM ($\epsilon = 16/255$) and PGD ($\epsilon = 4/255, 8/255$) attack. Batchsize: 128.

8.6 Additional experiments results on varying batch size

Tables 8, 9 and 10 show the results for different batch sizes. Adversarial accuracy generally improves when the batch size increases. Figure 6 depicts the dominance effect of the diagonal when the batch size is decreased, which motives the use of HRM in that scenarios instead of CEM.

	Test Acc	FGSM (ϵ =16)	PGD (ϵ = 4)	PGD (ϵ = 8)
ERM (Baseline)	90.34%	30.07%	26.86%	21.47%
CEM	90.17%	53.21%	50.24%	42.77%
HRM	90.62%	54.52%	50.63%	43.94%

Table 8: CIFAR10: FGSM ($\epsilon = 16/255$) and PGD ($\epsilon = 4/255, 8/255$) attack. Batch size: 256.

	Test Acc	FGSM (ϵ =16)	PGD (ϵ = 4)	PGD (ϵ = 8)
ERM (Baseline)	90.58%	21.28%	16.68%	12.49%
CEM	90.93%	46.43%	42.35%	35.56%
HRM	90.21%	46.70%	43.24%	36.05%

Table 9: CIFAR10: FGSM ($\epsilon = 16/255$) and PGD ($\epsilon = 4/255, 8/255$) attack. Batch size: 128.

	Test Acc	FGSM (ϵ =16)	PGD (ϵ = 4)	PGD (ϵ = 8)
ERM (Baseline)	90.79%	17.42%	12.38%	8.23%
CEM	89.72%	36.66%	32.70%	23.32%
HRM	90.12%	44.23%	39.33%	31.18%

Table 10: CIFAR10: FGSM ($\epsilon = 16/255$) and PGD ($\epsilon = 4/255, 8/255$) attack. Batch size: 64.

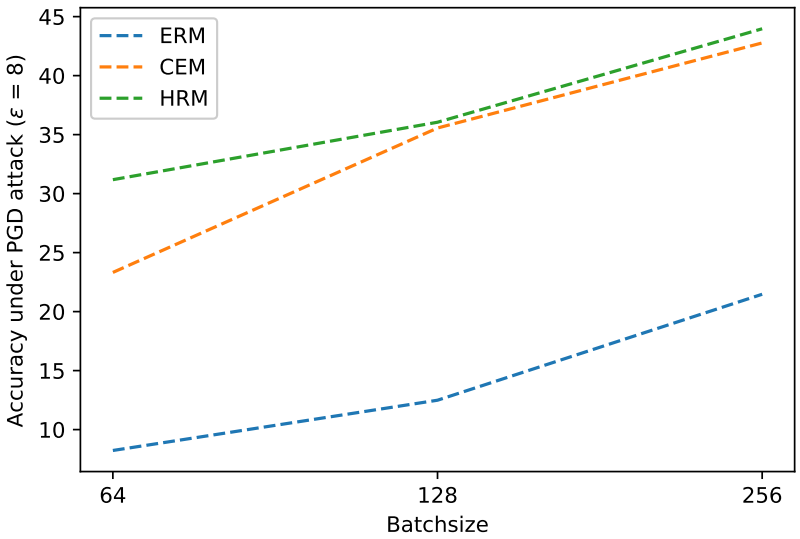
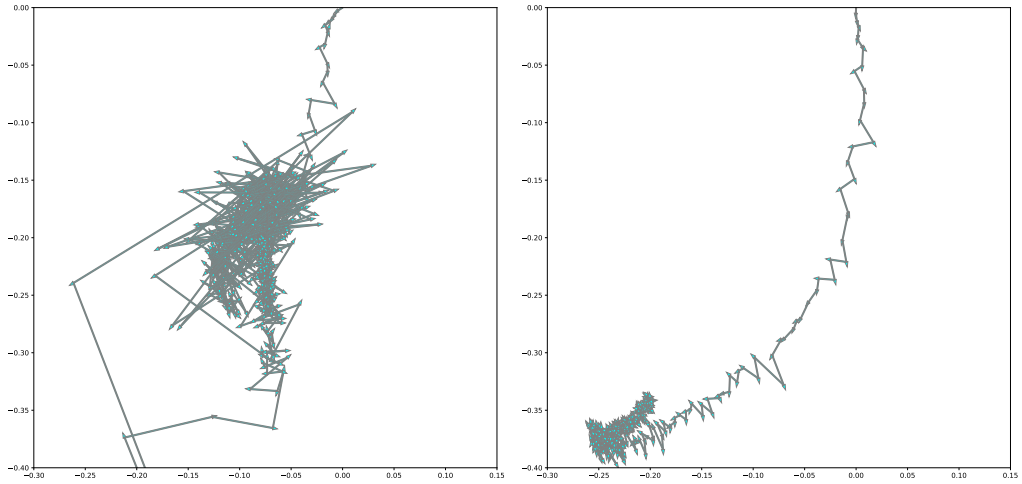


Figure 6: Varying training batch size. When the batch size decreases, the diagonal of CEM tends to dominate, and the adversarial accuracy drops w.r.t. HRM.

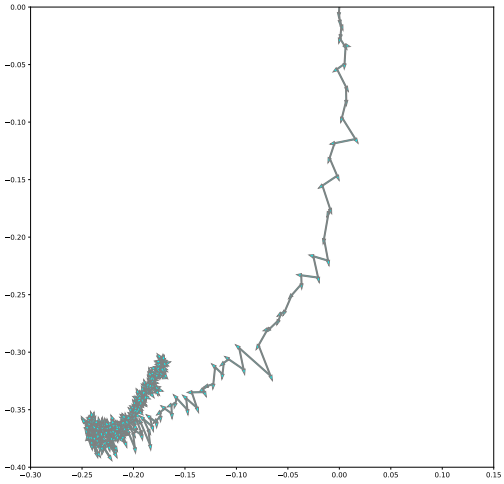
8.7 Training Stability

In previous sections, we have seen how increasing the batch size improves robustness, not only for our proposed HRM but also in standard ERM training. We show in Figure 7 the training process (gradients) to achieve the optimal model (parameters). For the sake of visualization we reduce the dimension of both gradients and parameters to their 2 principal components (PCA). The results show a more stable behaviour when cross-errors are involved in the training process.



(a) ERM

(b) HRM



(c) CEM

Figure 7: Training convergence of ERM (a), HRM (b) and CEM (c). Each arrow is a PCA reduction of the actual gradient that updates the weights, also PCA-reduced. HRM provides stability to the training process which is a known property for improve generalization.

8.8 Gradient Analysis Table

	Objective Functional $R(f,\theta)$	Gradient $\nabla_\theta R$
ERM	$\frac{1}{N}\sum_i \mathcal{L}_i$	$\frac{1}{N}\sum_i \nabla_\theta \mathcal{L}_i$
HRM	$\frac{1}{N(N-1)}\sum_{i\neq j}\sqrt{\mathcal{L}_i \cdot \mathcal{L}_j}$	$\sum_{i\neq j}\frac{1}{N(N-1)}\left(\sqrt{\frac{\mathcal{L}_j}{\mathcal{L}_i}}\cdot\nabla_\theta\mathcal{L}_i + \sqrt{\frac{\mathcal{L}_i}{\mathcal{L}_j}}\cdot\nabla_\theta\mathcal{L}_j\right)$
CEM	$\frac{1}{N^2}\sum_{i,j}\sqrt{\mathcal{L}_i \cdot \mathcal{L}_j}$	$\sum_{i,j}\frac{1}{N^2}\left(\sqrt{\frac{\mathcal{L}_j}{\mathcal{L}_i}}\cdot\nabla_\theta\mathcal{L}_i + \sqrt{\frac{\mathcal{L}_i}{\mathcal{L}_j}}\cdot\nabla_\theta\mathcal{L}_j\right)$

Table 11: Target Functionals (Cost functions) and Gradients for ERM, HRM and CEM for a set of samples: $\{x_1, x_2, ..., x_N\}$.