We investigate the HSIC (Hilbert-Schmidt independence criterion) bottleneck as a regularizer for learning an adversarially robust deep neural network classifier. In addition to the usual cross-entropy loss, we add regularization terms for every intermediate layer to ensure that the latent representations retain useful information for output prediction while reducing redundant information. We show that the HSIC bottleneck enhances robustness to adversarial attacks both theoretically and experimentally. In particular, we prove that the HSIC bottleneck regularizer reduces the sensitivity of the classifier to adversarial examples. Our experiments on multiple benchmark datasets and architectures demonstrate that incorporating an HSIC bottleneck regularizer attains competitive natural accuracy and improves adversarial robustness, both with and without adversarial examples during training. Our code and adversarially robust models are publicly available[1].

1 Introduction

Adversarial attacks [8, 17, 18, 3, 5] to deep neural networks (DNNs) have received considerable attention recently. Such attacks are intentionally crafted to change prediction outcomes, e.g., by adding visually imperceptible perturbations to the original, natural examples [25]. Adversarial robustness, i.e., the ability of a trained model to maintain its predictive power under such attacks, is an important property for many safety-critical applications [4, 6, 26]. The most common approach to construct adversarially robust models is via adversarial training [34, 36, 30], i.e., training the model over adversarially constructed samples.

Alemi et al. [11] propose using the so-called Information Bottleneck (IB) [27, 28] to enhance adversarial robustness. Proposed by Tishby and Zaslavsky [28], the information bottleneck expresses a tradeoff between (a) the mutual information of the input and latent layers vs. (b) the mutual information between latent layers and the output. Alemi et al. show empirically that using IB as a learning objective for DNNs indeed leads to better adversarial robustness. Intuitively, the IB objective increases the entropy between input and latent layers; in turn, this also increases the model’s robustness, as it makes latent layers less sensitive to input perturbations.

Nevertheless, mutual information is notoriously expensive to compute. The Hilbert-Schmidt independence criterion (HSIC) has been used as a tractable, efficient substitute in a variety of machine learning settings.


The remainder of this paper is structured as follows. We review related work in Sec. 2. In Sec. 3, we discuss the standard setting of adversarial robustness and HSIC. In Sec. 4, we provide a theoretical justification that HBaR reduces the sensitivity of the classifier to adversarial examples. Sec. 5 includes our experiments; we conclude in Sec. 6.

2 Related Work

Adversarial Attacks. Adversarial attacks often add a constrained perturbation to natural inputs with the goal of maximizing classification loss. Szegedy et al. [25] learn a perturbation via box-constrained L-BFGS that misleads the classifier but minimally distort the input. FGSM, proposed by Goodfellow et al. [8], is a one step adversarial attack perturbing the input based on the sign of the gradient of the loss. PGD [13, 17] generates adversarial examples through multi-step projected gradient descent optimization. DeepFool [18] is an iterative attack strategy, which perturbs the input towards the direction of the decision boundaries. CW [3] applies a rectifier function regularizer to generate adversarial examples near the original input. AutoAttack (AA) [5] is an ensemble of parameter-free attacks, that also deals with common issues like gradient masking [19] and fixed step sizes [17].

Adversarial Robustness. A common approach to obtaining robust models is adversarial training, i.e., training models over adversarial examples generated via the aforementioned attacks. For example, Madry et al. [17] show that training with adversarial examples generated by PGD achieves good robustness under different attacks. DeepDefense [34] penalizes the norm of adversarial perturbations. TRADES [36] minimizes the difference between the predictions of natural and adversarial...
examples to get a smooth decision boundary. MART \cite{30} pays more attention to adversarial examples from misclassified natural examples and adds a KL-divergence term between natural and adversarial samples to the cross-entropy loss. We show that our proposed method HBaR can be combined with several such state-of-the-art defense methods and boost their performance.

### 3.1 Adversarial Robustness

In standard \(k\)-ary classification, we are given a dataset \(D = \{(x_i, y_i)\}_{i=1}^n\), where \(x_i \in \mathbb{R}^d\), \(y_i \in \{0, 1\}^k\) are i.i.d. samples drawn from joint distribution \(P_{XY}\). A learner trains a neural network \(h_\theta : \mathbb{R}^d \rightarrow \mathbb{R}^k\) parameterized by weights \(\theta \in \mathbb{R}^{d\theta}\) to predict \(Y\) from \(X\) by minimizing

\[
\mathcal{L}(\theta) = \mathbb{E}_{XY}[\ell(\theta(X), Y)] \approx \frac{1}{n} \sum_{i=1}^n \ell(\theta(x_i), y_i),
\]

(1)

where \(\ell : \mathbb{R}^k \times \mathbb{R}^k \rightarrow \mathbb{R}\) is a loss function, e.g., cross-entropy. We aim to find a model \(h_\theta\) that has high prediction accuracy but is also adversarially robust: the model should maintain high prediction accuracy against a constrained adversary, that can perturb input samples in a restricted fashion. Formally, prior to submitting a sample \(x \in \mathbb{R}^d\) to the classifier, an adversary may perturb \(x\) by an arbitrary \(\delta \in S_r\), where \(S_r \subseteq \mathbb{R}^d\) is the \(\ell_\infty\)-ball of radius \(r\), i.e.,

\[
S_r = B(0, r) = \{\delta \in \mathbb{R}^d : \|\delta\|_\infty \leq r\}.
\]

(2)

The adversarial robustness \cite{17} of a model \(h_\theta\) is measured by the expected loss attained by such adversarial examples, i.e.,

\[
\mathcal{L}_r(\theta) = \mathbb{E}_{XY} \left[\max_{\delta \in S_r} \ell(h_\theta(X + \delta), Y)\right] \approx \frac{1}{n} \sum_{i=1}^n \max_{\delta \in S_r} \ell(h_\theta(x_i + \delta), y_i).
\]

(3)

An adversarially robust neural network \(h_\theta\) can be obtained via adversarial training, i.e., by minimizing the adversarial robustness loss in (3) empirically over the training set \(D\). In practice, this amounts to training via stochastic gradient descent (SGD) over adversarial examples \(x_i + \delta\) (see, e.g., \cite{17}). In each epoch, \(\delta\) is generated on a per sample basis via an inner optimization over \(S_r\), e.g., via projected gradient descent (PGD) on \(-\mathcal{L}\).

### 3 Background

#### 3.1 Adversarial Robustness

In standard \(k\)-ary classification, we are given a dataset \(D = \{(x_i, y_i)\}_{i=1}^n\), where \(x_i \in \mathbb{R}^d\), \(y_i \in \{0, 1\}^k\) are i.i.d. samples drawn from joint distribution \(P_{XY}\). A learner trains a neural network \(h_\theta : \mathbb{R}^d \rightarrow \mathbb{R}^k\) parameterized by weights \(\theta \in \mathbb{R}^{d\theta}\) to predict \(Y\) from \(X\) by minimizing

\[
\mathcal{L}(\theta) = \mathbb{E}_{XY}[\ell(\theta(X), Y)] \approx \frac{1}{n} \sum_{i=1}^n \ell(\theta(x_i), y_i),
\]

(1)

where \(\ell : \mathbb{R}^k \times \mathbb{R}^k \rightarrow \mathbb{R}\) is a loss function, e.g., cross-entropy. We aim to find a model \(h_\theta\) that has high prediction accuracy but is also adversarially robust: the model should maintain high prediction accuracy against a constrained adversary, that can perturb input samples in a restricted fashion. Formally, prior to submitting a sample \(x \in \mathbb{R}^d\) to the classifier, an adversary may perturb \(x\) by an arbitrary \(\delta \in S_r\), where \(S_r \subseteq \mathbb{R}^d\) is the \(\ell_\infty\)-ball of radius \(r\), i.e.,

\[
S_r = B(0, r) = \{\delta \in \mathbb{R}^d : \|\delta\|_\infty \leq r\}.
\]

(2)

The adversarial robustness \cite{17} of a model \(h_\theta\) is measured by the expected loss attained by such adversarial examples, i.e.,

\[
\mathcal{L}_r(\theta) = \mathbb{E}_{XY} \left[\max_{\delta \in S_r} \ell(h_\theta(X + \delta), Y)\right] \approx \frac{1}{n} \sum_{i=1}^n \max_{\delta \in S_r} \ell(h_\theta(x_i + \delta), y_i).
\]

(3)

An adversarially robust neural network \(h_\theta\) can be obtained via adversarial training, i.e., by minimizing the adversarial robustness loss in (3) empirically over the training set \(D\). In practice, this amounts to training via stochastic gradient descent (SGD) over adversarial examples \(x_i + \delta\) (see, e.g., \cite{17}). In each epoch, \(\delta\) is generated on a per sample basis via an inner optimization over \(S_r\), e.g., via projected gradient descent (PGD) on \(-\mathcal{L}\).
3.2 Hilbert-Schmidt Independence Criterion (HSIC)

The Hilbert-Schmidt Independence Criterion (HSIC) is a statistical dependency measure introduced by Gretton et al. \[10\]. HSIC is the Hilbert-Schmidt norm of the cross-covariance operator between the distributions in Reproducing Kernel Hilbert Space (RKHS). Similar to Mutual Information (MI), HSIC captures non-linear dependencies between random variables.

\[
\text{HSIC}(X, Y) = \mathbb{E}_{X,Y} \left[ k_X(X, X') k_Y(Y, Y') \right] + \mathbb{E}_{X,Y} \left[ k_X(X, X') \right] \mathbb{E}_{Y,Y'} \left[ k_Y(Y, Y') \right] - 2 \mathbb{E}_{X,Y} \left[ \mathbb{E}_{X'} \left[ k_X(X, X') \right] \mathbb{E}_{Y'} \left[ k_Y(Y, Y') \right] \right] , \tag{4}
\]

where \(X', Y'\) are independent copies of \(X, Y\), respectively, and \(k_X, k_Y\) are kernels.

In practice, we often approximate HSIC empirically. Given \(n\) i.i.d. samples \(\{(x_i, y_i)\}_{i=1}^n\) drawn from \(P_{X,Y}\), we estimate HSIC via:

\[
\hat{\text{HSIC}}(X, Y) = (n - 1)^{-2} \text{tr} \left( K_X H K_Y H \right) , \tag{5}
\]

where \(K_X\) and \(K_Y\) are kernel matrices with entries \(K_{X,ij} = k_X(x_i, x_j)\) and \(K_{Y,ij} = k_Y(y_i, y_j)\), respectively, and \(H = I - \frac{1}{n} 1 1^\top\) is a centering matrix.

4 Methodology

In this section, we present our method, HSIC bottleneck as regularizer (HBaR) as a means to enhance a classifier’s robustness. The effect of HBaR for adversarial robustness is illustrated in Figure 1; the HSIC bottleneck penalty reduces the sensitivity of the classifier to adversarial examples. We provide a theoretical justification for this below, in Theorems 1 and 2, but also validate the efficacy of the HSIC bottleneck extensively with experiments in Section 5.

4.1 HSIC Bottleneck as Regularizer for Robustness

Given a feedforward neural network \(h_\theta : \mathbb{R}^d \rightarrow \mathbb{R}^k\) parameterized by \(\theta\) with \(M\) layers, and an input r.v. \(X\), we denote by \(Z_i \in \mathbb{R}^{d_{Z_i}}, j \in \{1, \ldots, M\}\), the output of the \(j\)-th layer under input \(X\) (i.e., the \(j\)-th latent representation). We define our HBaR learning objective as follows:

\[
\hat{\mathcal{L}}(\theta) = \mathcal{L}(\theta) + \lambda_x \sum_{j=1}^M \text{HSIC}(X, Z_j) - \lambda_y \sum_{j=1}^M \text{HSIC}(Y, Z_j) , \tag{6}
\]

where \(\mathcal{L}\) is the standard loss given by Eq. (1) and \(\lambda_x, \lambda_y \in \mathbb{R}_+\) are balancing hyperparameters.

Together, the second and third terms in Eq. (6) form the HSIC bottleneck penalty. As HSIC measures dependence between two random variables, minimizing \(\text{HSIC}(X, Z_i)\) corresponds to removing redundant or noisy information contained in \(X\). Hence, this term also naturally reduces the influence of an adversarial attack, i.e., a perturbation added on the input data. This is intuitive, but we also provide theoretical justification in the next subsection. Meanwhile, maximizing \(\text{HSIC}(Y, Z_i)\) encourages this lack of sensitivity to the input to happen while retaining the discriminative nature of the classifier, captured by dependence to useful information w.r.t. the output label \(Y\). Note that minimizing \(\text{HSIC}(X, Z_i)\) alone would also lead to the loss of useful information, so it is necessary to keep the \(\text{HSIC}(Y, Z_i)\) term to make sure \(Z_i\) is informative enough of \(Y\).

The overall algorithm is described in Alg. 1. In practice, we perform Stochastic Gradient Descent (SGD) over \(\hat{\mathcal{L}}\); both \(\mathcal{L}\) and HSIC can be evaluated empirically over batches. For the latter, we use the estimator (5), restricted over the current batch. As we have \(m\) samples in a mini-batch, the complexity of calculating the empirical HSIC (5) is \(O(m^2d_\mathcal{Z})\) for a single layer, where \(d_\mathcal{Z} = \max_j d_{Z_j}\). Thus, the overall complexity for (6) is \(O(Mm^2d_\mathcal{Z})\). This computation is highly parallelizable, thus, the additional computation time of HBaR is small when compared to training a neural network via cross-entropy only.
maps the intermediate layer to the final layer. Then, we assume that both $k_{x}$, $k_{y}$, $k_{z}$, a neural network $h_{\theta}$ parameterized by $\theta$, mini-batch size $m$, learning rate $\alpha$.

**Output:** parameter of classifier $\theta$

**While** $\theta$ has not converged **do**

- Sample a mini-batch of size $m$ from input samples.
- Forward Propagation: calculate $z_{i}$ and $h_{\theta}(x)$. Compute kernel matrices for $X$, $Y$ and $Z_{i}$ using $k_{x}$, $k_{y}$, $k_{z}$ respectively inside mini-batch.
- Compute $\hat{L}(\theta)$ via (6), where HSIC is evaluated empirically via (5).
- Backward Propagation: $\theta \leftarrow \theta - \alpha \nabla \hat{L}(\theta)$.

**end**

### 4.2 Combining HBaR with Adversarial Examples

HBaR can also be naturally applied in combination with adversarial training. For $r > 0$ the magnitude of the perturbations introduced in adversarial examples, one can optimize the following objective instead of $\hat{L}(\theta)$ in Eq. (6):

$$
\hat{L}_{r}(\theta) = L_{r}(\theta) + \lambda_{x} \sum_{j=1}^{M} \text{HSIC}(X, Z_{j}) - \lambda_{y} \sum_{j=1}^{M} \text{HSIC}(Y, Z_{j}),
$$

(7)

where $L_{r}$ is the adversarial loss given by Eq. (3). This can be used instead of $L$ in Alg. 1. Adversarial examples need to be used in the computation of the gradient of the loss $L_{r}$ in each minibatch; these need to be computed on a per sample basis, e.g., via PGD over $S_{r}$, at additional computational cost. Note that the natural samples $(x_{i}, y_{i})$ in a batch are used to compute the HSIC bottleneck regularizer.

The HBaR penalty can similarly be combined with other adversarial learning methods and/or used with different means for selecting adversarial examples, other than PGD. We illustrate this in Section 5, where we combine HBaR with state-of-the-art adversarial learning methods TRADES [36] and MART [30].

### 4.3 HBaR Robustness Guarantees

We provide here a formal justification for the use of HBaR to enhance robustness: we prove that regularization terms $\text{HSIC}(X, Z_{j})$, $j = 1, \ldots, M$ lead to classifiers which are less sensitive to input perturbations. For simplicity, we focus on the case where $k = 1$ (i.e., binary classification). Let $Z \in \mathbb{R}^{d_{z}}$ be the latent representation at some arbitrary intermediate layer of the network. That is, $Z = Z_{j}$, for some $j \in \{1, \ldots, M\}$; we omit the subscript $j$ to further reduce notation clutter. Then $h_{\theta} = (g \circ f)$, where $f : \mathbb{R}^{d_{x}} \rightarrow \mathbb{R}^{d_{z}}$ maps the inputs to this intermediate layer, and $g : \mathbb{R}^{d_{z}} \rightarrow \mathbb{R}$ maps the intermediate layer to the final layer. Then, $Z = f(X)$ and $g(Z) = h_{\theta}(X) \in \mathbb{R}$ are the latent and final outputs, respectively. Recall that, in HBaR, $\text{HSIC}(X, Z)$ is associated with kernels $k_{x}$, $k_{z}$. We make the following technical assumptions:

**Assumption 1.** Let $X \subseteq \mathbb{R}^{d_{x}}$, $Z \subseteq \mathbb{R}^{d_{z}}$ be the supports of random variables $X$, $Z$, respectively. We assume that both $h_{\theta}$ and $g$ are continuous and bounded functions in $X$, $Z$, respectively, i.e.:

$$
h_{\theta} \in C(X), g \in C(Z).
$$

(8)

Moreover, we assume that all functions $h_{\theta}$ and $g$ we consider are uniformly bounded, i.e., there exist $0 < M_{X}, M_{Z} < \infty$ such that:

$$
M_{X} = \max_{h_{\theta} \in C(X)} \|h_{\theta}\|_{\infty} \quad \text{and} \quad M_{Z} = \max_{g \in C(Z)} \|g\|_{\infty}.
$$

(9)

The continuity stated in Assumption 1 is natural, if all activation functions are continuous. Boundedness follows if, e.g., $X$, $Z$ are closed and bounded (i.e., compact), or if activation functions are bounded (e.g., softmax, sigmoid, etc.).
Assumption 2. We assume kernels $k_X, k_Z$ are universal with respect to functions $h_\theta$ and $g$ that satisfy Assumption 2, i.e., if $\mathcal{F}$ and $\mathcal{G}$ are the induced RKHSs for kernels $k_X$ and $k_Z$, respectively, then for any $h_\theta, g$ that satisfy Assumption 2, and any $\varepsilon > 0$ there exist functions $h' \in \mathcal{F}$ and $g' \in \mathcal{G}$ such that $\|h_\theta - h'\|_\infty \leq \varepsilon$ and $\|g - g'\|_\infty \leq \varepsilon$. Moreover, functions in $\mathcal{F}$ and $\mathcal{G}$ are uniformly bounded, i.e., there exist $0 < M_F, M_G < \infty$ such that for all $h' \in \mathcal{F}$ and all $g' \in \mathcal{G}$:
\[
M_F = \max_{f' \in \mathcal{F}} \|f'\|_\infty \quad \text{and} \quad M_G = \max_{g' \in \mathcal{G}} \|g'\|_\infty.
\]

We note that several kernels used in practice are universal, including, e.g., the Gaussian and Laplace kernels. Moreover, given that functions that satisfy Assumption 1 are uniformly bounded by 9, such kernels can indeed remain universal while satisfying (10) via an appropriate rescaling.

Our first result shows that $\text{HSIC}(X, Z)$ at any intermediate layer $Z$ bounds the output variance:

Theorem 1. Under Assumptions 1 and 2 we have:
\[
\text{HSIC}(X, Z) \geq \frac{M_F M_G}{M_X M_Z} \sup_{\theta} \text{Var}(h_\theta(X)).
\]

The proof of Theorem 1 is in Appendix B in the supplement. We use a result by Greenfeld and Shalit 9 that links $\text{HSIC}(X, Z)$ to the supremum of the covariance of bounded continuous functionals over $X$ and $Z$. Theorem 1 indicates that the regularizer $\text{HSIC}(X, Z)$ at any intermediate layer naturally suppresses the variability of the output, i.e., increases the classifier prediction $h_\theta(X)$. To see this, observe that by Chebyshev’s inequality 20 the distribution of $h_\theta(X)$ concentrates around its mean when $\text{Var}(h_\theta(X))$ approaches 0. As a result, bounding $\text{HSIC}(X, Z)$ inherently also bounds the (global) variability of the classifier (across all parameters $\theta$). This observation motivates us to also maximize $\text{HSIC}(Y, Z)$ to recover essential information useful for classification: if we want to achieve good adversarial robustness as well as good predictive accuracy, we have to strike a balance between $\text{HSIC}(X, Z)$ and $\text{HSIC}(Y, Z)$. This perfectly aligns with the intuition behind the information bottleneck 28 and the well-known accuracy-robustness trade-off 17, 36, 29, 21. We also confirm this experimentally: we observe that both additional terms (the standard loss and $\text{HSIC}(Y, Z)$) are necessary for ensuring good prediction performance in practice (see Table 3).

Most importantly, by further assuming that features are normal, we can show that $\text{HSIC}$ bounds the power of an arbitrary adversary, as defined in Eq. (3):

Theorem 2. Assume that $X \sim \mathcal{N}(0, \sigma^2 I)$. Then, under Assumptions 1 and 2 we have:
\[
\frac{r \sqrt{-2 \log \alpha}}{\sigma M_F M_G} \text{HSIC}(X, Z) + o(r) \geq \mathbb{E}[||h_\theta(X + \delta) - h_\theta(X)||], \quad \text{for all } \delta \in \mathcal{S}_r.
\]

The proof of Theorem 2 can also be found in Appendix C in the supplement. We again use the result by Greenfeld and Shalit 9 along with Stein’s Lemma 15, that relates covariances of Gaussian r.v.s and their functions to expected gradients. In particular, we apply Stein’s Lemma to the bounded functionals considered by Greenfeld and Shalit by using a truncation argument. Theorem 2 implies that $\text{HSIC}(X, Z)$ indeed bounds the output perturbation produced by an arbitrary adversary: suppressing $\text{HSIC}$ sufficiently can ensure that the adversary cannot alter the output significantly, in expectation. In particular, if $\text{HSIC}(X, Z) = o\left(\frac{\sigma M_F M_G}{\sqrt{-2 \log \alpha} d_X M_Z}\right)$, then $\lim_{r \to 0} \sup_{\delta \in \mathcal{S}_r} \mathbb{E}||h_\theta(X + \delta) - h_\theta(X)||/r = 0$, i.e., the output is almost constant under small input perturbations.

5 Experiments

5.1 Experimental Setting

We experiment with three standard datasets, MNIST 14, CIFAR-10 12 and CIFAR-100 12. We use a 4-layer LeNet 17 for MNIST, ResNet-18 11 and WideResNet-28-10 35 for CIFAR-10,
Table 1: Natural test accuracy (in %), adversarial robustness (in %) on FGSM, PGD, CW, and AA attacked test examples on MNIST and CIFAR-100 of [row i, iii, v] adversarial learning baselines and [row ii, iv, vi] combining HBaR with each correspondingly. Each result is the average of five runs.

<table>
<thead>
<tr>
<th>Methods</th>
<th>MNIST by LeNet</th>
<th>CIFAR-100 by WideResNet-28-10</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Natural FGSM</td>
<td>Natural FGSM</td>
</tr>
<tr>
<td></td>
<td>PGD</td>
<td>PGD</td>
</tr>
<tr>
<td></td>
<td>98.40</td>
<td>59.91</td>
</tr>
<tr>
<td></td>
<td>94.44</td>
<td>29.85</td>
</tr>
<tr>
<td></td>
<td>89.63</td>
<td>26.05</td>
</tr>
<tr>
<td></td>
<td>93.27</td>
<td>25.38</td>
</tr>
<tr>
<td></td>
<td>89.03</td>
<td>22.28</td>
</tr>
<tr>
<td></td>
<td>91.50</td>
<td>20.91</td>
</tr>
<tr>
<td></td>
<td>91.57</td>
<td>21.61</td>
</tr>
<tr>
<td>HBaR + PGD</td>
<td>98.66</td>
<td>63.84</td>
</tr>
<tr>
<td></td>
<td>96.02</td>
<td>31.59</td>
</tr>
<tr>
<td></td>
<td>94.34</td>
<td>27.90</td>
</tr>
<tr>
<td></td>
<td>91.20</td>
<td>27.21</td>
</tr>
<tr>
<td></td>
<td>86.62</td>
<td>23.23</td>
</tr>
<tr>
<td>TRADES</td>
<td>97.64</td>
<td>60.29</td>
</tr>
<tr>
<td>HBaR + TRADES</td>
<td>95.23</td>
<td>34.57</td>
</tr>
<tr>
<td></td>
<td>93.49</td>
<td>31.96</td>
</tr>
<tr>
<td></td>
<td>93.47</td>
<td>31.57</td>
</tr>
<tr>
<td></td>
<td>89.99</td>
<td>28.72</td>
</tr>
<tr>
<td>MART</td>
<td>98.29</td>
<td>58.93</td>
</tr>
<tr>
<td>HBaR + MART</td>
<td>96.09</td>
<td>33.49</td>
</tr>
<tr>
<td></td>
<td>96.08</td>
<td>30.72</td>
</tr>
<tr>
<td></td>
<td>94.64</td>
<td>30.16</td>
</tr>
<tr>
<td></td>
<td>94.62</td>
<td>28.89</td>
</tr>
<tr>
<td></td>
<td>89.99</td>
<td>25.21</td>
</tr>
</tbody>
</table>

Table 2: Natural test accuracy (in %), adversarial robustness (in %) on FGSM, PGD, CW, and AA attacked test examples on CIFAR-10 by ResNet-18 and WideResNet-28-10 of [row i, iii, v] adversarial learning baselines and [row ii, iv, vi] combining HBaR with each correspondingly. Each result is the average of five runs.

<table>
<thead>
<tr>
<th>Methods</th>
<th>CIFAR-10 by ResNet-18</th>
<th>CIFAR-10 by WideResNet-28-10</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Natural FGSM</td>
<td>Natural FGSM</td>
</tr>
<tr>
<td></td>
<td>PGD</td>
<td>PGD</td>
</tr>
<tr>
<td></td>
<td>84.71</td>
<td>86.63</td>
</tr>
<tr>
<td></td>
<td>55.95</td>
<td>58.53</td>
</tr>
<tr>
<td></td>
<td>49.37</td>
<td>52.21</td>
</tr>
<tr>
<td></td>
<td>47.54</td>
<td>50.59</td>
</tr>
<tr>
<td></td>
<td>41.17</td>
<td>49.32</td>
</tr>
<tr>
<td></td>
<td>43.42</td>
<td>47.25</td>
</tr>
<tr>
<td>HBaR + PGD</td>
<td>85.73</td>
<td>87.91</td>
</tr>
<tr>
<td></td>
<td>57.13</td>
<td>59.69</td>
</tr>
<tr>
<td></td>
<td>49.63</td>
<td>52.72</td>
</tr>
<tr>
<td></td>
<td>48.32</td>
<td>51.17</td>
</tr>
<tr>
<td></td>
<td>41.80</td>
<td>49.52</td>
</tr>
<tr>
<td></td>
<td>44.46</td>
<td>47.60</td>
</tr>
<tr>
<td>TRADES</td>
<td>84.07</td>
<td>85.66</td>
</tr>
<tr>
<td>HBaR + TRADES</td>
<td>85.10</td>
<td>85.61</td>
</tr>
<tr>
<td></td>
<td>58.63</td>
<td>61.55</td>
</tr>
<tr>
<td></td>
<td>53.21</td>
<td>56.62</td>
</tr>
<tr>
<td></td>
<td>52.36</td>
<td>55.67</td>
</tr>
<tr>
<td></td>
<td>50.07</td>
<td>54.02</td>
</tr>
<tr>
<td></td>
<td>49.38</td>
<td>52.71</td>
</tr>
<tr>
<td>MART</td>
<td>82.15</td>
<td>85.94</td>
</tr>
<tr>
<td>HBaR + MART</td>
<td>82.44</td>
<td>85.52</td>
</tr>
<tr>
<td></td>
<td>59.85</td>
<td>60.54</td>
</tr>
<tr>
<td></td>
<td>54.73</td>
<td>53.42</td>
</tr>
<tr>
<td></td>
<td>53.67</td>
<td>51.81</td>
</tr>
<tr>
<td></td>
<td>50.12</td>
<td>49.32</td>
</tr>
<tr>
<td></td>
<td>47.97</td>
<td>46.99</td>
</tr>
</tbody>
</table>

and WideResNet-28-10 [35] for CIFAR-100. We use cross-entropy as loss $L(\theta)$. Licensing information for all existing assets can be found in Appendix D in the supplement.

Algorithms. We compare HBaR to the following non-adversarial learning algorithms: Cross-Entropy (CE), Stage-Wise HSIC Bottleneck (SWHB) [16], XIC [9], and Variational Information Bottleneck (VIB) [1]. We also incorporate HBaR to several adversarial learning algorithms, as described in Section 4.2, and compare against the original methods, without the HBaR penalty. The adversarial methods we use are: Projected Gradient Descent (PGD) [17], TRADES [36], and MART [30]. Further details and parameters can be found in Appendix E in the supplement.

Performance Metrics. For all methods, we evaluate the obtained model $h_\theta$ via the following metrics: (a) Natural (i.e., clean test data) accuracy, and adversarial robustness via test accuracy under (b) FGSM, the fast gradient sign attack [8], (c) PGD$, the PGD attack with $m$ steps used for the internal PGD optimization [17], (d) CW, the CW-loss within the PGD framework [3], and (e) AA, AutoAttack [5]. All five metrics are reported in percent (%) accuracy. Following prior literature, we set step size to 0.01 and radius $r = 0.3$ for MNIST, and step size as $2/255$ and $r = 8/255$ for CIFAR-10 and CIFAR-100. All attacks happen during the test phase and have full access to model parameters (i.e., are white-box attacks). All experiments are carried out on a Tesla V100 GPU with 32 GB memory and 5120 cores.

5.2 Results

Combining HBaR with Adversarial Examples. We show how HBaR can be used to improve robustness when used as a regularizer, as described in Section 4.2 along with state-of-the-art adversarial learning methods. We run each experiment by five times and report the mean natural test accuracy and adversarial robustness of all models on MNIST, CIFAR-10, and CIFAR-100 datasets by four architectures in Table[1] and Table[2]. Combined with all adversarial training baselines, HBaR consistently improves adversarial robustness against all types of attacks on all datasets. The resulting improvements are larger than 2 standard deviations (that range between 0.05-0.2) in most cases; we report the results with standard deviations in Appendix C in the supplement. Although natural accuracy is generally restricted by the trade-off between robustness and accuracy [36], we observe that incorporating HBaR comes with an actual improvement over natural accuracy in most cases.
Figure 2: CIFAR-10 by ResNet-18: Adversarial robustness of IB-based baselines and proposed HBaR under (a) PGD attacks, (b) CW attack by various of constant ε, and (c) AA using different radius. Interestingly, while achieving the highest adversarial robustness under almost all cases, HBaR achieves natural accuracy (95.27%) comparable to CE (95.32%) which is much higher than VIB (92.35%), XIC (92.93%) and SWHB (59.18%).

Figure 3: Visualization of the HBaR quantities (a) HSIC(X, Z_M), (b) HSIC(Y, Z_M), (c) natural test accuracy, and (d) adversarial robustness against PGD attack (PGD^0 and PGD^20) on MNIST and CIFAR-10, respectively) as a function of training epochs, on MNIST by LeNet (top) and CIFAR-10 by ResNet (bottom). Different colored lines correspond to CE, HBaR-high (HBaR with high weights λ), and HBaR-low (HBaR method small weights λ). HBaR-low parameters are selected so that the values of the loss ℒ and each of the HSIC terms are close after the first epoch.

Adversarial Robustness Analysis without Adversarial Training. Next, we show that HBaR can achieve modest robustness even without adversarial examples during training. We evaluate the robustness of HBaR on CIFAR-10 by ResNet-18 against various adversarial attacks, and compare HBaR with other information bottleneck penalties without adversarial training in Figure 2. Specifically, we compare the robustness of HBaR with other IB-based methods under various attacks and hyperparameters. Our proposed HBaR achieves the best overall robustness against all three types of attacks while retaining competitive natural test accuracy. Interestingly, HBaR achieves natural accuracy (95.27%) comparable to CE (95.32%) which is much higher than VIB (92.35%), XIC (92.93%) and SWHB (59.18%). We observe SWHB underperforms HBaR on CIFAR-10 for both natural accuracy and robustness. One possible explanation may be that when the model is deep, minimizing HSIC without backpropagation, as in SWHB, does not suffice to transmit the learned information across layers. Compared to SWHB, HBaR backpropagates over the HSIC objective through each intermediate layer and computes gradients only once in each batch, improving accuracy and robustness while reducing computational cost significantly.

Synergy between HSIC Terms. Focusing on Z_M, the last latent layer, Figure 2 shows the evolution per epoch of: (a) HSIC(X, Z_M), (b) HSIC(Y, Z_M), (c) natural accuracy (in %), and (d) adversarial robustness (in %) under PGD attack on MNIST and CIFAR-10. Different lines correspond to CE, HBaR-high (HBaR with high weights λ), and HBaR-low (HBaR method small weights λ). HBaR-low parameters are selected so that the values of the loss ℒ and each of the HSIC terms are close after the first epoch. Figure 2(c) illustrates that all three settings achieve good natural accuracy on both datasets. However, in Figure 2(d), only HBaR-high, that puts sufficient weight on HSIC terms, attains relatively high adversarial robustness. In Figure 2(a), we see that CE leads to high HSIC(X, Z_M) for the shallow LeNet, but low in the (much deeper) ResNet-18, even lower than HBaR-low. Moreover, we also see that the best performer in terms of adversarial robustness, HBaR-high, lies in between the other two w.r.t. HSIC(X, Z_M). Both of these observations indi-
Figure 4: HSIC plane dynamics versus adversarial robustness. The x-axis plots HSIC between the last intermediate layer $Z_M$ and the input $X$, while the y-axis plots HSIC between $Z_M$ and the output $Y$. The color scale indicates adversarial robustness against PGD attack (PGD\textsuperscript{40} and PGD\textsuperscript{20} on MNIST and CIFAR-10, respectively). The arrows indicate dynamic direction w.r.t. training epochs. Each marker in the figures represents a different setting: dots, stars, and triangles represent CE-only, HBaR-high, and HBaR-low, respectively, compatible with the definition in Figure 3.

Figure 3(b) confirms this, as HBaR-high achieves relatively high HSIC($Y,Z_M$) on both datasets. Figure 4 provides another perspective of the same experiments via the learning dynamics on the HSIC plane. We again observe that the best performer in terms of robustness HBaR-high lies in between the other two methods, crucially attaining a much higher HSIC($Y,Z_M$) than HBaR-low. Moreover, for both HBaR methods, we clearly observe the two distinct optimization phases first observed by Shwartz-Ziv and Tishby [23] in the context of the mutual information bottleneck: the fast empirical risk minimization phase, where the neural network tries to learn a meaningful representation by increasing HSIC($Y,Z_M$) regardless of information redundancy (HSIC($X,Z_M$) increasing), and the representation compression phase, where the neural network turns its focus onto compressing the latent representation by minimizing HSIC($X,Z_M$), while maintaining highly label-related information. Interestingly, the HBaR penalty produces the two-phase behavior even though our networks use ReLU activation functions; Shwartz et al. [23] only observed these two optimization phases on neural networks with tanh activation functions, a phenomenon further confirmed by Saxe et al. [22].

**Ablation Study.** Motivated by the above observations, we turn our attention to how the three terms in the loss function in Eq. (6) affect HBaR. As illustrated in Table 3, removing any part leads to either a significant natural accuracy or robustness degradation. Specifically, using $L(\theta)$ only (row [i]) lacks adversarial robustness; removing $L(\theta)$ (row [ii]) or the penalty on $Y$ (row [iii]) degrades natural accuracy significantly (a similar result was also observed in [2]); finally, removing the penalty on $X$ improves the natural accuracy while degrading adversarial robustness. The three terms combined together by proper hyperparameters $\lambda_x$ and $\lambda_y$ (row [v]) achieve both high natural accuracy and adversarial robustness. We provide a comprehensive ablation study on the sensitivity of $\lambda_x$ and $\lambda_y$ and draw conclusions in Appendix F in the supplement (Tables 7 and 8).
Table 3: Ablation study on HBaR. Rows [i-iv] indicate the effect of removing each component of the learning objective defined in Eq.(6) (row [v]). We evaluate each objective over $\text{HSIC}(X, Z_M)$, $\text{HSIC}(Y, Z_M)$, natural test accuracy (in %), and adversarial robustness (in %) against PGD$^{40}$ and PGD$^{20}$ on MNIST and CIFAR-10 respectively. We set $\lambda_x$ as 1 and 0.006, $\lambda_y$ as 50 and 0.05, for MNIST and CIFAR-10 respectively.

<table>
<thead>
<tr>
<th>Rows</th>
<th>Objectives</th>
<th>MNIST by LeNet</th>
<th>CIFAR-10 by ResNet-18</th>
</tr>
</thead>
<tbody>
<tr>
<td>[i]</td>
<td>$\mathcal{L}(\theta)$</td>
<td>45.29 8.73</td>
<td>99.23 0.00</td>
</tr>
<tr>
<td>[ii]</td>
<td>$\lambda_x \sum_j \text{HSIC}(X, Z_j) - \lambda_y \sum_j \text{HSIC}(Y, Z_j)$</td>
<td>16.45 8.65</td>
<td>30.08 9.47</td>
</tr>
<tr>
<td>[iii]</td>
<td>$\mathcal{L}(\theta) + \lambda_x \sum_j \text{HSIC}(X, Z_j)$</td>
<td>0.00 0.00</td>
<td>11.38 10.00</td>
</tr>
<tr>
<td>[iv]</td>
<td>$\mathcal{L}(\theta) - \lambda_y \sum_j \text{HSIC}(Y, Z_j)$</td>
<td>56.38 9.00</td>
<td>99.33 0.00</td>
</tr>
<tr>
<td>[v]</td>
<td>$\mathcal{L}(\theta) + \lambda_x \sum_j \text{HSIC}(X, Z_j) - \lambda_y \sum_j \text{HSIC}(Y, Z_j)$</td>
<td>15.68 8.89</td>
<td>98.90 8.33</td>
</tr>
</tbody>
</table>

6 Conclusions

We investigate the HSIC bottleneck as regularizer (HBaR) as a means to enhance adversarial robustness. We theoretically prove that HBaR suppresses the sensitivity of the classifier to adversarial examples while retaining its discriminative nature. One limitation of our method is that the robustness gain is modest when training with only natural examples. Moreover, a possible negative societal impact is overconfidence in adversarial robustness: over-confidence in the adversarially-robust models produced by HBaR as well as other defense methods may lead to overlooking their potential failure on newly-invented attack methods; this should be taken into account in safety-critical applications like healthcare [6] or security [26]. We extend the discussion on the limitations and potential negative societal impacts of our work in Appendix H and I, respectively, in the supplement.

7 Acknowledgements

The authors gratefully acknowledge support by the National Science Foundation under grants CCF-1937500 and CNS-2112471, and the National Institutes of Health under grant NHLBI U01HL089856.
References


