2023 53rd Annual IEEE/IFIP International Conference on Dependable Systems and Networks Workshops (DSN-W) | 979-8-3503-2543-0/23/$31.00 ©2023 IEEE | DOI: 10.1109/DSN-W58399.2023.00041

IB-RAR: Information Bottleneck as Regularizer for Adversarial Robustness

Xiaoyun Xu
Digital Security Group
Radboud University
Nijmegen, The Netherlands
xiaoyun.xu@ru.nl

Guilherme Perin
Leiden Institute of Advanced Computer Science
Leiden University
Leiden, The Netherlands
guilhermeperin7@gmail.com

Stjepan Picek
Digital Security Group
Radboud University
Nijmegen, The Netherlands
stjepan.picek@ru.nl

***Abstract*—This paper proposes a novel method, IB-RAR, which uses Information Bottleneck (IB) to strengthen adversarial robustness for both adversarial training and non-adversarial-trained methods. We first use the IB theory to build regularizers as learning objectives in the loss function. Then we filter out unnecessary features of intermediate representation according to their mutual information (MI) with labels, as the network trained with IB provides easily distinguishable MI for its features. Experimental results show that IB-RAR can be naturally combined with adversarial training and provides consistently better accuracy on new adversarial examples. The IB-RAR method improves the accuracy by an average of 2.66% against five adversarial attacks for ResNet-18, wide ResNet-28-10, and VGG-16, trained with three adversarial training benchmarks and the CIFAR-10, CIFAR-100, and Tiny ImageNet datasets. In addition, IB-RAR also provides good robustness for undefended methods, such as training with cross-entropy loss only. Finally, without adversarial training, the VGG-16 network trained using IB-RAR on the CIFAR-10 dataset reaches an accuracy of 35.86% against PGD examples, while using all layers reaches 25.61% accuracy.**

***Index Terms*—Deep Learning, Adversarial Attack, Adversarial Training, Information Bottleneck, Mutual Information**

I. Introduction

Deep learning networks are vulnerable to adversarial attacks [2], [22]. Neural network predictions can be easily fooled by subtle adversarial perturbations, while the input remains visually imperceptible to humans. Such perturbations can be generated by specific algorithms, such as Fast Gradient Sign Method (FGSM) [7], projected gradient descent (PGD) [17], and Carlini & Wagner (CW) [3]. The main goal of these algorithms is to find as small perturbations as possible that mislead the prediction model. This potential vulnerability raises concerns about the reliability of practical deep learning applications, especially in security-sensitive fields, such as vulnerability detection [4], drug discovery [15], and financial market predictions [6].

Previous works have proposed many possible causes for successful adversarial attacks. Goodfellow et al. [7] argued that the adversarial examples are generated by the excessive linearity behavior of DNNs in high-dimensional spaces. Ilyas et al. [10] have demonstrated that adversarial attacks can arise from features (can be well-generalized) instead of bugs (do not generalize due to effects of poor statistical concentration). The features may be robust or not robust. Non-robust features can be easily manipulated by imperceptible noise, while robust features will not. Still, the community needs to reach a consensus on the underlying reason for the prevalence of adversarial examples.

To further analyze the impact of adversarial examples, IB is used as a learning objective to improve adversarial robustness [1], [25]. The IB is supposed to find the optimal trade-off between compression of input X and prediction of Y by MI ($I(\cdot)$) [20]. IB provides both performance and adversarial robustness when embedded into the learning objective. Intuitively, this is because X is mapped to Y through intermediate representation T (outputs of hidden layers). Compression of X ($I(X,T)$) naturally removes the noise in X and makes it difficult to transfer small perturbations via the bottleneck. However, computing mutual information is difficult in practice, especially when dealing with high-dimensional data. To address this problem, Alemi et al. [1] proposed Variational Information Bottleneck (VIB). They used the internal representation of a certain intermediate layer as a stochastic encoding T of the input data X. They aimed to learn the most informative representation T about the target Y, measured by the mutual information between their corresponding encoding values. Their experiments also showed that VIB is robust to overfitting and adversarial attacks. Ma et al. [16] proposed the HSIC Bottleneck and replaced mutual information with Hilbert Schmidt Independence Criterion (HSIC). They used the HSIC bottleneck as a learning objective for every layer of the network, which is an alternative to the conventional CE loss and back-propagation. Wang et al. [25] proposed HBaR, which combined HSIC Bottleneck of all hidden layers and back-propagation to improve both adversarial robustness and accuracy of clean data.

These IB-related methods use one layer (VIB) or all layers (HBaR) by default to build IB in their learning objectives, but we find that the IB of each layer has a different impact on robustness (see Table III). Deeper layers usually provide more robustness with IB. The reason is that shallower layers usually generate representations with noise that are not informative enough to be distinguishable (see Figure 1). Therefore, MI computed from shallower layers is less meaningful than from deeper layers. We aim to compute mutual information (MI) between intermediate layers and their inputs or between inter-

979-8-3503-2543-0/23/$31.00 ©2023 IEEE
DOI 10.1109/DSN-W58399.2023.00041

mediate layers and their targets. Then the MI is embedded into the loss function according to IB. We summarize the following two questions about applying IB as a learning objective:

1) Which intermediate layers do we need to use? To address question 1), we propose using only robust layers to compute MI that is then used to apply IB in the loss function. We refer to robust layers as layers providing obviously higher accuracy than the network trained with only CE loss under PGD attack (since PGD provides good robustness against various attacks, as discussed later). To reduce the impact of adversarial training, we evaluate the performance of robust layers without adversarial training. This paper empirically shows that compared to training with cross-entropy (CE) only, each layer of the network provides different degrees of robustness when applying IB as a learning objective. Then, using robust layers for IB objective upgrades robustness to adversarial attacks.
2) Can the representation of the non-robust layers be further improved? To address question 2), we compute a mask to remove unnecessary feature channels of convolutional layers, as the outputs of non-robust layers are extracted by subsequent convolutional layers. When the network is trained with the loss function with IB and learns more informative features, some features are not relevant to the classification or target as they are not informative enough.

Our evaluation consists of two parts: (1) Combining our method with state-of-the-art adversarial training methods, e.g., PGD [17], TRADES [27], and MART [24]. Experimental results show that our method can improve the robustness of adversarial training methods. (2) Combining without adversarial training methods. Experimental results show that our method provides robustness compared to other IB-related techniques and plain CE. We also find that the robustness of VGG-16 mainly comes from the last convolutional block and the first two fully connected layers when using our method. When using these three layers to compute MI, the IB-RAR method provides higher accuracy on adversarial examples than using all layers. Ablation study results reveal which part of our method provides robustness and the connection among them. In addition, applying IB as a learning objective also accelerates training convergence according to experimental results. Our implementation is available at https://github.com/xiaoyunxxy/IB-RAR/.

Our main contributions are:

- We apply IB as a regularizer to improve robustness and natural accuracy (on clean data), and we remove unnecessary features based on the regularized network.
- We show that our method can be naturally embedded into state-of-the-art adversarial training methods. Our method improves accuracy on adversarial examples by an average of 2.66% against five adversarial attacks for ResNet-18, wide ResNet-28-10, and VGG-16 in Tables I and II.
- We show that our method can improve robustness for weaker methods, like plain stochastic gradient descent (SGD) trained with the CE loss function only. Without adversarial training, the VGG-16 network trained using our method reaches an accuracy of 35.86% against PGD examples, while using all layers reaches 25.61%.

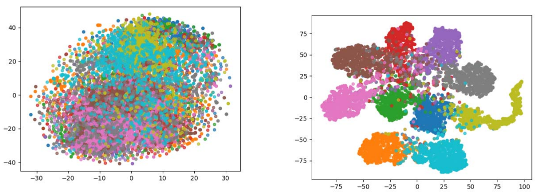


Fig. 1. t-SNE [23] visualization for the first convolutional block and the penultimate fully connected layer of CIFAR-10 with VGG-16. Different colors refer to different classes.

II. Methodology

We first propose using IB as a regularizer for the learning objective, which also helps the network to learn better generalization of training data [20]. Specifically, we embed mutual information from intermediate representations to inputs X and targets Y. However, the outputs of convolutional layers contain independent feature channels. Then, we find that IB-regularized feature channels generalize better than not using IB. Thus, we propose filtering out low correlation feature channels among well-generalized features according to their mutual information to their label. Note that the IB regularizer is the foundation of the filtering process. Figure 2 shows the structure of our method.

A. Threat Model

Adversary Goals. This work focuses on adversarial attacks on image classifiers. The adversary aims to create imperceptible perturbation for the input image, so the network misclassifies perturbed images. The experiments investigate untargeted individual adversarial attacks.

Adversary Knowledge. Our evaluation is conducted under white-box accessibility. The adversary has complete knowledge of the target network and its parameters. The adversary also has full access to the training data of the target network. The adversary knows the defense method, so IB-RAR is also evaluated by a specifically designed adaptive attack discussed in Appendix Section A-B.

Adversary Capabilities. The adversary can perturb the input to well-trained networks when the perturbation is imperceptible. Following previous research, *imperceptibility* is defined as the distance from the perturbed image to its original copy. The distance is formalized as ℓ_n-norm, i.e., $||\delta||_n \leq r$.

B. Mutual Information Loss

Problem Setup: We consider an L-layer neural network F_θ for classification in d_Y-dimensional space, and training dataset $D = \{(x_i, y_i)\}_{i=0}^{n-1}$ in d_X-dimensional space, where $x_i \in \mathbb{R}^{d_X}$ and $y_i \in \{0,1\}^{d_Y}$. The x_i refers to an example from training data, and we use X to indicate a batch of training data in this section. A network assigns to x_i one element in $\{0,1\}^{d_Y}$. The training aims to minimize the difference

Algorithm 1 Training with loss based on IB

Input: training data D, network F_θ with parameters θ, batch size m, learning rate a, loss $\mathcal{L}_{CE}$, optimizer SGD
Output: optimized weights θ
  while Maximum epoch not reached **do**
    Sample X, a batch of data from D
    Forward: Calculate $F_\theta(X)$ and $T = \{T_l | 0 \leq l < L\}$
    $T_{last} = T_{last} * mask$
    Calculate $\sum_{l=1}^{L} I(X, T_l)$ and $\sum_{l=1}^{L} I(Y, T_l)$
    Calculate loss as in Eq. (1)
    Backward: update θ by : $\theta \leftarrow \theta - a\nabla\mathcal{L}$
  end while

between predicted results and the ground truth, which is quantified by the standard CE loss: $\mathcal{L}_{CE}(\theta, F_\theta(x_i), y_i)$. T_l indicates the output of l_{th} layer to describe the intermediate representation of a network.

The IB is embedded as a learning objective by our novel loss function:

$$\min_{\theta} \ \mathcal{L} = \mathcal{L}_{CE} + \alpha \sum_{l=1}^{L} I(X, T_l) - \beta \sum_{l=1}^{L} I(Y, T_l). \qquad (1)$$

Specifically, the second term $\alpha \sum_l I(X, T_l)$ minimizes the relevance between inputs and intermediate features as the loss is supposed to decrease. Decreasing $I(X, T_l)$ compresses input to an efficient representation, which naturally removes noise and irrelevant information from X in T_l [20]. The term X refers to a batch of inputs at each iteration while training. The third term $\beta \sum_l I(Y, T_l)$ maximizes the relevance to the ground truth. The term Y refers to a batch of labels corresponding to X. All hidden layer outputs are embedded in the loss through summations. Because the compression measured with $I(X, T_l)$ also indicates a loss of useful information about Y (as this information is indiscriminate to all content in input X), $I(Y, T_l)$ becomes necessary to guarantee the accuracy on clean data. Algorithm 1 describes how to train a network with our proposed loss from Eq. (1). The $mask$ and T_{last} are discussed in Section II-C. As the computation of mutual information is difficult, we use HSIC [8] as an alternative plan for $I(\cdot)$.

Combination with adversarial training: In addition to training with clean data, the IB loss from Eq. (1) can also be easily combined with adversarial training as follows:

$$\begin{aligned} \mathcal{L}_{adv} &= \max_{\delta \in S} \mathcal{L}_{CE}(\theta, F_\theta(X + \delta), X) \\ \min_{\theta} \ & \mathcal{L}_{adv} + \alpha \sum_{l=1}^{L} I(X, T_l) - \beta \sum_{l=1}^{L} I(Y, T_l). \end{aligned} \qquad (2)$$

This way, it is easy to perform adversarial training by replacing the loss in Algorithm 1 with Eq. (2). Here, the adversarial perturbation is generated with the PGD algorithm, as it has been shown robust to various attacks [17].

Selection of Robust Layers: While other IB-related methods use one layer (such as VIB [1]) or all layers (such as HSIC Bottleneck [16] and HBaR [25]), we find that deploying

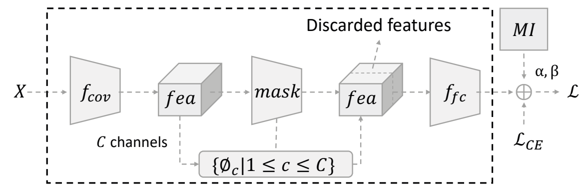


Fig. 2. The structure of IB-RAR. First, we use IB as a regularization combined with CE. Then, feature channels extracted from the convolutional layer are filtered by ϕ_c.

IB to different hidden layers will provide different robustness (see Table III). Using a part of hidden layers for IB can get higher adversarial robustness than all layers or a single layer. We refer to this part of layers as robust layers. To distinguish robust layers from others, we apply IB to each hidden layer and train an independent network for each layer. If a network shows higher accuracy on adversarial examples compared to the network trained with CE only, the corresponding layer is considered a robust layer.

C. Removing Unnecessary Features

After training the network with MI loss from Eq. (1), the network is supposed to learn a more informative generalization of training data. The output of convolutional layers will have a closer connection to their targets, quantified by MI in this paper. Specifically, feature channels containing more noise will be irrelevant to their label. The network provides the most informative representations after extracting all convolutional layers. Otherwise, those convolutional layers are not necessary for the network structure. Therefore, we evaluate feature channels given by the last convolutional layer according to their MI to the target Y. Still, there is another reason for choosing the last convolutional layer. According to results in [12], distilled non-robust features from the last convolutional layer significantly decrease the accuracy on clean and adversarial examples. It means that non-robust (unnecessary) features have a large negative effect on the last convolutional layer, which should be discarded.

Structurally, a network F_θ trained with our MI loss is given by the concatenation of L hidden layers outputs F_{θ_l}. The network's output is the output of the last layer in the network, i.e., F_{θ_L}. Each layer uses the output of the previous layer as input. We consider F_θ has C kernels to extract multiple feature channels at the last convolution layer:

$$F_{\theta_{last}}(x) = T_{last} = \{f_c | 1 \leq c \leq C\}.$$

Following, we compute a mask based on the MI values of each feature channel:

$$\begin{aligned} T_{last} &= T_{last} * mask \\ mask &= \{\phi_c | 1 \leq c \leq C\} \\ \phi_c &= \begin{cases} 1, & I(f_c, Y) \geq thr \\ 0, & otherwise. \end{cases} \end{aligned} \qquad (3)$$

Then the mask is used to filter the feature channels. Channels with MI less than the threshold thr are removed. The

TABLE I

TOP-1 NATURAL ACCURACY (IN %) ON CLEAN EXAMPLES AND ADVERSARIAL ACCURACY (IN %) ON ADVERSARIAL EXAMPLES. THE ADVERSARIAL EXAMPLES ARE GENERATED WITH PGD, CW, FGSM, FAB, AND NIFGSM ON CIFAR-10 AND TINY IMAGENET. IN THE METHODS PART, PGD, TRADES, AND MART ARE BENCHMARK METHODS. THE METHOD "(IB-RAR)" REFERS TO BENCHMARKS COMBINED WITH OUR METHOD. EACH RESULT IS THE AVERAGE OF THREE RUNS.

	CIFAR-10 with VGG-16						**Tiny ImageNet with VGG-16**					
Methods \ Inputs	Natural	PGD	CW	FGSM	FAB	NIFGSM	Natural	PGD	CW	FGSM	FAB	NIFGSM
PGD	75.02	42.45	37.80	47.32	41.03	47.59	37.54	17.73	13.77	19.46	13.76	22.14
PGD (IB-RAR)	**76.22**	**45.09**	**41.83**	**50.53**	**46.22**	**51.93**	**40.25**	**18.30**	**14.08**	**20.07**	**14.29**	**22.62**
TRADES	73.44	43.92	38.28	47.94	41.64	48.41	36.80	18.13	13.73	19.57	14.01	22.16
TRADES (IB-RAR)	**80.63**	**44.13**	**41.81**	**51.45**	**43.63**	**51.69**	**39.10**	**18.45**	**14.19**	**20.22**	**14.49**	**22.87**
MART	73.52	44.64	37.58	48.73	40.56	48.95	34.94	17.49	13.06	18.88	13.68	21.23
MART (IB-RAR)	**80.54**	44.34	**41.45**	**52.19**	**44.72**	**51.93**	**36.68**	**18.05**	**13.36**	**19.33**	**13.81**	**22.02**

TABLE II

TOP-1 NATURAL ACCURACY (IN %) ON CLEAN EXAMPLES AND ADVERSARIAL ACCURACY (IN %) ON ADVERSARIAL EXAMPLES. THE ADVERSARIAL EXAMPLES ARE GENERATED WITH PGD, CW, FGSM, FAB, AND NIFGSM ON CIFAR-10 AND CIFAR-100. IN THE METHODS PART, PGD, TRADES, AND MART ARE BENCHMARK METHODS. THE "METHOD (IB-RAR)" REFERS TO BENCHMARKS COMBINED WITH OUR METHOD. EACH RESULT IS THE AVERAGE OF THREE RUNS.

	CIFAR-10 with ResNet-18						**CIFAR-100 with WRN-28-10**					
Methods \ Inputs	Natural	PGD	CW	FGSM	FAB	NIFGSM	Natural	PGD	CW	FGSM	FAB	NIFGSM
PGD	75.05	45.21	74.09	48.60	42.26	49.71	39.88	9.74	13.66	16.85	10.28	14.53
PGD (IB-RAR)	**75.10**	**45.55**	**74.10**	**48.83**	**42.74**	**50.03**	37.68	**16.60**	**15.98**	**19.44**	**14.85**	**19.48**
TRADES	73.04	45.91	72.16	48.51	42.59	49.92	39.38	10.44	14.69	17.60	10.42	15.38
TRADES (IB-RAR)	**73.07**	**46.13**	72.16	**48.85**	**42.74**	**50.09**	36.41	**19.18**	**16.67**	**20.69**	**16.61**	**21.95**
MART	72.96	46.17	72.00	49.19	41.62	50.34	39.91	12.30	14.29	17.85	11.73	16.57
MART (IB-RAR)	**76.85**	**48.92**	**75.78**	**52.52**	**45.01**	**54.72**	**40.65**	**23.44**	**17.96**	**24.46**	**19.24**	**26.41**

threshold is decided according to the sorted MI values of these feature channels. Empirically, we use a small threshold to eliminate 5% of all feature channels. In other words, the MI values of that 5% of feature channels are lower than the MI values of all other channels. The threshold is the maximum of MI values of that 5% of feature channels. The application of the mask is shown in Algorithm 1. Note that removing unnecessary features is built on our MI loss, as it requires non-robust features to be more distinct from other features concerning MI values. Experimental evidence can be found in the ablation study, row (5) of Table V.

III. EXPERIMENTAL EVALUATION

Following prior literature, experiments are conducted with four standard datasets: CIFAR-10 [13], SVHN [18], CIFAR-100 [13], and Tiny ImageNet [19]. We use VGG-16 [21] for CIFAR-10, SVHN, and Tiny ImageNet. We use ResNet-18 [9] for CIFAR-10, SVHN. We use WideResNet-28-10 [26] for CIFAR-100. The implementation is built with PyTorch and Torchattacks [11] frameworks.

Algorithms: We evaluate our method with the following adversarial learning algorithms: Projected Gradient Descent (PGD) [17], TRADES [27], and MART [24]. Clean examples are not used for PGD adversarial training but are used in TRADES and MART for evaluation following previous works. We combine our method with these algorithms and compare them against the performance of the original algorithms. In addition, we also compare our method to non-adversarial training algorithms: Cross-Entropy, HSIC Bottleneck as Regularizer (HBaR) [25], and Variational Information Bottleneck (VIB) [1].

Metrics: We evaluate accuracy on natural inputs (Test Acc., i.e., accuracy on clean data) and adversarial examples (Adv. Acc.) for all algorithms. The adversarial examples are generated by: (1) PGD^n [17], the PGD attack with n steps in optimization; (2) FGSM [7]; (3) CW [3]; (4) FAB [5]; (5) NIFGSM [14]. We set parameters for attacks (implemented with Torchattacks) following the prior literature: step size = 2/255 (alpha), r = 8/255 (eps, the limitation for perturbation δ), default steps= 10, and CW steps = 200.

We use the following hyperparameters for all training:

- StepLR: lr = 0.01, step_size=20, gamma=0.2.
- Optimizer: SGD, weight_decay=1e-2.
- Maximum epoch: 60.
- Batch size: 100.

Adaptive Evaluation. To demonstrate the effectiveness of IB-RAR as a defense, we provide two levels of adaptive evaluation: (1) To demonstrate that the success of IB-RAR is not limited to a few cases and that the attack algorithms converged, we use multiple attacks and iteration steps. The results of adversarial robustness are shown in Table I, Table II, and Figure 3. (2) We assume that the adversary designs a new attack specifically targeted to IB-RAR, as the adversary has full knowledge of IB-RAR and white-box access to the network, which is discussed in Appendix Section A-B.

A. Adversarial Robustness Results with Adversarial Training

We show that our method reaches better adversarial robustness along with state-of-the-art adversarial training benchmarks. Different regularizers (α and β in $\mathcal{L}$) are evaluated to find the optimal hyperparameters. We also find that our method can boost the convergence of the network.

1) Accuracy on Adversarial Examples: Tables I and II show test accuracy and adversarial accuracy results on CIFAR-10, CIFAR-100, and Tiny ImageNet, respectively. PGD refers to adversarial training with PGD examples. Results for SVHN are in Appendix due to space limitation. TRADES and MART are baseline methods mentioned in the experimental setting. The "method (IB-RAR)" refers to using our method to improve the baseline method, i.e., using the mutual information loss in Eq. (2) and using the mask in Eq. (3) to remove unnecessary feature channels.

Combined with all benchmark methods, our method improves adversarial robustness compared to baselines. In Table I, our method also improves the test accuracy on clean examples, especially for TRADES and MART. Note that we use clean examples to compute MI in Eq. (2). Suppose we use adversarial examples to compute MI, i.e., using $I(X+\delta, T_l)$ to build the IB objective in the loss. In that case, the performance increases when defending against the PGD attack (or keeping almost the same performance) but decreases the performance against other attacks.

B. Robustness Without Adversarial Training

1) Using Partial Layers is Better: We empirically show that using partial layers to compute IB loss is better, as shown by results provided in Table III. We deploy MI loss (Eq. (1)) to every layer of VGG-16 and use CIFAR-10 for training, as there are both convolutional and fully connected layers in the VGG structure. Each row in Table III shows the performance of a network, which is trained by computing IB loss (Eq. (1)) for a single block of VGG-16.

Clearly, the robustness mainly comes from Conv Block 5 (the fifth convolutional block in VGG-16), FullyC 1 (the first fully connected layer in VGG-16), and FullyC 2. We refer to Conv Block 5, FullyC 1, and FullyC 2 as robust layers, as they provide obvious robustness compared to other layers. Using all layers to build the two regularizers for MI loss degrades its robustness compared to using robust layers. Based on MI loss, we further improve its robustness on other convolutional blocks by filtering out unnecessary feature channels, see row (2) and row (6) in Table V for comparison. Compared to using all layers or other single layers, our method provides the best accuracy (35.86%) under the PGD^{10} attack. This is achieved by the defender without any prior knowledge of adversarial examples.

This phenomenon also occurs in other networks trained with other datasets, i.e., every single hidden layer can provide a different degree of adversarial robustness when computing MI of intermediate representation for IB objective. Their behaviors are similar but not the same. For example, when VGG-16 is trained with SVHN, the last four layers provide adversarial accuracy, i.e., Conv Block 4 (6.44%), Conv Block 5 (16.83%), FullyC 1 (8.97%), and FullyC 2 (9.98%). When training ResNet-18 with SVHN, the last layer provides higher adversarial accuracy (6.13%) than other layers. The accuracy of trained VGG-16 and ResNet-18 with CE only and SVHN dataset is lower than 1%. Based on our observations, the robust layers will be the last few layers of the network.

Layer	Adv. acc.	Test acc.
Conv Block 1	0.04	89.32
Conv Block 2	0.05	90.17
Conv Block 3	0.02	90.53
Conv Block 4	0.01	89.66
Conv Block 5	8.25	89.58
FullyC 1	9.85	91.04
FullyC 2	3.27	90.97
All Layers	25.61	91.96
Rob. Layers	35.86	90.97

TABLE III
THE ADV. ACC. AND TEST ACC. OF USING A SINGLE LAYER TO COMPUTE MI IN EQ. (1) FOR OUR METHOD. THE ADV. ACC. IS EVALUATED UNDER OUR DEFAULT PGD ATTACK. THE NETWORK IS VGG-16 TRAINED WITH CIFAR-10. ROB (ROBUST) LAYERS REFERS TO USING OUTPUTS OF BLOCK 5, FULLYC 1, AND FULLYC 2.

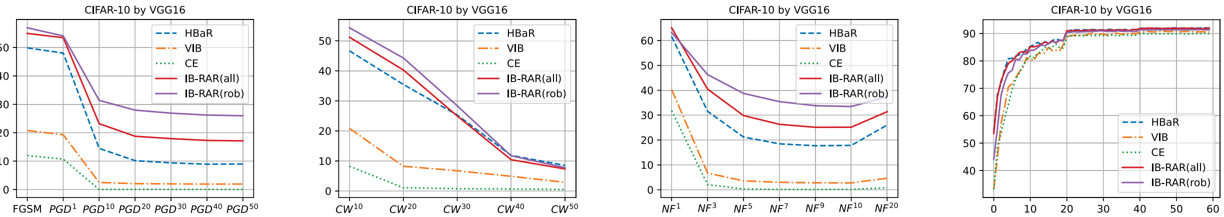
2) Comparison with Other IB-related Methods: Figure 3 shows that our method achieves the best robustness compared to other IB-based baselines when training without adversarial examples. Specifically, we compare our method with CE, HBaR [25], and VIB [1] under the same conditions. CE refers to training with only cross-entropy loss function, i.e., no defense on this baseline.

We obtain improved accuracy on clean data compared to VIB and CE only. Our method achieves the natural accuracy for IB-RAR(rob) of 91.33%, IB-RAR(all) of 91.97%, while HBaR, VIB, and CE only achieve 91.93%, 90.52%, and 89.88%, respectively.

We use a progressively increasing number of steps to make sure the attacks are converged. Under PGD, CW, and NIFGSM attacks, our method continuously shows better accuracy compared to baselines.

C. Discussion and Future Work

One possible explanation for why our method works is that there are shared features among different classes of training data. Shared features refer to the similar characteristic of objects in two classes of data. For example, cats and dogs are very similar, while cats and airplanes are not so similar in terms of shape. Clusters in Figure 4(a) also show the similarity between classes in terms of distance. The MI loss and mask reduce that shared feature and increase the distance among classes in Figure 4.



(a) PGD Attacks (b) CW Attacks (c) NIFGSM Attacks (d) Clean Inputs

Fig. 3. CIFAR-10 with VGG-16: comparison of our method and IB-based baselines. The performance is evaluated under different optimization steps of (a) PGD attacks, (b) CW attacks, (c) NIFGSM attacks, and (d) clean data. The IB-RAR(rob) refers to our method, which uses IB regularizers for only robust hidden layers. The IB-RAR(all) refers to using IB regularizers for all hidden layers. The accuracy on clean data at the last epoch is IB-RAR(rob) 91.33%, IB-RAR(all) 91.97%, HBaR 91.93%, VIB 90.52%, CE only 89.88%. Each result is the average of three runs.

In addition, to check whether similar classes tend to be classified as each other, we evaluate the number of times the network predicted the adversarial example as a specific class (top-4 classes) The test set of CIFAR-10 is used to generate the adversarial examples, containing 1000 images for each class. In Table IV, cars are thought to be the truck 681 times. The highest number of classifications of the truck class is also car class, i.e., 427 times. Such a bidirectional tendency also exists in other classes. The network learned the most often shared features from these pairs compared to other classes. It is easier for the adversary to find imperceptible perturbations of similar pairs, as their distance in classification should be close to each other.

This intuitive idea can be a starting point for future work that could investigate, for instance, the following aspect. Currently, our method builds the IB objective by using inputs, outputs, and intermediate network representations. It is not specifically designed for adversarial perturbation or shared features. The straightforward next step is distilling shared features for every class since the shared features could help adversarial attack algorithms find small enough perturbations. Then according to distilled features, the network can learn well-generalized features but discard shared features. As discarding shared features may also result in the loss of useful information for the class, a key problem might be controlling the trade-off between discarding shared features and retaining enough information for generalization.

IV. Related Work

The IB is supposed to find the optimal trade-off between compression of input X and prediction of Y [20]. Empirically, IB provides both performance and adversarial robustness when embedded into the learning objective. Alemi et al. [1] proposed Variational Information Bottleneck (VIB). They use the internal representation of a certain intermediate layer as a stochastic encoding Z of the input data x. They aim to learn the most informative representation Z about the target Y, measured by the mutual information between their corresponding encoding values. Their experiments also showed that VIB is robust to overfitting and adversarial attacks. Ma et al. [16] proposed the HSIC Bottleneck and replaced mutual information with Hilbert Schmidt Independence Criterion (HSIC). They used the HSIC bottleneck as a learning objective for every layer of the network, an alternative to the conventional CE loss and back-propagation. Wang et al. [25] proposed HBaR, which combined HSIC Bottleneck of all hidden layers and back-propagation to improve both adversarial robustness and accuracy of clean data. Previous works tend to use one layer (such as VIB) or all layers (such as HSIC Bottleneck and HBaR) for variants of IB, but we find that the IB of each layer has a different impact on robustness. As such, using partial layers will increase the robustness against adversarial attacks, as discussed in Section III-B1.

V. Conclusions

This paper proposes an improved IB-based loss function to improve adversarial robustness and a feature channel mask to remove unnecessary features. We first use IB as a regularizer to improve robustness on fully connected layers and learn better generalization of input data. Based on a well-generalized network, we remove less relevant feature channels of convolutional layers according to MI between these channels and the true labels. We also discuss that using partial layers for MI loss improves robustness against adversarial attacks. Our experimental results show that our method consistently improves the adversarial robustness of state-of-the-art adversarial training technologies. The IB-RAR method improves accuracy by an average of 2.66% against five adversarial attacks for all networks in Tables I and II, compared to three adversarial training benchmarks. IB-RAR can also provide modest robustness to weaker methods without prior knowledge of adversarial examples. Our findings also show that our method increases the accuracy of clean data, as the noise in input data is removed. Finally, our experimental evaluation results demonstrate that our method can enhance the robustness against various adversarial attacks.

References

[1] Alexander A. Alemi, Ian Fischer, Joshua V. Dillon, and Kevin Murphy. Deep variational information bottleneck. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017.

[2] 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 Hendrik Blockeel, Kristian Kersting, Siegfried Nijssen, and Filip Železný, editors, *Machine Learning and Knowledge Discovery in Databases*, pages 387–402, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg.

[3] Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *2017 IEEE Symposium on Security and Privacy (SP)*, pages 39–57, 2017.

[4] Saikat Chakraborty, Rahul Krishna, Yangruibo Ding, and Baishakhi Ray. Deep learning based vulnerability detection: Are we there yet. *IEEE Transactions on Software Engineering*, pages 1–1, 2021.

[5] Francesco Croce and Matthias Hein. Minimally distorted adversarial examples with a fast adaptive boundary attack. In Hal Daumé III and Aarti Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 2196–2205. PMLR, 13–18 Jul 2020.

[6] Thomas Fischer and Christopher Krauss. Deep learning with long short-term memory networks for financial market predictions. *European Journal of Operational Research*, 270(2):654–669, 2018.

[7] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

[8] Arthur Gretton, Olivier Bousquet, Alex Smola, and Bernhard Schölkopf. Measuring statistical dependence with hilbert-schmidt norms. In Sanjay Jain, Hans Ulrich Simon, and Etsuji Tomita, editors, *Algorithmic Learning Theory*, pages 63–77, Berlin, Heidelberg, 2005. Springer Berlin Heidelberg.

[9] 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 (CVPR)*, June 2016.

[10] Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Logan Engstrom, Brandon Tran, and Aleksander Madry. Adversarial examples are not bugs, they are features. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

[11] Hoki Kim. Torchattacks: A pytorch repository for adversarial attacks. *arXiv preprint arXiv:2010.01950*, 2020.

[12] Junho Kim, Byung-Kwan Lee, and Yong Man Ro. Distilling robust and non-robust features in adversarial examples by information bottleneck. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 17148–17159. Curran Associates, Inc., 2021.

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

[14] Jiadong Lin, Chuanbiao Song, Kun He, Liwei Wang, and John E. Hopcroft. Nesterov accelerated gradient and scale invariance for adversarial attacks. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020.

[15] Junshui Ma, Robert P. Sheridan, Andy Liaw, George E. Dahl, and Vladimir Svetnik. Deep neural nets as a method for quantitative structure–activity relationships. *Journal of Chemical Information and Modeling*, 55(2):263–274, 2015. PMID: 25635324.

[16] Wan-Duo Kurt Ma, J. P. Lewis, and W. Bastiaan Kleijn. The hsic bottleneck: Deep learning without back-propagation. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(04):5085–5092, Apr. 2020.

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

[18] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y. Ng. Reading digits in natural images with unsupervised feature learning. In *NIPS Workshop on Deep Learning and Unsupervised Feature Learning 2011*, 2011.

[19] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252, 2015.

[20] Ravid Shwartz-Ziv and Naftali Tishby. Opening the black box of deep neural networks via information. *CoRR*, abs/1703.00810, 2017.

[21] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[22] 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.

[23] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(86):2579–2605, 2008.

[24] Yisen Wang, Difan Zou, Jinfeng Yi, James Bailey, Xingjun Ma, and Quanquan Gu. Improving adversarial robustness requires revisiting misclassified examples. In *International Conference on Learning Representations*, 2019.

[25] Zifeng Wang, Tong Jian, Aria Masoomi, Stratis Ioannidis, and Jennifer Dy. Revisiting hilbert-schmidt information bottleneck for adversarial robustness. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 586–597. Curran Associates, Inc., 2021.

[26] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. In Edwin R. Hancock Richard C. Wilson and William A. P. Smith, editors, *Proceedings of the British Machine Vision Conference (BMVC)*, pages 87.1–87.12. BMVA Press, September 2016.

[27] Hongyang Zhang, Yaodong Yu, Jiantao Jiao, Eric Xing, Laurent El Ghaoui, and Michael Jordan. Theoretically principled trade-off between robustness and accuracy. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 7472–7482. PMLR, 09–15 Jun 2019.

APPENDIX A
ADDITIONAL RESULTS

A. Ablation Study

We conduct an ablation study to verify the effectiveness of the proposed mutual information loss and the mask to remove the unnecessary feature channels. Here, we refer to them as $\mathcal{L}$ and FC. The results are shown in Table V. The network in a row (1) of Table V is trained with CE loss function only. It gets almost zero accuracies on PGD and CW attacks and very low accuracy on FGSM, as training with CE only does is vulnerable. The network trained with mutual information loss, row (2) of $\mathcal{L}$, gains modest accuracy under adversarial attack,

Target class	**Predicted results**			
plane :	bird-352	ship-247	deer-156	truck-110
car :	truck-681	ship-166	plane-55	frog-24
bird :	deer-260	frog-259	dog-141	plane-120
cat :	dog-415	deer-173	bird-144	frog-134
deer :	bird-285	frog-196	cat-169	horse-147
dog :	cat-299	frog-208	bird-169	horse-143
frog :	cat-411	bird-240	deer-187	dog-63
horse :	dog-335	deer-335	truck-82	bird-75
ship :	plane-280	bird-196	truck-181	deer-116
truck :	car-427	ship-192	horse-135	plane-101

TABLE IV

THE ADVERSARIAL EXAMPLE CLASSIFICATION TENDENCY TABLE OF CIFAR-10 TRAINED WITH VGG-16. THE TARGET CLASS COLUMN IS THE GROUND TRUTH. THE REST OF EACH ROW IS THE PREDICTION RESULTS AND THE CLASS COUNT. CLASS COUNT REFERS TO THE NUMBER OF TIMES AN INPUT OF THE TARGET CLASS WAS CLASSIFIED AS THAT CLASS.