MIMIR: Masked Image Modeling for Mutual Information-based Adversarial Robustness

Xiaoyun Xu[1], Shujian Yu[2], Zhuoran Liu[1] and Stjepan Picek[3,1]
[1]Radboud University Nijmegen, The Netherlands
[2]Vrije Universiteit Amsterdam, The Netherlands
[3]University of Zagreb Faculty of Electrical Engineering and Computing, Croatia
Email: xiaoyun.xu@ru.nl, s.yu3@vu.nl, z.liu@cs.ru.nl, stjepan.picek@ru.nl

***Abstract*—Vision Transformers (ViTs) have emerged as a fundamental architecture and serve as the backbone of modern vision-language models. Despite their impressive performance, ViTs exhibit notable vulnerability to evasion attacks, necessitating the development of specialized Adversarial Training (AT) strategies tailored to their unique architecture. While a direct solution might involve applying existing AT methods to ViTs, our analysis reveals significant incompatibilities, particularly with state-of-the-art (SOTA) approaches such as Generalist [1] (CVPR 2023) and DBAT [2] (USENIX Security 2024). This paper presents a systematic investigation of adversarial robustness in ViTs and provides a novel theoretical Mutual Information (MI) analysis in its autoencoder-based self-supervised pre-training. Specifically, we show that MI between the adversarial example and its latent representation in ViT-based autoencoders should be constrained via derived MI bounds. Building on this insight, we propose a self-supervised AT method, MIMIR, that employs an MI penalty to facilitate adversarial pre-training by masked image modeling with autoencoders. Extensive experiments on CIFAR-10, Tiny-ImageNet, and ImageNet-1K show that MIMIR can consistently provide improved natural and robust accuracy, where MIMIR outperforms SOTA AT results on ImageNet-1K. Notably, MIMIR demonstrates superior robustness against unforeseen attacks and common corruption data and can also withstand adaptive attacks where the adversary possesses full knowledge of the defense mechanism. Our code and trained models are publicly available at: https://github.com/xiaoyunxxy/MIMIR.**

I. Introduction

ViTs [3] and their variants [4], [5] have achieved substantial progress and serve as foundational components in modern vision-language models. Prominent multimodal frameworks, including CLIP [6], BLIP [7], and Mini-GPT4 [8], typically employ ViTs as their image encoders. However, similar to convolutional neural networks (CNNs), attention-based models provide limited robustness against evasion attacks [9], [10], [11], [12], [13]. Evasion attacks [14], [15] (also known as adversarial attacks), where well-trained deep models are fooled by introducing human-imperceptible perturbations to inputs, remain a persistent challenge in deep learning security. In 2024, the National Institute of Standards and Technology (NIST) explicitly listed adversarial attacks as a significant threat to AI systems, and pointed out the importance of conducting robustness testing and mitigation, such as adversarial training and formal verification, when deploying AI tools [16]. Nevertheless, improving adversarial robustness remains a difficult task, where even SOTA methods, such as [17], [18], [19], achieve only marginal robustness gains, commonly below 2% compared with those before them.

So far, Adversarial Training (AT) is widely recognized as the most practically effective defense [12], [11], [20] against evasion attacks. AT operates by augmenting the training dataset with adversarially perturbed samples [21], yet introduces two key limitations: (1) substantial computational overhead due to the generation of adversarial examples during training [21], and (2) a potential degradation in natural accuracy [22]. Numerous methods have been proposed to mitigate these challenges. Techniques such as FreeAT [23] optimize efficiency by reusing gradient information during adversarial example generation, while FastAT [24] employs an improved Fast Gradient Sign Method (FGSM) to accelerate training. TRADES [25], SCORE [26], Generalist [1], and DBAT [2] explore how to achieve the best trade-off between natural and robust accuracy. Additionally, pre-training strategies have also been leveraged to enhance the performance of AT [27], [17].

Applying existing AT methods to ViTs presents unique challenges due to the fundamental differences between attention-based architectures and CNNs. Unlike CNNs, ViTs lack inductive biases [3], including locality, two-dimensional neighborhood structure, and translation equivariance. These biases are inherent to CNNs as prior knowledge, enabling efficient learning with limited data, whereas ViTs typically require larger training datasets to achieve comparable generalization performance [3]. Consequently, AT for ViTs entails significantly higher computational costs. Initial research on AT for ViTs explored their unique attention mechanism. For instance, robustness can be improved by randomly dropping gradients according to attention [11] or improving training efficiency by dropping low-attention image embeddings [12]. The majority of recent works have focused on adapting CNN-based AT methodologies to ViTs, given AT's success in building robust CNNs. Unfortunately, standard CNN AT techniques are not fully transferable to ViTs. Empirical studies [9], [20] reveal that *strong data augmentations* (such as Randaugment [28],

Network and Distributed System Security (NDSS) Symposium 2026
23-27 February 2026, San Diego, CA, USA
ISBN 979-8-9919276-8-0
https://dx.doi.org/10.14722/ndss.2026.241813
www.ndss-symposium.org

CutMix [29], and MixUp [30], which improve robustness in CNNs) often degrade AT performance for ViTs. To mitigate this, recent work [9] suggests progressively increasing augmentation intensity (e.g., distortion magnitudes in RandAugment or the sampling probability of MixUp/CutMix) during training. Furthermore, SOTA AT strategies, such as Generalist [1] and DBAT [2], are less effective when applied to ViTs (see Table I), and there is a lack of evaluation on large datasets, such as ImageNet-1K, which further limits their generalizability.

Pre-training has emerged as a complementary approach to ViT AT, with studies showing that adversarial fine-tuning of naturally pre-trained models can enhance robustness [11], [17]. AdvXL [31] notably advanced this paradigm by developing efficient AT for web-scale datasets. However, the mechanisms underlying pre-training's effectiveness are not fully understood, and results are inconsistent across implementations. For instance, models pre-trained on ImageNet-21K using SimMIM [32] demonstrate comparable performance to scratch-trained counterparts, while CLIP [6] pre-training has been shown to degrade performance in some configurations [33].

While previous methods of ViT AT, such as [17], [20], [9], focus on searching for better combinations of training hyperparameters, they suffer from performance drops across different architectures and datasets. In contrast, we aim for a generalizable method via pre-training. Specifically, this work presents a systematic investigation of self-supervised pre-training for ViT robustness through the lens of Mutual Information (MI) and Information Bottleneck (IB) theory. IB introduces a joint objective of simultaneously minimizing the MI between inputs and latent features while maximizing the MI between labels and latent features to mitigate the impact of the adversarial noise in the inputs. Regarding the ViT AT, we develop a novel theoretical justification for self-supervised autoencoders, demonstrating that reducing MI between inputs and latent features enhances ViT robustness. Based on this finding, we propose a theoretically grounded adversarial pre-training method, **M**asked **I**mage **M**odeling for Mutual **I**nformation-based Adversarial **R**obustness (**MIMIR**). [1] Specifically, we convert Masked Image Modeling (MIM) into an effective and efficient adversarial pre-training method. The basic idea is to predict the masked content of inputs, which is a self-supervised learning task. The effectiveness of MIMIR is analyzed and guaranteed by our theoretical justification. The efficiency comes from discarding the masked content (75% of image patches are discarded in our experiments), which greatly reduces the computing requirements. Figure 1 provides an illustrative diagram of MIMIR.

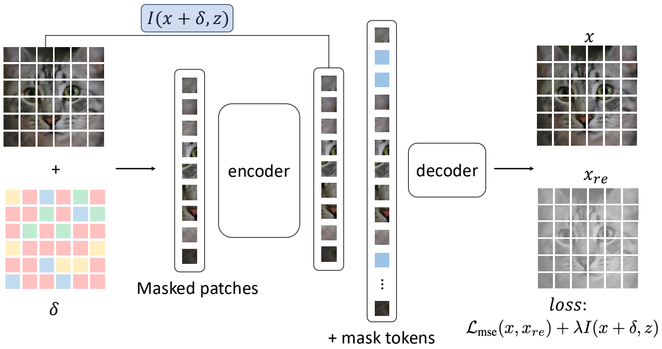

Fig. 1. Diagram illustrating the working mechanism of MIMIR. In the pre-training, adversarial images $x+\delta$ are generated and separated into image patches as the inputs of the encoder. The output of decoder x_{re} and the natural input image x are used to calculate the loss. After pre-training, a trained encoder is combined with a randomly initialized classification layer as the final complete model. Then the complete model is further fine-tuned for classification tasks.

We validate MIMIR's effectiveness through extensive experiments on CIFAR-10 [34], Tiny-ImageNet [35], and ImageNet-1K [36], showing consistent improvements in both natural and adversarial accuracy. In addition, we test the generalizability of MIMIR by combining MIMIR with three MIM methods for three representative architectures, including MAE [37] for ViTs, Group Window Attention [38] for hierarchical transformer (Swin [39]), and SparK [40] for CNN (ConvNext [41]), where MIMIR outperforms SOTA AT methods on ImageNet-1K. Our main contributions are:

- By revisiting the current ViT AT strategies, we point out that ViT AT methods compromise natural accuracy and lack systematic study. To this end, we provide a theoretical analysis using adversarial examples and the Information Bottleneck. We theoretically show that the Mutual Information between adversarial examples and the learned latent representation should be decreased for better robustness.
- Based on our analysis, we propose a self-supervised defense, MIMIR, against adversarial attacks on ViTs. We evaluate MIMIR using multiple architectures on three datasets under various adversarial attacks, demonstrating its effectiveness. MIMIR achieves SOTA adversarial robustness on ImageNet-1K following the standardized evaluation by RobustBench.[2]
- We show that MIMIR is robust against unforeseen attacks and common corrupted data (ImageNet-C [42]) and can resist adaptive attacks where the adversary is aware of MIMIR's design.

II. Background

A. Evasion Attack

Evasion attacks [43], [14], [15], also known as adversarial attacks, refer to applying imperceptible perturbations to the original input of the machine learning model (in this work, neural network), which generates adversarial examples to fool the victim model. Given an L-layer neural network F_θ for classification in d_Y-dimensional space and a training dataset $D=\{(x_i,y_i)\}_{i=1}^n$ in d_X-dimensional space, the two primary goals of adversarial attacks are:

[1] Mimir is a figure in Norse mythology, renowned for his knowledge and wisdom.

[2] https://robustbench.github.io/

1) The generated perturbation δ can successfully mislead the network by maximizing (e.g., PGD attack [21]):

$$\max_{\delta \in S} \mathcal{L}_{CE}(\theta, x_i + \delta, y_i), \tag{1}$$

where $x_i \in \mathbb{R}^{d_X}$ and $y_i \in \{0,1\}^{d_Y}$, θ are the parameters of the current network and $\mathcal{L}_{CE}$ is the standard CE (Cross-Entropy) loss. The perturbation aims to decrease confidence in the ground truth labels while increasing confidence in wrong labels. Thus, the loss between the misleading outputs and the ground truth labels increases.

2) The generated adversarial examples are as similar as possible to the original clean examples by limiting δ to a relatively small domain:

$$S = B(x_i, r) = \{\delta \in \mathbb{R}^{d_X} : \|\delta\|_\infty \leq r\}, \tag{2}$$

where S is the l_∞-ball of radius r at a position x_i in $\mathbb{R}^{d_X}$. The distance between x_i and $x_i + \delta$ can be evaluated by norms such as l_2 and l_∞.

When using the above attacks to generate adversarial examples for AT, the learning objective is:

$$\min_{\theta} \max_{\delta \in S} \mathcal{L}_{CE}(\theta, x_i + \delta, y_i). \tag{3}$$

B. Masked Image Modeling - MIM

MIM refers to a self-supervised pre-training framework that aims to reconstruct pre-defined targets, such as discrete tokens [44], raw RGB pixels [37], [45], or features [46]. The final goal is to use the pre-trained model as a starting point for downstream fine-tuning. The downstream tasks include, for instance, classification and object detection. More specifically, to build a high-performance ViT f_e without a classification layer, we consider f_e as an encoder to extract discriminative input features. Then, we design a lightweight decoder f_d, which uses the output of f_e as its input. The goal of f_d is to reconstruct the original inputs (let us consider MAE [37] as an example). The aim is to decrease the distance between the input x and $x_{re} = f_d \circ f_e(x)$. After the encoder f_e and decoder f_d are trained, we use f_e plus a manually initialized classification layer as the starting point of fine-tuning.

Text

C. Mutual Information - MI

MI measures the mutual dependence between two random variables, X and Y. It quantifies the amount of information contained in one random variable about another random variable or the reduced uncertainty of a random variable when another random variable is known. It can be written as:

$$I(X,Y) = \int_{\mathcal{Y}} \int_{\mathcal{X}} P_{(X,Y)}(x,y) \log\left(\frac{P_{(X,Y)}(x,y)}{P_{(X)}(x)P_{(Y)}(y)}\right), \tag{4}$$

where $P_{(X,Y)}$ is the joint probability density function of X and Y. $P_{(X)}$ and $P_{(Y)}$ are the marginal probability density function of X and Y. MI can be equivalently expressed as:

$$I(X,Y) = H(X) - H(X|Y). \tag{5}$$

Estimating the exact MI is not easy, as it is difficult to precisely estimate $P_{X,Y}$ or P_X and P_Y in high-dimensional space [47]. In practice, Deep Deterministic Information Bottleneck (DIB) [48] suggested using the matrix-based Renyi's α-order entropy I_α [49], [50] to estimate MI, which avoids density estimation and variational approximation. An alternative way is the Hilbert-Schmidt Independence Criterion (HSIC) [51], which is a kernel-based dependence measure defined in a reproducing kernel Hilbert space (RKHS) and is usually used as a surrogate of MI. Details about definitions and empirical estimators of I_α and HSIC are provided in Appendix J. In this paper, we evaluate both I_α and HSIC as our MI measurements.

D. Information Bottleneck - IB

The IB concepts were first proposed in [52] and further developed for deep learning in [53], [54]. IB describes the generalization of a deep network in two phases: 1) empirical error minimization (ERM) and 2) representation compression [54]. For a network with input x and label y, there is an intermediate representation t_l for each layer l, i.e., the output of the l-th layer. The IB principle aims to keep more relevant information in t_l about the target y while decreasing the irrelevant information about the input x. The information between intermediate representation t_l and input x or label y is quantified by MI, denoted by $I(\cdot)$. During neural network training, in the ERM phase, the model increases shared information between t_l with respect to both x and y. Afterward, in the compression phase, the model decreases information contained in t_l about x but preserves (or even increases) information about y. The reduction of $I(x, t_l)$ can be interpreted as a way of reducing noise or compressing irrelevant or redundant features in x. At the end of the training, the model strikes a trade-off that maximizes $I(y, t_l)$ and minimizes $I(x, t_l)$. Formally, the IB minimizes the following Lagrangian:

$$\mathcal{L} = I(x, t_l) - \beta I(y, t_l), \tag{6}$$

where β is a Lagrange multiplier that controls the trade-off between predicting y and compressing x.

III. MIMIR

A. Threat Model

Adversary's goal. The attacker aims to fool the trained model with both non-targeted and targeted attacks. The goal is to decrease the overall classification accuracy (non-targeted) or compel the model to recognize any inputs as a specific target (targeted). Meanwhile, the adversarial perturbations applied to the input should be invisible so that they will not be easily detected. During the training phase, the model optimizes its parameters to minimize the loss between predicted outputs and true labels, thereby enhancing classification accuracy. In contrast, the adversary's objective is to develop algorithms that generate perturbations capable of maximizing this loss. For a targeted attack, the attacker decreases the loss between the output and the specified target label. To maintain the imperceptibility of the perturbations, the magnitude of the adversarial modifications is constrained by distance measures (such as $l2$ and l_∞), ensuring that the alterations to the input data remain within a visually indistinguishable range.

Adversary knowledge. The attacker has white-box access to the model, including training data, architectures, hyperparameters, and model weights. The attacker can implement iterative attacks and unlimited queries to update adversarial examples multiple times in white and black-box settings. Adversarial examples can be created according to model architectures, parameters, the gradients of the loss function, and datasets. In addition, we also consider adaptive adversaries who are also aware of potential defenses. The adversary can design new attacks for a specific model according to the design details of the defense method.

Defender's goal. From the defender's perspective, the main goal is to train a robust model against potential adversarial attacks. The defender considers the following objectives:

- The defender aims to prevent the performance of natural data from decreasing significantly, but allows a slight drop of natural accuracy for a trade-off in exchange for robustness.
- The defense method should provide a notable improvement compared to models without defenses when subjected to various adversarial attacks, especially to adaptive attacks that are aware of the details of the defense method.
- The defense method should be efficient and scalable to large datasets such as ImageNet-1K [36].

B. Design Intuition

MIM has been established as an effective pre-training paradigm for Vision Transformers (ViTs), demonstrating strong performance across diverse downstream tasks [55], [37], [46], [44]. The core methodology involves masking foreground regions of input images and tasking the model with their reconstruction. Masking foreground (as opposed to background) regions removes high-information content and results in a harder task (than reconstructing background content), forcing the model to develop stronger feature representations to reconstruct semantically meaningful patterns [56].

Building upon these principles, we introduce an adversarial extension to MIM by incorporating adversarial perturbations into the input space. Our formulation is grounded in three interconnected hypotheses:

- Adversarial Robustness through Reconstruction: If a model can reconstruct natural images from adversarially perturbed inputs, its latent representations must inherently discard perturbation-specific information while preserving natural data semantics.
- IB Perspective: The encoder-decoder architecture naturally imposes an information bottleneck. When processing adversarial examples $x + \delta$, the system must suppress perturbation-derived information (δ) while retaining natural data information (x) to achieve accurate reconstruction (see Figure 1).
- Optimal Masking Strategy: Complete foreground masking (to build a difficult task) is suboptimal, as it eliminates essential reconstruction signals. Instead, our method employs partial masking to maintain a tractable yet challenging learning objective.

Algorithm 1 MIMIR Pre-training

```
Input: training data $D$, number of epochs $E$, encoder $f_e$,
    decoder $f_d$, network parameters $\theta$, $\mathcal{L}_{\text{mse}}$, $\lambda$.
Output: optimized weights $\theta$
for $e = 0 \to E - 1$ do
    $x \leftarrow$ sample_batch($D$)
    $\delta \leftarrow$ random_initialization
    $x_{re} \leftarrow f_d \circ f_e(x + \delta)$
    $\delta \leftarrow \max_{\delta \in S} \mathcal{L}_{\text{mse}}(x + \delta, x_{re})$
    Forward:
    $z \leftarrow f_e(x + \delta)$
    $x_{re} \leftarrow f_d(z)$
    $loss \leftarrow \mathcal{L}_{\text{mse}}(x, x_{re}) + \lambda I(x + \delta, z)$
    Backward:
    $\theta \leftarrow \theta - \alpha \nabla loss$
end for
```

C. Design Details

Autoencoder. MIMIR consists of an encoder f_e and a decoder f_d aligned with the general design of MAE [37]. As with other autoencoders, the encoder extracts discriminative features z from inputs x. The decoder reconstructs the original inputs according to the discriminative features. Following the design of ViT [3], the input x is separated into non-overlapping image patches. We randomly mask out a part of the patches and use the remaining patches as inputs for the following process in the encoder. This random masking process uses uniform distribution to prevent potential sampling bias, such as all foreground information being masked, as it becomes infeasible to find the reconstruction target. Thus, we aim to keep a part of the foreground as a hint for reconstruction. The information of masked content is recorded as mask tokens m, which are not used by the encoder but reserved for later use by the decoder. Each token is a learned vector indicating the presence of a masked patch to be predicted. The mask token is shared by all inputs of the decoder. Like unmasked patches, mask tokens are also assigned corresponding positional embeddings to be in the correct location in the reconstructed image. We emphasize that mask tokens are not used in the encoder part.

To train a ViT, we use the same transformer blocks as ViT to build the encoder. The encoder only processes the visible patches, making training much more efficient. When converting to other architectures, such as ConViT [5], we use corresponding transformer blocks to build the encoder. The decoder accepts the encoded visible image patches and mask tokens as inputs. The decoder is built using the same transformer blocks as the encoder instead of using ViT [3] transformer blocks for all. Then, the decoder is followed by a fully connected layer, which outputs the same number of patches as the original image.

Adversarial Pre-training Target. The training target is to extract discriminative features from visible image patches by the encoder and then reconstruct the invisible patches by the decoder. Therefore, we need a differentiable measurement to quantify the distance between the original image and the reconstructed results. Following the original MAE [37], this distance is measured by Mean Squared Error (MSE). To create a more difficult reconstruction task, we apply adversarial perturbations δ to the inputs of the encoder. Thus, the adversarial perturbations are also masked along with the image upon input into the encoder. The decoder reconstructs the original natural inputs by using the latent features z extracted from adversarial examples. The outputs of decoder x_{re} and x are used to calculate the MSE loss, which is further used to optimize the model. Note that the reconstruction differs from the original MAE; we do not use the encoder inputs as reconstruction targets. Formally, the pre-training process (described in Algorithm 1) can be written as follows:

$$\begin{aligned} z = f_e(x+\delta),\ x_{re} = f_d(z), \\ loss_{\text{mse}} = \mathcal{L}_{\text{mse}}(x, x_{re}). \end{aligned} \tag{7}$$

MI as Penalty. Inspired by IB, we show in Section III-D that MI between latent representation and adversarial examples decreases as the accuracy on adversarial examples, i.e., $I(x+\delta,z)$, decreases while training. Motivated by this finding, we directly use $I(x+\delta,z)$ as a penalty in our final loss function:

$$loss_{\text{mi}} = \mathcal{L}_{\text{mse}}(x, x_{re}) + \lambda I(x+\delta, z), \tag{8}$$

where λ is a regularizer for the MI penalty. We use $I(x+\delta,z)$ instead of $I(x,z)$ as a penalty. This is because $x \to x+\delta \to z$ follows the Markov chain since z is extracted from $x+\delta$. According to Data Processing Inequality (DPI) [57], $I(x,z) \leq I(x+\delta,z)$. $I(x+\delta,z)$ is closer to z on the Markov chain.

Generating Adversarial Examples. To conduct the adversarial pre-training, we need an attack that finds proper adversarial perturbations δ. As the autoencoder does not provide classification outputs, it is not possible to directly use existing adversarial attacks, such as PGD [21]. Nevertheless, it is feasible to design a new algorithm to find δ by maximizing $loss_{\text{mse}}$ in Eq. (7). As the feature z is extracted from only visible image patches, we only attack the visible patches. We do not add any perturbations to mask tokens since the outputs of the autoencoder are only impacted by visible patches. Then, the adversarial pre-training learning objective can be written as:

$$\begin{aligned} \mathcal{L}_{\text{adv}} = \max_{\delta \in S} \mathcal{L}_{\text{mse}}(f_d \circ f_e(x+\delta), x), \\ \min_{\theta} \mathcal{L}_{\text{adv}} + \lambda I(x+\delta, z), \end{aligned} \tag{9}$$

where θ are the autoencoder parameters. After the autoencoder is trained, we discard the decoder and initialize a classification layer for the encoder to build a complete model. Finally, the complete model is fine-tuned by AT methods.

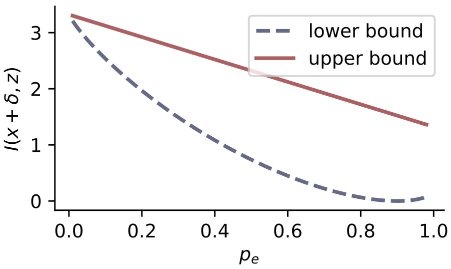

Fig. 2. The example plots for the lower and upper bounds on the MI in Propositions III.2 and III.3. The entropy ($H(\cdot)$) is chosen uniformly at random from a set of 10 classes. The lower bound reaches its minimum at $p_e = 0.9$.

D. Theoretical Justification

Next, we provide theoretical justification showing that MI between the adversarial example and its latent representation, i.e., $I(x+\delta,z)$, should be constrained. Let F denote any classifier trained on natural samples with desirable prediction accuracy, which may suffer from adversarial attacks. We begin our analysis by first presenting Lemma III.1.

Lemma III.1. *Let $F(x+\delta)$ and $F(x_{re})$ denote, respectively, the predicted labels of adversarial sample $x+\delta$ and reconstructed sample x_{re}, we have:*

$$I(F(x+\delta), F(x_{re})) \leq I(F(x+\delta), x_{re}) \leq I(x+\delta, z). \tag{10}$$

Proof. There are two Markov chains:

$$\begin{aligned} x+\delta \to F(x+\delta), \\ z \to x_{re} \to F(x_{re}), \end{aligned} \tag{11}$$

which implies that $F(x+\delta)$ is an indirect observation of $x+\delta$, whereas both $F(x_{re})$ and x_{re} are indirect observations of z.

By the data processing inequality (DPI), we have

$$I(x+\delta, z) \geq I(F(x+\delta), z), \tag{12}$$

and

$$I(F(x+\delta), z) \geq I(F(x+\delta), x_{re}) \geq I(F(x+\delta), F(x_{re})). \tag{13}$$

□

Now, we define p_e as the probability that the predicted label of $x+\delta$ by F is not equal to that of x_{re}, i.e., $p_e = \mathbb{P}(F(x+\delta) \neq F(x_{re}))$. Intuitively, our autoencoder is trained to recover only the natural sample x without any interference from δ. Hence, a relatively large value of p_e is expected. In the following, we establish the connection between p_e and $I(x+\delta,z)$ with both lower and upper bounds, showing that minimizing $I(x+\delta,z)$ also encourages a large value of p_e.

Proposition III.2. *Let $H(\cdot)$ denote the information entropy and $H_b(p_e) = -p_e \log_2 p_e - (1-p_e)\log_2(1-p_e)$ be the binary entropy, we have:*

$$H(F(x+\delta)) - H_b(p_e) - p_e \log(|F(x+\delta)|-1) \leq I(x+\delta, z), \tag{14}$$

where $|F(x+\delta)|$ is the total number of categories.[3]

Proof. By the chain rule of MI, we have

$$I(F(x+\delta), F(x_{re})) = H(F(x+\delta)) - H(F(x+\delta)|F(x_{re})). \tag{15}$$

By applying Fano's inequality [58], [59], we obtain:

$$H(F(x+\delta)|F(x_{re})) \leq H_b(p_e) + p_e \log(|F(x+\delta)| - 1). \tag{16}$$

Adding $I(F(x+\delta), F(x_{re}))$ to both sides of Eq. (16):

$$\begin{aligned} H(F(x+\delta)) - H_b(p_e) - p_e \log(|F(x+\delta)| - 1) \\ \leq I(F(x+\delta), F(x_{re})) \\ \leq I(x+\delta, z). \end{aligned} \tag{17}$$

The last line of Eq. (17) is by Lemma III.1. □

Therefore, we obtain a lower bound of $I(x+\delta, z)$. If we use CIFAR-10 ($|F(x+\delta)| = 10$) and assume the predicted labels $F(x+\delta)$ follow a uniform distribution, we can visualize the lower bound as a function of p_e as shown in Figure 2, from which we observe an obvious monotonic inverse relationship between $I(x+\delta)$ and p_e in the range $p_e \in [0, 0.9]$. In fact, we can also obtain an upper bound under the assumption that $I(F(x+\delta), x_{re}) \approx I(x+\delta, z)$, i.e., there is no information loss in the two Markov chains in Eq. (11).

Proposition III.3. *If $I(F(x+\delta), x_{re}) \approx I(x+\delta, z)$, we have:*

$$I(x+\delta, z) \lesssim H(F(x+\delta)) - 2p_e, \tag{18}$$

in which the notation "$\lesssim$" refers to less than or similar to.

Proof. By the Hellman-Raviv inequality [60], [61], we have:

$$\begin{aligned} 2p_e &\leq H(F(x+\delta)|x_{re}) \\ &= H(F(x+\delta)) - I(F(x+\delta), x_{re}) \\ &\approx H(F(x+\delta)) - I(x+\delta, z). \end{aligned} \tag{19}$$

□

Similar to the lower bound, the upper bound also indicates $I(x+\delta, z)$ is inversely proportional to p_e as shown in Figure 2. In fact, apart from the above-mentioned lower and upper bounds, there exists an alternative and intuitive way to understand the mechanism of minimizing $I(x+\delta, z)$. For simplicity, let us assume the natural data x and adversarial perturbations δ are independent[4], then:

$$I(x+\delta, z) = I(x, z) + I(\delta, z). \tag{20}$$

According to [63], minimizing the expected reconstruction error between natural sample x and corrupted input $x+\delta$ amounts to maximizing a lower bound of the mutual information $I(x, z)$, even though z is a function of the corrupted input. Therefore, by minimizing $I(x+\delta, z)$, the network is forced to minimize $I(\delta, z)$ (since $I(x, z)$ is maximized). In other words, only the adversarial information about δ has been removed from z when minimizing $I(x+\delta, z)$. This also explains the robustness of z.

TABLE I
COMPARISON BETWEEN END2END AT AND PRE-TRAINING (MIMIR) + FINE-TUNING USING VIT-S ON CIFAR-10.

Training	Method	Natural	PGD	AutoAttack
End2End	AT [21]	75.36	32.84	26.17
	Fast AT [24]	76.81	32.57	21.41
	TRADES [25]	74.96	32.12	24.90
	MART [66]	72.42	24.47	23.45
	Generalist [1]	60.88	14.44	11.20
	DBAT [2]	68.32	18.83	5.25
MIMIR	AT [21]	86.56↑11.20	56.76↑23.92	45.39↑19.22
	Fast AT [24]	87.22↑10.41	49.17↑16.6	35.89↑14.48
	TRADES [25]	88.19↑13.23	56.42↑24.3	51.70↑26.8
	MART [66]	80.55↑8.13	50.81↑26.34	39.92↑16.47
	Generalist [1]	88.81↑27.93	37.85↑23.41	33.67↑22.47
	DBAT [2]	88.56↑20.24	41.08↑22.25	24.59↑19.34

IV. EXPERIMENTAL EVALUATION

A. Experimental Setup

We evaluate MIMIR on three datasets: ImageNet-1K [36], Tiny-ImageNet [35], and CIFAR-10 [34], with diverse and commonly used architectures with multiple scales: ViT [3], ConViT [5], Swin Transformer [39], and ConvNext [41]. Details of datasets are provided in Appendix A. Hyperparameters of the decoder are included in Appendix B.

Training Setup. We train models from scratch for all experiments. Following MAE [37], we do pre-training by MIMIR for 800 epochs. Please note that we also compare our MIMIR + fine-tuning paradigm with the End2End paradigm. The End2End paradigm refers to the supervised training of a model from scratch without self-supervised pre-training. To compare between End2End and pre-training + fine-tuning, the common training schedule practice is pre-training 800 epochs + fine-tuning 100 epochs versus End2End training 300 epochs in the existing works [37], [32], [64].

The number of warmup epochs is 40 for pre-training. We use AdamW [65] as the optimizer for both pre-training and fine-tuning. We apply the cosine decay as the learning rate scheduler. At pre-training, MIMIR uses the 1-step PGD to generate adversarial examples for all three datasets. The perturbation budget is $\epsilon = 8, \alpha = 10$. For the fine-tuning stage, we use the 10-step PGD AT with perturbation bound $\epsilon = 8, \alpha = 2$ for CIFAR-10 and Tiny-ImageNet. For ImageNet-1K, the perturbation bound is $\epsilon = 4$. We use different steps of PGD or APGD to generate the adversarial perturbation for training. Details on training hyperparameters are provided in Appendix C.

Evaluation Metrics. We use **natural accuracy** and **robust accuracy** as evaluation metrics. Natural accuracy refers to

TABLE II
STANDARD DEVIATION OF 3 RUNS ON 3 DATASETS.

Performance	CIFAR-10	Tiny-ImageNet	ImageNet-1K
Natural	86.89±0.04	63.61±0.20	71.58±0.14
PGD	56.19±0.33	26.44±0.05	42.44±0.08

[3] For instance, for CIFAR-10, $|F(x+\delta)| = 10$.

[4] This assumption is mild for certain scenarios, such as when considering universal or image-agnostic perturbations [62].

TABLE III

Comparison with SOTA results on ImageNet-1K under $\epsilon = 4/255$. The "Adv. Steps" refers to attack steps for generating adversarial examples for AT. The results of [31] are evaluated using 20-step PGD (AutoAttack is designed as a more powerful alternative to PGD), which is marked as † in the table. Although AdvXL only uses 20 epochs, it takes more time due to huge datasets for pre-training and fine-tuning.

Architecture	Params (M)	FT	FT Epoch	Adv. Steps	Source	Natural	AutoAttack
DeiT-S	22.1	PGD	100	1	Augmentation warm-up [9]	66.62	36.56
DeiT-S	22.1	PGD	110	1	Light Recipe [20]	66.80	37.90
ViT-S	22.1	PGD	120	3	EasyRobust [67]	66.43	39.20
ViT-S	22.1	PGD	300	3	Adversarially Trained [33]	70.7	43.7
RobArch-S	26.1	PGD	110	3	RobArch [18]	70.17	44.14
ViT-S	22.1	APGD	300	2	Pre-training+AT [17]	69.22	44.04
ViT-S	22.1	PGD	300	3	MIMIR	**71.52**	45.90
ViT-S	22.1	APGD	300	2	MIMIR	71.00	46.10
ViT-S	22.1	APGD	300	3	MIMIR	70.96	**46.16**
ViT-B	86.6	ARD+PRM	10	5	ARD+PRM [11]	69.10	34.62
Swin-B	87.7	ARD+PRM	10	5	ARD+PRM [11]	74.36	38.61
ViT-B	86.6	PGD	120	3	EasyRobust [67]	70.64	43.04
Swin-B	87.7	PGD	120	3	EasyRobust [67]	75.05	47.42
RobArch-L	104	PGD	100	3	RobArch [18]	73.44	48.94
ViT-B	86.6	PGD	300	3	Adversarially Trained [33]	74.7	49.7
ViT-B	86.6	PGD	20	3	AdvXL [31]	73.4	53.0†
ViT-B	86.6	APGD	300	2	Pre-training+AT [17]	74.10	50.30
ViT-B	86.6	APGD	100	2	MIMIR	74.40	51.92
ViT-B	86.6	PGD	100	3	MIMIR	75.68	52.96
ViT-B	86.6	PGD	300	3	MIMIR	**76.98**	53.84
ViT-B	86.6	APGD	300	2	MIMIR	76.32	**54.28**

the accuracy of natural and unmodified inputs. The robust accuracy measures the accuracy under the AutoAttack (AA) [68]. AutoAttack is an ensemble of diverse parameter-free attacks, including white-box and black-box attacks. In our experiments, we use the standard version of AutoAttack that contains four attacks, including APGD-ce [68], APGD-t [68], FAB-t [69], and Square [70]. The perturbation budgets for evaluation are $\epsilon = 8$ for CIFAR-10 and Tiny-ImageNet, $\epsilon = 4$ for ImageNet-1K. In addition, we also evaluate the MIMIR-trained models with unforeseen attacks, such as CW attacks, attacks with l_2 norm, and out-of-distribution data (ImageNet-Corruption [42]).

Training stability. We also show that the natural accuracy and robustness of MIMIR are stable evaluation metrics. Due to the high computation cost, we cannot report the standard deviation for all experiments. To show that our method MIMIR has low variances, we train ViT-S on three datasets three times (1-step PGD AT for ImageNet-1K) and report the standard deviation and average performance in Table II.

B. Main Results

We first explore different AT methods and MIMIR for ViT on CIFAR-10, demonstrating the fundamental incompatibility between conventional AT approaches and ViT architectures. Following this baseline evaluation, we scale our investigation to the more challenging ImageNet-1K dataset, demonstrating the generalizability and scalability of our proposed MIMIR framework. The subsequent sections present comprehensive experimental results across three benchmark datasets: CIFAR-10, Tiny-ImageNet, and ImageNet-1K. This multi-scale evaluation strategy allows a thorough analysis of MIMIR's effectiveness under varying conditions, from smaller to large-scale visual recognition tasks.

In addition, we also explore the effectiveness of elucidating diffusion model (EDM) data on AT in Appendix D. This generated data is commonly used in AT to improve robustness [71], [72], [73], [74]. Specifically, we use 5 million generated CIFAR-10 data and 1 million Tiny-ImageNet data provided by [71]. Table XVIII in Appendix D shows that EDM data significantly improves the robustness.

CIFAR-10. Table I shows the performance of End2End adversarial training from scratch and Pre-training (MIMIR) + Fine-tuning on ViT-S trained on CIFAR-10. We provide the performance of 6 established or SOTA AT methods on CIFAR-10, indicating that traditional AT training strategies are not applicable to ViTs. Importantly, our experimental results also demonstrate that MIMIR can substantially improve all AT methods. The reason is that training ViTs from scratch is known to be difficult [3], [75] and even more difficult for adversarial training [11]. For example, robust accuracy is lower than 30% on ViT-B without pre-training [11]. In contrast, MIMIR provides a more straightforward methodology and avoids this difficulty by switching to pre-training with a theoretically grounded MIM learning task.

Time Consumption (End2End vs. Pre-training(PT)+Fine-tuning(FT)). Note that we follow the standard way to compare End2End and MIMIR (Pre-training + Fine-tuning) training methods by fixing the training schedule, following existing works [37], [32], [64], [77], [40], where we include pre-training 800 epochs + fine-tuning 100 epochs versus supervised End2End training of 300 epochs. The reason for having a larger number of pre-training epochs is that self-

TABLE IV
TIME CONSUMPTION OF END2END VS. PRE-TRAINING+FINE-TUNING.

Arch	GPU	AT Method	Epochs	Hours
ViT-S	4 A6000	PGD$_{10}$	300	187.64
	4 A6000	MIMIR(PT)+PGD$_{10}$(FT)	800(PT)+100(FT)	123.76
ViT-B	4 A6000	PGD$_{10}$	300	451.39
	4 A6000	MIMIR(PT)+PGD$_{10}$(FT)	800(PT)+100(FT)	263.77
Swin-L	4 H100	PGD$_3$	300	363.42
	4 H100	MIMIR(PT)+PGD$_3$(FT)	800(PT)+100(FT)	231.14

supervised pre-training is much more efficient (see Table XXII in Appendix I) than End2End training, and the pre-trained backbone can be used multiple times for various fine-tuning tasks. For example, in Table I, the 6 End2End AT methods cost $300 \times 6 = 1800$ fine-tuning epochs. MIMIR costs 800 pre-training epochs + 100×6 fine-tuning epochs, i.e., we only conduct the pre-training once for results in Table I. MIMIR pre-training epoch is more efficient than a fine-tuning epoch by discarding 75% of image patches.

More specifically, Table IV shows the total time consumption of End2End vs. Pre-training+Fine-tuning on three architectures with ImageNet-1K. Although MIMIR takes more training epochs, its pre-training+fine-tuning paradigm still costs less time than End2End adversarial training and gains much better performance on both natural and adversarial examples. Additional time consumption results with different datasets can be found in Table XXII in Appendix I.

ImageNet-1K. Table III compares MIMIR with previous works concerning adversarial robustness on ImageNet-1K ($l_\infty, \epsilon = 4/255$), which follows the evaluation of common standardized RobustBench [78]. Similar to other works [20], [33], [17], we consider simpler AT methods (i.e., PGD and APGD AT) instead of the latest AT methods, such as Generalist [1] and DBAT [2]. Indeed, since the latest methods introduce tailored components for CNNs to improve their adversarial robustness, they might not be effective for ViTs. The number of parameters, training epochs, steps in the inner maximization of AT, and clean and robust accuracy are reported to provide a more detailed understanding of the performance. The robust accuracy is evaluated by AutoAttack on the RobustBench [78] validation set (5,000 images). We divide the models into: *small* ($\approx$ 22M) and *large* ($\approx$ 86M) models, corresponding to ViT-S and ViT-B. Experimental results demonstrate that MIMIR outperforms all previous works across various training setups.

MIMIR with Various Architectures. In Table VI, we show that MIMIR can be applied to diverse architectures. Specifically, we use three representative options, including ViT+convolutional blocks (CVST) [17], the latest CNN architecture (ConvNext [41]), and a hierarchical vision transformer (Swin [39]). The ViT+CVST refers to using ConvStem [79] to replace the patch embedding in ViTs with a convolutional block. The ViT+CVST shows improved robustness compared to pure ViT models according to experiments in [17].

As CNN and hierarchical architecture cannot accept variable-length inputs, MIMIR is not directly compatible with ConvNext and Swin. To adapt MIMIR to the hierarchical Swin Transformer, we implemented Masked Image Modeling using Group Window Attention [38], which groups image patches within each local window of arbitrary size and performs masked self-attention in each group. To apply MIMIR to ConvNext, we use SparK [40] for CNN to handle irregular and randomly masked input images, which is achieved by sparse convolution. MIMIR achieves better or comparable results compared to SOTA results on RobustBench [78].

MIMIR against Unforeseen Attacks. Except for adversaries who are limited by the adversarial budget, e.g., $l_\infty = 8$ or 4 or from PGD-family in our main experiments, MIMIR also shows the potential to provide robustness against unforeseen attacks and naturally corrupted data (ImageNet-C [42]). Table V demonstrates the robustness of MIMIR against practical unforeseen attacks, including non-PGD attacks (l_2 and l_∞ CW attack [76]), l_2 AutoAttack, and ImageNet-C [42]. It is clear that MIMIR still performs well against these unforeseen attacks. In particular, MIMIR achieved top accuracy compared to the results of the ImageNet-C Leaderboard on the RobustBench.

C. Ablation Study and Further Analysis

Step by step ablation. Table VII provides an ablation study to verify the design choices of MIMIR. The ablation uses 100 epochs of 1-step PGD (PGD$_1$) AT as the baseline. Then, we apply end-to-end clean ImageNet-1K pre-training (weights available in `timm` library[5]) as initialization of AT. After that, we replace the clean pre-training with MAE, adv MAE, and MIMIR step by step. The adv MAE refers to using adversarial examples but not using the MI $I(x + \delta, z)$ in the loss. The pre-training schedule is 800 epochs. We also use stronger adversarial fine-tuning for better performance, including 2-step PGD (PGD$_2$), APGD (APGD$_2$) FT, and a longer fine-tuning scheduler (300 epochs). Our results indicate that MIMIR outperforms baselines and can be further improved under the long training schedule.

Longer epochs (lower loss) provide better performance. *Pre-training epoch.* Our experimental framework employs an 800-epoch pre-training as the baseline configuration. To systematically evaluate the impact of training duration, we conduct a comprehensive ablation study using ViT-S architectures, varying the number of pre-training epochs while maintaining a fixed 50-epoch fine-tuning for all models. As demonstrated in Figure 3, we observe several key phenomena. Extended pre-training schedules consistently yield lower MIMIR loss values, and this reduction in loss results in measurable improvements in both adversarial and natural accuracy. In addition, the improved performance with loss MIMIR loss also indicates that the model capacity is not saturated within the tested epoch range and could be further improved with a larger number of epochs.

[5] https://github.com/huggingface/pytorch-image-models/blob/main/timm/models/vision_transformer.py

TABLE V
THE PERFORMANCE OF MIMIR ON IMAGENET-1K AGAINST UNFORESEEN THREATS. THE l_2 VERSION OF THE CW [76] ATTACK IS LIMITED BY c. HIGHER c ALLOWS A MORE POWERFUL PERTURBATION.

Architecture	Method	Natural	CW ($l_2, c = 1$)	CW ($l_\infty, \epsilon = 4/255$)	AA ($l_2, \epsilon = 2.0$)	AA ($l_\infty, \epsilon = 4/255$)	ImageNet-C [42]
ViT-S	[17]	69.22	44.28	45.98	37.52	44.04	45.62
	MIMIR	**70.96**	**66.34**	**49.22**	**49.12**	**46.16**	**48.78**
ViT-B	[17]	74.10	59.42	52.76	52.12	50.30	53.69
	MIMIR	**76.32**	**65.30**	**56.82**	**56.72**	**54.28**	**57.07**

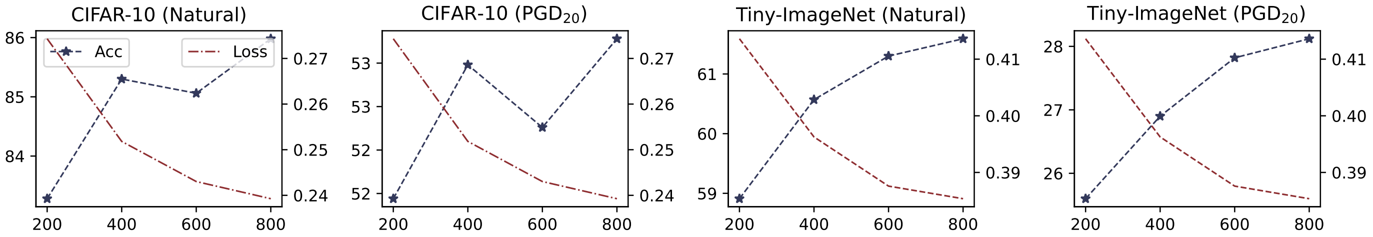

Fig. 3. Natural, adversarial accuracy, and MIMIR pre-training loss of ViT-S with different numbers of pre-training epochs under a 20-step PGD attack. The performance increases as the number of pre-training epochs increases (loss decreases).

TABLE VI
COMPARSION WITH SOTA IMAGENET-1K RESULTS ON ROBUSTBENCH [78] WITH DIFFERENT ARCHITECTURES. †: THE CVST MODULES ARE ALSO PRE-TRAINED WITH MIMIR.

Architecture	Method	FT Epoch	Natural	AutoAttack
ViT-S+CVST	[17]	300	72.56	48.08
	MIMIR	300	72.72	**48.44**
	MIMIR†	300	**73.02**	48.09
ViT-B+CVST	[17]	250	76.30	54.66
	MIMIR	300	**76.72**	54.04
	MIMIR†	300	76.32	**55.08**
ConvNext-T	[17]	300	72.40	48.60
	MIMIR	300	**72.50**	**48.76**
Swin-B	[33]	300	76.16	**56.16**
	MIMIR	150	**76.62**	55.90
Swin-L	[33]	300	**78.92**	59.56
	MIMIR	100	78.62	**59.68**

TABLE VII
ABLATION OF PRE-TRAINING (PT) AND FINE-TUNING (FT) METHODS ON IMAGENET-1K. (†: CATASTROPHIC OVER-FITTING [24] DUE TO 1-STEP AT WHEN FINE-TUNING, WHICH CAN BE FIXED BY 2-STEP AT. THE FIXED NATURAL AND ROBUST ACCURACY ARE 69.96 AND 36.90, RESPECTIVELY.)

Architecture	Training Recipe	Natural	AutoAttack
ViT-S	PGD$_1$ FT w/o PT	66.02	31.40
	clean PT + PGD$_1$ FT	67.04	33.70
	MAE PT + PGD$_1$ FT	69.98	35.64
	adv MAE PT + PGD$_1$ FT	68.24	19.32†
	MIMIR PT + PGD$_1$ FT	71.02	37.22
	MIMIR PT + PGD$_2$ FT	70.78	38.16
	MIMIR PT + APGD$_2$ FT	68.78	42.86
	100 → 300 epochs of FT	71.00	46.10

Fine-tuning epoch. To isolate and quantify the contribution of MIMIR pre-training to model performance, we employ a short fine-tuning for the pre-trained models. This is to train the randomly initialized classification layer since we do not have the classification layer at pre-training. This approach allows us to evaluate the quality of the representations learned during pre-training. In Figure 5, we show that MIMIR pre-training plus 5 or 10 epochs of fine-tuning is enough to achieve similar performance compared to 100-epoch fine-tuning. These results suggest that the majority of the model's final performance is attributable to the MIMIR pre-training phase.

MI measure. In Section III-D, we provide lower and upper bound (Eq. (14)) of $I(x+\delta, z)$. According to the two bounds, $I(x+\delta, z)$ is supposed to decrease while the autoencoder learns to reconstruct the natural image x. This motivates us to directly embed $I(x+\delta, z)$ as a minimizing learning objective. In this paper, we use I_α [48] and HSIC [80] as estimators (detailed definitions in Appendix J). Table VIII demonstrates the performance with different values of λ. According to the results, we use HSIC with $\lambda = 1e - 05$ for all other experiments.

In addition, Figure 4 provides the quantities of HSIC values while pre-training with or without using MIMIR. It is clear that MIMIR can help to decrease the mutual information between adversarial perturbation and the learned features, i.e., $I(x+\delta, z)$.

1-step is better than the 10-step of AT in pre-training. We also show that MIMIR outperforms original MAE [37] and adv MAE with different PGD steps (to generate adversarial examples for training). MAE in Table IX refers to using the original MAE for pre-training and then fine-tuning with 10-step PGD. The adv MAE refers to using adversarial examples without the MI $I(x+\delta, z)$ in the loss. The adv MAE (10-steps) refers to using the 10-step PGD algorithm ($\epsilon = 8, \alpha = 2$) to generate adversarial examples at pre-training. The adv MAE provides higher accuracy than MAE, which supports our statement that using adversarial examples in Masked

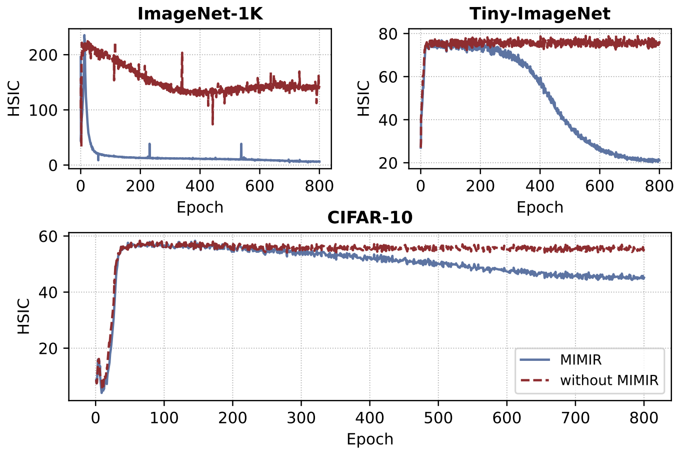

Fig. 4. The HSIC values (we use HSIC as an alternative to MI) while pre-training ViT-S with and without using MIMIR.

TABLE VIII
COMPARISON BETWEEN HSIC [51] AND I_α [48] USING VIT-T ON CIFAR-10. MODELS ARE PRE-TRAINED 800 EPOCHS AND ADVERSARIALLY FINE-TUNED WITH 10-STEP PGD FOR 50 EPOCHS.

Pre-train	λ	Estimator	Natural	PGD
MIMIR	0.001	HSIC	69.63	43.17
MIMIR	0.001	I_α	75.00	46.11
MIMIR	1e-05	HSIC	**76.30**	**47.60**
MIMIR	1e-05	I_α	75.53	46.75
MIMIR	1e-06	HSIC	74.90	46.19
MIMIR	1e-06	I_α	74.60	45.66

Image Modeling creates a more difficult reconstruction task. This more difficult task further improves the performance of downstream models (see also Table XII). We use the default learning rate (i.e., $5.0e-4$) of MAE, so there is a performance drop in experiments in Tables VIII and IX since AT prefers larger learning rates for CIFAR-10, as shown in Table XVII in the appendix.

Data augmentation is not always harmful. Prior research [9], [20] has established that strong data augmentation techniques can adversely affect ViTs during adversarial training, as they may make training samples challenging to learn. However, we observe that strong data augmentation does not impair model performance when combined with longer fine-tuning periods.

The strong data augmentation refers to the combination of Randaugment [28], CutMix [29], and MixUp [30]. In this section, we evaluate two different solutions to ease this problem. First, we only use simple data augmentation for adversarial training, including random crop (or random resize crop for ImageNet-1K) and random horizontal flip ("weak aug"). Second, we use a 10-epoch warmup procedure for strong data augmentation. The warmup of Randaugment is implemented by progressively increasing the distortion magnitude from 1 to 9 ("warmup aug"). For CutMix and MixUp, we warm up by increasing the mixup probability from 0.5 to 1.0. As shown in Figure 6, "weak aug" provides the best accuracy. The "warmup aug" shows a slightly improved accuracy compared to fusing strong augmentation. Therefore, we provide a different result from [9] on the smaller dataset CIFAR-10, i.e., we show that weak augmentation is better than warmup augmentation. Even data augmentation with reduced amplitude is still difficult to learn at the beginning of adversarial training. Although strong augmentation is harmful to a normal training schedule, we show in Table X that CutMix [29], MixUp [30], and Randaugment [28] increase the accuracy of adversarial training when training with a longer schedule, e.g., 800 epochs of fine-tuning. We conjecture that combining data and strong augmentations is helpful but difficult for adversarial training to learn. Thus, more epochs are needed to learn meaningful representation. Loss and accuracy curves can be found in Appendix E.

TABLE IX
COMPARISON BETWEEN DIFFERENT PRE-TRAINING SETTINGS. ALL MODELS ARE PRE-TRAINED FOR 800 EPOCHS AND THEN FINE-TUNED WITH 10-STEP PGD FOR 50 EPOCHS USING VIT-T ON CIFAR-10.

Pre-train	λ	Estimator	Natural	PGD
MAE	0.0	-	69.02	42.31
adv MAE (1-step)	0.0	-	74.69	46.28
adv MAE (10-step)	0.0	-	73.96	45.77
MIMIR	1e-05	HSIC	**76.30**	**47.60**

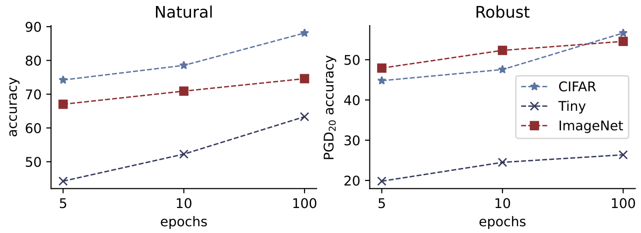

Fig. 5. Natural and adversarial accuracy of ViT-S adversarially fine-tuned for 5 or 10 epochs on CIFAR-10, Tiny-ImageNet, and ImageNet-1K.

D. Adaptive Attacks

We evaluate MIMIR against adaptive adversaries following common practices [81]. Adaptive adversaries possess the capability to devise targeted attacks specifically tailored to exploit the mechanisms of MIMIR, particularly if they have prior knowledge of its architecture and defensive strategies. For example, the adversary may attack feature space [82], [83] since MIMIR trains the backbone to extract robust features. Here, the backbone refers to the ViT model without the classification layer, i.e., the encoder of MIMIR.

We provide two adaptive attacks specifically designed against MIMIR. First, we introduce the PGD Mutual Information attack (PGD-MI), which utilizes the MI $I(x+\delta, z)$ to generate adversarial examples, as $I(x+\delta, z)$ is used in MIMIR pre-training as a penalty in the loss. PGD-MI attacks the model by directly increasing the MI $I(x+\delta, z)$. Specifically, we add the MI loss into the PGD algorithm:

$$\max_{\delta \in S} \mathcal{L}_{CE}(x_i + \delta, y_i) + \lambda I(x + \delta, z). \tag{21}$$

TABLE X
DATA AUGMENTATION WITH LONGER FINE-TUNING SCHEDULE.

Arch	Epoch	Augmentation	Natural	PGD$_{20}$
ViT-B	800	Weak Augmentation	89.90	60.26
		+ CutMix [29],MixUp [30]	91.01	60.62
		+ Randaugment [28]	90.19	62.75

TABLE XI
ADVERSARIAL ACCURACY BY ADAPTIVE ATTACKS. THE MODELS ARE PRE-TRAINED FOR 800 EPOCHS BY MIMIR AND FINE-TUNED FOR 100 EPOCHS BY 1-STEP PGD AT.

Dataset	Model	PGD$_{20}$	PGD-MI$_{100}$	PGD-fea$_{100}$
CIFAR-10	ConViT-S	56.35	56.16	78.52
	ViT-S	56.63	56.31	78.41
	ViT-B	58.14	57.85	80.49
Tiny-ImageNet	ConViT-S	26.39	26.29	58.50
	ViT-S	26.37	26.18	57.36
	ViT-B	25.41	25.05	58.90
ImageNet-1K	ConViT-S	53.86	53.84	72.10
	ViT-S	54.56	54.55	72.27
	ViT-B	55.41	55.36	73.51

where the value of λ in MIMIR pre-training is available to adversaries. Second, we introduce a PGD feature attack (PGD-fea) that directly attacks the feature extracted by ViT backbones following [82]. In particular, we attack the feature extractor from the backbones after the adversarial fine-tuning. The PGD-fea attack increases the Euclidean distance between features extracted from natural and adversarial examples. We implement it using the PGD algorithm:

$$\max_{\delta \in S} \mathcal{L}_{\text{mse}}(f_e(x), f_e(x+\delta)). \tag{22}$$

Both PGD-MI and PGD-fea are optimized for 100 steps to ensure the attacking algorithm converges. The perturbation budget is the same as the previous evaluation, i.e., $\epsilon = 8/255$ for CIFAR-10 and Tiny-ImageNet, and $\epsilon = 4/255$ for ImageNet-1K.

Table XI demonstrates the adaptive evaluation results for PGD-MI and PGD-fea attacks. PGD-MI performs slightly better than the standard PGD attack, which means MI is exploitable information for perturbation crafting, but cannot significantly reduce the robustness. Furthermore, MIMIR-trained models are converged to a local optimal where the majority of predictions are constantly around the ground truth within the ball function of ϵ. This also explains the resilience against both PGD and PGD-MI variants. Regarding PGD-fea, it aims to maximize feature-space divergence rather than the distance of output logits. However, MIMIR learned robust features that cannot be easily separated, so PGD-fea performs worse than PGD and PGD-MI. The collective results demonstrate that MIMIR's IB framework induces robust learning dynamics that resist both output-space and feature-space attacks.

E. Fine-tuning with Natural Images

In Table XII, we show the results of MIMIR when fine-tuning with natural images. We compared the performance with MAE [37]. We fine-tune for 50 epochs for CIFAR-10 and Tiny-ImageNet, and 100 epochs for ImageNet-1K. The results in Table XII are reported with 800 pre-training epochs. The base learning rate used in Table XII is 0.001. The fine-tuning batch size is 512 for CIFAR-10 and Tiny-ImageNet, and 1024 for ImageNet-1K. We use weak data augmentation ("weak aug"), which includes random crop and random horizontal flip. Not surprisingly, MIMIR outperforms MAE when fine-tuning with natural data. This is because MIMIR creates a harder learning task, which is helpful to learn more discriminative representation, as we discussed in Section III-B, Design Intuition.

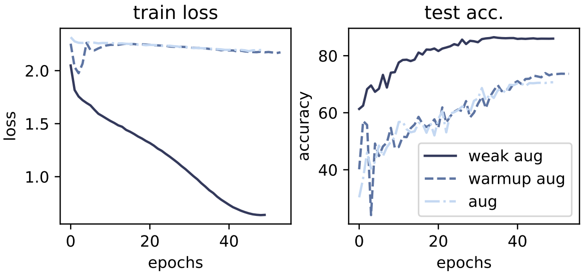

Fig. 6. Training loss and natural accuracy of ViT-S with three different data augmentations on CIFAR-10.

TABLE XII
NATURAL ACCURACY OF MAE AND MIMIR (800 EPOCHS PRE-TRAINING FOR BOTH) THAT ARE FINE-TUNED ON NATURAL IMAGES.

Architecture	Pre-train	CIFAR-10	Tiny-ImageNet	ImageNet-1K
ViT-B	MAE	96.79	73.38	82.92
	MIMIR	**96.91**	**75.43**	**83.20**
ConViT-S	MAE	94.95	69.03	78.37
	MIMIR	**95.38**	**70.40**	**79.21**
ViT-S	MAE	95.95	70.00	77.45
	MIMIR	95.95	**71.14**	**78.69**

According to Tables XII and XIII, MIMIR consistently shows improved performance on natural data. Although the models in Table XIII show poor robustness due to fine-tuning on natural data, MIMIR pre-trained ones provide slightly better robustness. We want to clarify that poor robustness is expected when fine-tuning with natural data. First, it is known that standard training on natural data learns non-robust features [84], which hurts performance under adversarial attacks. Second, MIMIR pre-training is implemented using MSE loss plus an MI penalty between natural inputs and adversarial images. The adversarial perturbations and MI penalty help MIMIR create a more difficult and discriminative learning task to learn meaningful and robust features. This process does not include the classification layer of the final model. Therefore, MIMIR needs a fine-tuning process for superior performance on natural data and adversarial inputs. In other words, the superior performance of our experiments comes from the combination of MIMIR and the simple fine-tuning process.

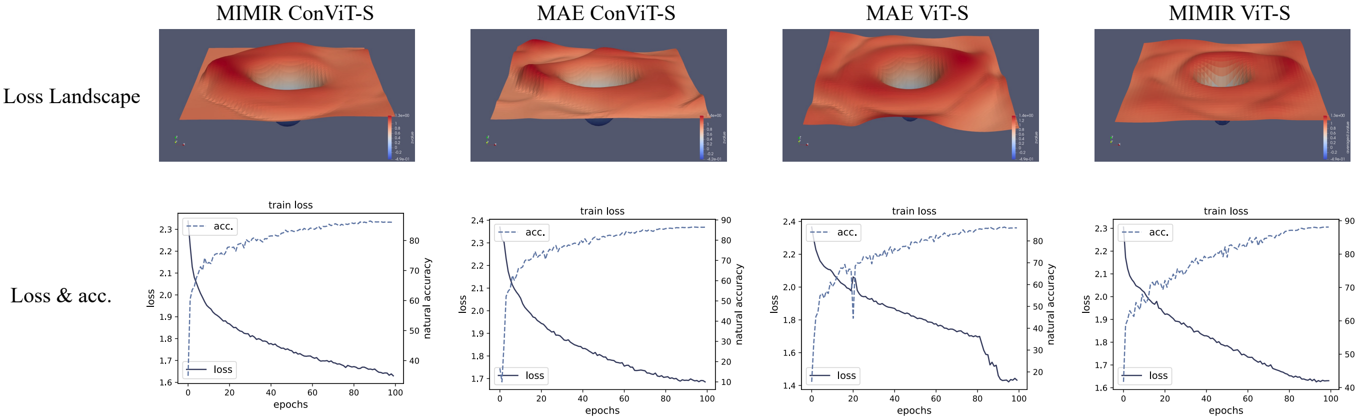

Fig. 7. The loss landscapes of MIMIR and MAE pre-trained models.

TABLE XIII
NATURAL AND ADVERSARIAL ACCURACY OF VIT-S THAT IS FINE-TUNED FOR 5 OR 50 EPOCHS WITH NATURAL IMAGES.

Fine-tune	Dataset	Pre-train	Natural	PGD
5 epochs	CIFAR-10	MIMIR	93.22	0.55
		MAE	93.16	0.01
	Tiny-ImageNet	MIMIR	63.65	0.00
		MAE	62.21	0.00
	ImageNet-1K	MIMIR	72.45	0.20
		MAE	69.47	0.02
50 epochs	CIFAR-10	MIMIR	95.95	0.28
		MAE	95.95	0.29
	Tiny-ImageNet	MIMIR	71.14	0.00
		MAE	70.00	0.00
	ImageNet-1K	MIMIR	78.69	0.18
		MAE	77.45	0.05

F. Visualization of the Loss Landscape

To show that the robustness of MIMIR-trained models does not stem from gradient masking, we plot the loss landscape [85] in Figure 7. The loss landscape is the visualization of the loss function as parameters change. The basic idea is to plot the loss around the optimal parameters. Formally, we consider in the 2D case,

$$f_l(\alpha, \beta) = L(\theta^* + \alpha\theta_1 + \beta\theta_2), \qquad (23)$$

where θ_1 and θ_2 are two direction vectors, α and β are two arguments of f_l. In practice, we use the parameters of trained models, i.e., θ^*. The landscapes of all models are smooth, i.e., the gradient at a certain point is clear and can also be easily estimated by local average gradients, which means the gradient is masked.

V. RELATED WORK

A. Vision Transformer

The transformers [86] were first proposed in natural language processing (NLP). With the mechanism of global self-attention, transformers can effectively capture the non-local relationships among all text tokens [87], [88], [89]. A substantial effort is made to apply the transformer and self-attention mechanism in computer vision [3], [90], [5]. The pioneering work, ViT [3], demonstrated that the pure transformer architecture could achieve competitive performance on various tasks. ViT also reveals that transformers lack inductive biases [3]. For example, locality, two-dimensional neighborhood structure, and translation equivariance are inherent to CNNs but not applicable to ViTs [3]. Due to this shortcoming, ViTs usually require large-scale training to get competitive performance, such as pre-training on ImageNet-21K [36] and JFT-300M [91]. To alleviate the ViT need for large datasets, DeiT [92] introduced a teacher-student strategy to distill knowledge from a teacher CNN for a student ViT. In the MIM field, MAE [37] uses a masked autoencoder with a lightweight decoder as a visual representation learner. Its learning objective is to reconstruct the original image by the decoder while using masked images as input to the autoencoder. The advantage is that MAE can randomly discard 75% image patches when pre-training under ImageNet-1K [36], which means more efficient training.

B. Adversarial Attacks on ViTs

The concept of adversarial attacks first appeared in [43], which proposed a formal framework and algorithms against the adversarial spam detection domain. Then, the adversarial attacks were popularized by Biggio et al. [14] and Szegedy et al. [15] in image classification. The generation of adversarial examples depends on the model's gradient or estimated gradient in a black box situation [93]. Therefore, adversarial attacks can be easily applied to transformers by using the gradient of attention blocks concerning inputs. This also raises the question of whether transformers are more robust than CNNs. Benz et al. [94] found that CNNs are less robust than ViTs due to their shift-invariant property. Bhojanapalli et al. [95] found that ResNet models are more robust than transformers at the same model size under FGSM attack, but under PGD [21] attack, transformer models show better robustness. As ViTs process the input image as a sequence of patches, Gu et al. [13] found that ViTs are more robust than CNNs to naturally corrupted patches because the attention mechanism helps ignore naturally corrupted image patches. The later work [9] revealed that CNNs could be as robust as ViTs against adversarial attacks if CNNs are trained with proper hyperparameters.

C. Adversarial Defense

PGD [21] adversarial training is considered one of the most effective defenses for CNNs and can withstand adaptively designed attacks [96]. However, PGD AT is harmful to the accuracy of clean data [21], [26], [2]. Generalist [1] solves this problem by formulating different training strategies for robust and natural generalization separately. DBAT [2] solves the decrease in natural accuracy by adding dummy classes [97] to the classification space.

Due to the difference between CNNs and ViTs, there have been some recent efforts to explore new adversarial training approaches for ViTs [11], [20], [12]. Mo et al. [11] presented a new adversarial training strategy based on the following observations: 1) pre-training with natural data can provide better robustness after adversarial fine-tuning, 2) gradient clipping is necessary for adversarial training, and 3) using SGD as the optimizer is better than Adam. Debenedetti et al. [20] also presented an improved training strategy for ViTs by evaluating different combinations of data augmentation policies. As adversarial training is time-consuming, AGAT [12] leverages the attention score while training to discard non-critical image patches after every layer. Unlike previous works, we provide a different training paradigm by using MIM for adversarial pre-training. Our method is efficient as we discard 75% image patches while pre-training. Our method is effective as we eliminate the information of adversarial perturbations from two information sources of natural and adversarial inputs. We also provide theoretical proof that the information of adversarial perturbations is eliminated.

D. Self-Supervised Adversarial Pre-Training

Self-supervised learning [98], [99], [37], [100] refers to extracting meaningful representation from unlabeled data, which can be used for downstream recognition tasks. Self-supervised methods are beneficial for out-of-distribution detection on difficult, near-distribution outliers [27], which leads to using self-supervised training to improve adversarial robustness [101], [27], [102], [103], [104], [105]. The basic idea is to build a min-max learning object similar to traditional adversarial training. For example, Jiang et al. [102] considered using two adversarial samples or combining one adversarial sample and one natural sample to learn a consistent representation in contrastive learning. In more recent work, You et al. [106] proposed NIM De^3 to denoise adversarial perturbations. However, the motivation of these works relies on complex self-supervised pre-training technologies, making it more difficult to understand the inner mechanisms or provide theoretical results. MIMIR not only provides better performance but also provides intuitive insights with theoretical motivation.

VI. Discussion and Limitations

Following the principle of IB, we can intuitively consider a bottleneck between the encoder and decoder. As the reconstruction output is constrained by natural data x, the bottleneck will filter out information from adversarial perturbations δ. We provide a theoretical guarantee of this bottleneck. In Eq. (8), we embed this bottleneck as a learning object to further improve the performance, which also confirms the correctness of our theoretical guarantee. With the two information sources of x and δ, the model is trained to learn the robust features from x and forget the information of δ under the constraint of the reconstruction target.

While MIMIR shows better performance, there are still limitations. MIMIR is a pre-training method. Adversarial fine-tuning is necessary to build the final robust model. Thus, the shortcomings of traditional adversarial training cannot be completely avoided. In our experiments, we utilize the simple PGD algorithm for fine-tuning, but one can further improve MIMIR pre-trained models with more advanced approaches. In addition, MIMIR follows the design of MAE, and we also utilize the characteristic that ViTs can process variable-length inputs. Therefore, the current MIMIR cannot directly handle CNNs. While it is not trivial, we apply MIMIR to the latest CNN architecture by sparse convolution from SparK [40]. However, sparse convolution is not as efficient as dropping patch embeddings. We leave these limitations to future work.

VII. Conclusions

This paper provides a novel theoretical analysis of AT for ViTs through the lens of IB. We found that constraining the MI between adversarial perturbations and their latent representations in ViT-based autoencoders, as governed by derived MI bounds, is critical for enhancing model robustness. Building upon this theoretical foundation, we propose MIMIR as a theoretically grounded pre-training method to improve adversarial robustness for ViTs. MIMIR operates by processing adversarial examples as inputs while reconstructing their natural data as targets. This approach leverages the inherent information bottleneck in autoencoder architectures to achieve two key objectives: (1) progressively eliminating perturbation-related information while (2) preserving the essential features of the original data distribution. Our extensive experimental evaluation demonstrates that MIMIR significantly outperforms existing adversarial training methods across multiple benchmark datasets, achieving SOTA results on ImageNet-1K. In addition, MIMIR is robust against unforeseen attacks and common corrupted data and can resist adaptive attacks.

Acknowledgment

The authors thank the anonymous reviewers for their constructive comments that helped improve this paper and the artifact. This work used the Dutch national e-infrastructure with the support of the SURF Cooperative using grant no. EINF-10853. This work was partially supported by the Horizon Europe programme under the project SHASAI (No. 101225866).

REFERENCES

[1] H. Wang and Y. Wang, "Generalist: Decoupling natural and robust generalization," in *CVPR*, 2023.

[2] M. Levi and A. Kontorovich, "Splitting the difference on adversarial training," in *USENIX Security*, 2024.

[3] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, "An image is worth 16x16 words: Transformers for image recognition at scale," in *ICLR*, 2021.

[4] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo, "Swin transformer: Hierarchical vision transformer using shifted windows," in *ICCV*, 2021.

[5] S. D'Ascoli, H. Touvron, M. L. Leavitt, A. S. Morcos, G. Biroli, and L. Sagun, "Convit: Improving vision transformers with soft convolutional inductive biases," in *ICML*, 2021.

[6] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, G. Krueger, and I. Sutskever, "Learning transferable visual models from natural language supervision," in *ICML*, 2021.

[7] J. Li, D. Li, C. Xiong, and S. Hoi, "BLIP: Bootstrapping language-image pre-training for unified vision-language understanding and generation," in *ICML*, 2022.

[8] D. Zhu, J. Chen, X. Shen, X. Li, and M. Elhoseiny, "MiniGPT-4: Enhancing vision-language understanding with advanced large language models," in *ICLR*, 2024.

[9] Y. Bai, J. Mei, A. L. Yuille, and C. Xie, "Are transformers more robust than cnns?" in *NeurIPS*, 2021.

[10] A. Aldahdooh, W. Hamidouche, and O. Deforges, "Reveal of vision transformers robustness against adversarial attacks," *arXiv preprint arXiv:2106.03734*, 2021.

[11] Y. Mo, D. Wu, Y. Wang, Y. Guo, and Y. Wang, "When adversarial training meets vision transformers: Recipes from training to architecture," in *NeurIPS*, 2022.

[12] B. Wu, J. Gu, Z. Li, D. Cai, X. He, and W. Liu, "Towards efficient adversarial training on vision transformers," in *ECCV*, 2022.

[13] J. Gu, V. Tresp, and Y. Qin, "Are vision transformers robust to patch perturbations?" in *ECCV*, 2022.

[14] B. Biggio, I. Corona, D. Maiorca, B. Nelson, N. Šrndić, P. Laskov, G. Giacinto, and F. Roli, "Evasion attacks against machine learning at test time," in *ECML PKDD*, 2013.

[15] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus, "Intriguing properties of neural networks," in *ICLR*, 2014.

[16] A. Vassilev, A. Oprea, A. Fordyce, and H. Anderson, "Adversarial machine learning: A taxonomy and terminology of attacks and mitigations," National Institute of Standards and Technology (NIST), Tech. Rep., 2024.

[17] N. D. Singh, F. Croce, and M. Hein, "Revisiting adversarial training for imagenet: Architectures, training and generalization across threat models," in *NeurIPS*, 2023.

[18] S. Peng, W. Xu, C. Cornelius, K. Li, R. Duggal, D. H. Chau, and J. Martin, "Robarch: Designing robust architectures against adversarial attacks," 2023.

[19] Y. Bai, M. Zhou, V. M. Patel, and S. Sojoudi, "MixedNUTS: Training-free accuracy-robustness balance via nonlinearly mixed classifiers," *TMLR*, 2024.

[20] E. Debenedetti, V. Sehwag, and P. Mittal, "A light recipe to train robust vision transformers," in *SaTML*, 2023.

[21] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards deep learning models resistant to adversarial attacks," in *ICLR*, 2018.

[22] A. Raghunathan, S. M. Xie, F. Yang, J. C. Duchi, and P. Liang, "Adversarial training can hurt generalization," *arXiv preprint arXiv:1906.06032*, 2019.

[23] A. Shafahi, M. Najibi, M. A. Ghiasi, Z. Xu, J. Dickerson, C. Studer, L. S. Davis, G. Taylor, and T. Goldstein, "Adversarial training for free!" in *NeurIPS*, 2019.

[24] E. Wong, L. Rice, and J. Z. Kolter, "Fast is better than free: Revisiting adversarial training," in *ICLR*, 2020.

[25] H. Zhang, Y. Yu, J. Jiao, E. Xing, L. E. Ghaoui, and M. Jordan, "Theoretically principled trade-off between robustness and accuracy," in *ICML*, 2019.

[26] T. Pang, M. Lin, X. Yang, J. Zhu, and S. Yan, "Robustness and accuracy could be reconcilable by (proper) definition," in *ICML*, 2022.

[27] D. Hendrycks, M. Mazeika, S. Kadavath, and D. Song, "Using self-supervised learning can improve model robustness and uncertainty," in *NeurIPS*, 2019.

[28] E. D. Cubuk, B. Zoph, J. Shlens, and Q. Le, "Randaugment: Practical automated data augmentation with a reduced search space," in *NeurIPS*, 2020.

[29] S. Yun, D. Han, S. J. Oh, S. Chun, J. Choe, and Y. Yoo, "Cutmix: Regularization strategy to train strong classifiers with localizable features," in *ICCV*, 2019.

[30] H. Zhang, M. Cissé, Y. N. Dauphin, and D. Lopez-Paz, "mixup: Beyond empirical risk minimization," in *ICLR*, 2018.

[31] Z. Wang, X. Li, H. Zhu, and C. Xie, "Revisiting adversarial training at scale," *CVPR*, 2024.

[32] Z. Xie, Z. Zhang, Y. Cao, Y. Lin, J. Bao, Z. Yao, Q. Dai, and H. Hu, "Simmim: A simple framework for masked image modeling," in *CVPR*, 2022.

[33] C. Liu, Y. Dong, W. Xiang, X. Yang, H. Su, J. Zhu, Y. Chen, Y. He, H. Xue, and S. Zheng, "A comprehensive study on robustness of image classification models: Benchmarking and rethinking," *IJCV*, 2024.

[34] A. Krizhevsky, G. Hinton *et al.*, "Learning multiple layers of features from tiny images," 2009.

[35] Y. Le and X. Yang, "Tiny imagenet visual recognition challenge," *CS 231N*, vol. 7, no. 7, p. 3, 2015.

[36] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, 2009.

[37] K. He, X. Chen, S. Xie, Y. Li, P. Dollár, and R. Girshick, "Masked autoencoders are scalable vision learners," in *CVPR*, 2022.

[38] L. Huang, S. You, M. Zheng, F. Wang, C. Qian, and T. Yamasaki, "Green hierarchical vision transformer for masked image modeling," in *NeurIPS*, 2022.

[39] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo, "Swin transformer: Hierarchical vision transformer using shifted windows," in *ICCV*, 2021.

[40] K. Tian, Y. Jiang, qishuai diao, C. Lin, L. Wang, and Z. Yuan, "Designing BERT for convolutional networks: Sparse and hierarchical masked modeling," in *ICLR*, 2023.

[41] Z. Liu, H. Mao, C.-Y. Wu, C. Feichtenhofer, T. Darrell, and S. Xie, "A convnet for the 2020s," in *CVPR*, 2022.

[42] D. Hendrycks and T. Dietterich, "Benchmarking neural network robustness to common corruptions and perturbations," *ICLR*, 2019.

[43] N. Dalvi, P. Domingos, Mausam, S. Sanghai, and D. Verma, "Adversarial classification," in *KDD*, 2004.

[44] H. Bao, L. Dong, S. Piao, and F. Wei, "BEit: BERT pre-training of image transformers," in *ICLR*, 2022.

[45] X. Li, W. Wang, L. Yang, and J. Yang, "Uniform masking: Enabling mae pre-training for pyramid-based vision transformers with locality," *arXiv:2205.10063*, 2022.

[46] C. Wei, H. Fan, S. Xie, C.-Y. Wu, A. Yuille, and C. Feichtenhofer, "Masked feature prediction for self-supervised visual pre-training," in *CVPR*, 2022.

[47] Z. Goldfeld and Y. Polyanskiy, "The information bottleneck problem and its applications in machine learning," *IEEE Journal on Selected Areas in Information Theory*, vol. 1, no. 1, pp. 19–38, 2020.

[48] X. Yu, S. Yu, and J. C. Príncipe, "Deep deterministic information bottleneck with matrix-based entropy functional," in *ICASSP*, 2021.

[49] L. G. Sanchez Giraldo, M. Rao, and J. C. Principe, "Measures of entropy from data using infinitely divisible kernels," *IEEE Transactions on Information Theory*, vol. 61, no. 1, pp. 535–548, 2015.

[50] S. Yu, L. G. S. Giraldo, R. Jenssen, and J. C. Principe, "Multivariate extension of matrix-based renyi's alpha-order entropy functional," *IEEE transactions on pattern analysis and machine intelligence*, vol. 42, no. 11, pp. 2960–2966, 2019.

[51] A. Gretton, O. Bousquet, A. Smola, and B. Schölkopf, "Measuring statistical dependence with hilbert-schmidt norms," in *Algorithmic Learning Theory*, 2005.

[52] N. Tishby, F. C. N. Pereira, and W. Bialek, "The information bottleneck method," *CoRR*, vol. physics/0004057, 2000.

[53] N. Tishby and N. Zaslavsky, "Deep learning and the information bottleneck principle," in *2015 IEEE Information Theory Workshop (ITW)*, 2015.

[54] R. Shwartz-Ziv and N. Tishby, "Opening the black box of deep neural networks via information," *CoRR*, vol. abs/1703.00810, 2017.

[55] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, P.-A. Manzagol, and L. Bottou, “Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion.” *Journal of machine learning research*, vol. 11, no. 110, pp. 3371–3408, 2010.

[56] H. Wang, K. Song, J. Fan, Y. Wang, J. Xie, and Z. Zhang, “Hard patches mining for masked image modeling,” in *CVPR*, 2023.

[57] N. J. Beaudry and R. Renner, “An intuitive proof of the data processing inequality,” *arXiv preprint arXiv:1107.0740*, 2011.

[58] R. M. Fano, *The transmission of information*. Massachusetts Institute of Technology, Research Laboratory of Electronics . . . , 1949, vol. 65.

[59] O. Ocal, O. H. Elibol, G. Keskin, C. Stephenson, A. Thomas, and K. Ramchandran, “Adversarially trained autoencoders for parallel-data-free voice conversion,” in *ICASSP*, 2019.

[60] M. Hellman and J. Raviv, “Probability of error, equivocation, and the chernoff bound,” *IEEE Transactions on Information Theory*, vol. 16, no. 4, pp. 368–372, 1970.

[61] G. Brown, “An information theoretic perspective on multiple classifier systems,” in *International Workshop on Multiple Classifier Systems*, 2009.

[62] S.-M. Moosavi-Dezfooli, A. Fawzi, O. Fawzi, and P. Frossard, “Universal adversarial perturbations,” in *CVPR*, 2017.

[63] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol, “Extracting and composing robust features with denoising autoencoders,” in *ICML*, 2008.

[64] J. Liu, X. Huang, J. Zheng, Y. Liu, and H. Li, “Mixmae: Mixed and masked autoencoder for efficient pretraining of hierarchical vision transformers,” in *CVPR*, 2023.

[65] I. Loshchilov and F. Hutter, “Decoupled weight decay regularization,” in *ICLR*, 2019.

[66] Y. Wang, D. Zou, J. Yi, J. Bailey, X. Ma, and Q. Gu, “Improving adversarial robustness requires revisiting misclassified examples,” in *ICLR*, 2019.

[67] X. Mao, Y. Chen, X. Li, G. Qi, R. Duan, R. Zhang, and H. Xue, “Easyrobust: A comprehensive and easy-to-use toolkit for robust computer vision,” https://github.com/alibaba/easyrobust, 2022.

[68] C. Francesco and M. Hein, “Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks,” in *ICML*, 2020.

[69] F. Croce and M. Hein, “Minimally distorted adversarial examples with a fast adaptive boundary attack,” in *ICML*, 2020.

[70] M. Andriushchenko, F. Croce, N. Flammarion, and M. Hein, “Square attack: A query-efficient black-box adversarial attack via random search,” in *ECCV*, 2020.

[71] Z. Wang, T. Pang, C. Du, M. Lin, W. Liu, and S. Yan, “Better diffusion models further improve adversarial training,” in *ICML*, 2023.

[72] S. Peng, W. Xu, C. Cornelius, M. Hull, K. Li, R. Duggal, M. Phute, J. Martin, and D. H. Chau, “Robust principles: Architectural design principles for adversarially robust cnns,” *arXiv preprint arXiv:2308.16258*, 2023.

[73] S.-A. Rebuffi, S. Gowal, D. A. Calian, F. Stimberg, O. Wiles, and T. Mann, “Data augmentation can improve robustness,” in *NeurIPS*, 2021.

[74] S. Gowal, S.-A. Rebuffi, O. Wiles, F. Stimberg, D. A. Calian, and T. A. Mann, “Improving robustness using generated data,” in *NeurIPS*, 2021.

[75] H. Zhu, B. Chen, and C. Yang, “Understanding why vit trains badly on small datasets: An intuitive perspective,” *arXiv preprint arXiv:2302.03751*, 2023.

[76] N. Carlini and D. Wagner, “Towards evaluating the robustness of neural networks,” in *SP*, 2017.

[77] S. Woo, S. Debnath, R. Hu, X. Chen, Z. Liu, I. S. Kweon, and S. Xie, “Convnext v2: Co-designing and scaling convnets with masked autoencoders,” in *CVPR*, 2023.

[78] F. Croce, M. Andriushchenko, V. Sehwag, E. Debenedetti, N. Flammarion, M. Chiang, P. Mittal, and M. Hein, “Robustbench: a standardized adversarial robustness benchmark,” in *NeurIPS Datasets and Benchmarks Track (Round 2)*, 2021.

[79] T. Xiao, M. Singh, E. Mintun, T. Darrell, P. Dollar, and R. Girshick, “Early convolutions help transformers see better,” in *NeurIPS*, 2021.

[80] W.-D. K. Ma, J. P. Lewis, and W. B. Kleijn, “The hsic bottleneck: Deep learning without back-propagation,” in *AAAI*, 2020.

[81] F. Tramer, N. Carlini, W. Brendel, and A. Madry, “On adaptive attacks to adversarial example defenses,” in *NeurIPS*, 2020.

[82] S. Sabour, Y. Cao, F. Faghri, and D. J. Fleet, “Adversarial manipulation of deep representations,” in *ICLR*, 2016.

[83] Z. Liu, Z. Zhao, and M. Larson, “Who’s afraid of adversarial queries? the impact of image modifications on content-based image retrieval,” in *ICMR*, 2019.

[84] A. Ilyas, S. Santurkar, D. Tsipras, L. Engstrom, B. Tran, and A. Madry, “Adversarial examples are not bugs, they are features,” in *NeurIPS*, 2019.

[85] H. Li, Z. Xu, G. Taylor, C. Studer, and T. Goldstein, “Visualizing the loss landscape of neural nets,” in *NeurIPS*, 2018.

[86] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin, “Attention is all you need,” in *NeurIPS*, 2017.

[87] J. Devlin, M. Chang, K. Lee, and K. Toutanova, “BERT: pre-training of deep bidirectional transformers for language understanding,” in *NAACL-HLT*, 2019.

[88] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei, “Language models are few-shot learners,” in *NeurIPS*, 2020.

[89] Z. Dai, Z. Yang, Y. Yang, J. G. Carbonell, Q. V. Le, and R. Salakhutdinov, “Transformer-xl: Attentive language models beyond a fixed-length context,” in *ACL*, 2019.

[90] H. Touvron, M. Cord, A. Sablayrolles, G. Synnaeve, and H. Jégou, “Going deeper with image transformers,” in *ICCV*, 2021.

[91] C. Sun, A. Shrivastava, S. Singh, and A. Gupta, “Revisiting unreasonable effectiveness of data in deep learning era,” in *ICCV*, 2017.

[92] H. Touvron, M. Cord, M. Douze, F. Massa, A. Sablayrolles, and H. Jegou, “Training data-efficient image transformers and distillation through attention,” in *ICML*, 2021.

[93] A. Ilyas, L. Engstrom, A. Athalye, and J. Lin, “Black-box adversarial attacks with limited queries and information,” in *ICML*, 2018.

[94] P. Benz, S. Ham, C. Zhang, A. Karjauv, and I. S. Kweon, “Adversarial robustness comparison of vision transformer and mlp-mixer to cnns,” in *BMVC*, 2021.

[95] S. Bhojanapalli, A. Chakrabarti, D. Glasner, D. Li, T. Unterthiner, and A. Veit, “Understanding robustness of transformers for image classification,” in *ICCV*, 2021.

[96] A. Athalye, N. Carlini, and D. Wagner, “Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples,” in *ICML*, 2018.

[97] B. Chen, W. Deng, and H. Shen, “Virtual class enhanced discriminative embedding learning,” in *NeurIPS*, 2018.

[98] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick, “Momentum contrast for unsupervised visual representation learning,” in *CVPR*, 2020.

[99] X. Chen and K. He, “Exploring simple siamese representation learning,” in *CVPR*, 2021.

[100] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, “A simple framework for contrastive learning of visual representations,” in *ICML*, 2020.

[101] T. Chen, S. Liu, S. Chang, Y. Cheng, L. Amini, and Z. Wang, “Adversarial robustness: From self-supervised pre-training to fine-tuning,” in *CVPR*, 2020.

[102] Z. Jiang, T. Chen, T. Chen, and Z. Wang, “Robust pre-training by adversarial contrastive learning,” in *NeurIPS*, 2020.

[103] L. Fan, S. Liu, P.-Y. Chen, G. Zhang, and C. Gan, “When does contrastive learning preserve adversarial robustness from pretraining to finetuning?” in *NeurIPS*, 2021.

[104] Q. Wu, H. Ye, Y. Gu, H. Zhang, L. Wang, and D. He, “Denoising masked autoencoders help robust classification,” in *ICLR*, 2023.

[105] S.-A. Rebuffi, O. Wiles, E. Shelhamer, and S. Gowal, “Adversarially self-supervised pre-training improves accuracy and robustness,” *ICLR 2023 Workshop DG Poster*, 2023.

[106] Z. You, D. Liu, and C. Xu, “Beyond pretrained features: Noisy image modeling provides adversarial defense,” *arXiv preprint arXiv:2302.01056*, 2023.

[107] G. Huang, Y. Sun, Z. Liu, D. Sedra, and K. Q. Weinberger, “Deep networks with stochastic depth,” in *ECCV*, 2016.

TABLE XIV
MODEL ARCHITECTURES OF THE ENCODER AND DECODER.

Model	Layers	Hidden size	MLP ratio	Heads
ViT-T (encoder)	12	192	4	3
ViT-S (encoder)	12	384	4	6
ViT-B (encoder)	12	768	4	12
decoder	2	128	4	16

TABLE XV
PRE-TRAINING HYPERPARAMETERS.

Config	Value
optimizer	AdamW
base learning rate	1.5e-4
weight decay	0.05
optimizer momentum	$\beta_1 = 0.9, \beta_2 = 0.95$
batch size	512(CIFAR-10, Tiny), 2,048 (ImageNet-1K)
learning rate schedule	cosine decay
warmup epochs	40
training epochs	800
augmentation	RandomResizedCrop, RandomHorizontalFlip

APPENDIX

A. Datasets

We use three commonly used datasets to evaluate MIMIR: CIFAR-10 [34], Tiny-ImageNet [35], and ImageNet-1K [36]. CIFAR-10 [34] comprises 50,000 images with size $3 \times 32 \times 32$ in 10 classes. ImageNet-1K [36] is the most commonly used dataset for the evaluation of ViTs and their variants, which is composed of more than 1.2 million high-resolution images in 1,000 classes. In our experiments, images from ImageNet-1K are resized to $3 \times 224 \times 224$. For completeness, we also include Tiny-ImageNet [35] as a medium size dataset between CIFAR-10 [34] and ImageNet-1K [36]. Tiny-ImageNet [35] contains 100,000 images with size $3 \times 64 \times 64$ in 200 classes.

B. Decoder Hyperparameters

We use transformer blocks but fewer layers as the backbone of the decoder. For CIFAR-10, we use the patch size of 2, 4 for Tiny-ImageNet, and 16 for ImageNet-1K. Table XIV shows the hyperparameters of decoder architectures. For different ViT architectures, we use the transformer blocks of the respective architectures to build the encoder.

C. Details of Training Hyperparameters

In Tables XV and XVI, we provide the default hyperparameters used in our experiments. We use different patch sizes for different datasets: patch size 2 for CIFAR-10, 4 for Tiny-ImageNet, and 16 for ImageNet-1K. Using smaller patch sizes increases the time consumption when calculating self-attention, but MIMIR pre-training discards 75% patches, making it still efficient. Due to the depth and comparatively small embedding size of CaiT, we use a different drop path and layer-wise decay when fine-tuning (for ImageNet-1K). For CaiT-XXS24, we use 0.95 and 0.15 as layer-wise decay and dropout, and 0.85 and 0.35 for CaiT-S36. We also apply the stochastic depth decay rule [107] to CaiT. CaiT-S36 models are only fine-tuned for 50 epochs due to time consumption, and it is sufficient to get superior results. The batch size to fine-tune CaiT is 512 due to the limitation of GPU memory. Other hyperparameters are consistent with Tables XV and XVI.

TABLE XVI
FINE-TUNING HYPERPARAMETERS.

Config	Value
optimizer	AdamW
base learning rate	0.5e-2 (CIFAR-10), 1e-3 (ImageNet, Tiny)
weight decay	0.05
optimizer momentum	$\beta_1 = 0.9, \beta_2 = 0.999$
layer-wise lr decay	0.65
batch size	128 (CIFAR-10), 256 (Tiny), 1,024 (ImageNet)
learning rate schedule	cosine decay
warmup epochs	10
training epochs	100
augmentation	RandomResizedCrop, RandomHorizontalFlip
augmentation (IN1K)	CutMix, MixUp, Randaugmen
drop path	0.1

TABLE XVII
DIFFERENT LEARNING RATES. FINE-TUNED FOR 50 EPOCHS.

Dataset	Models	LR	Natural	PGD_{10}
CIFAR-10	ViT-T	5.0e-4	76.30	47.60
		1.0e-3	80.69	49.56
		1.0e-2	85.62	48.78
		5.0e-2	85.12	50.30
		1.0e-1	84.51	50.40

D. Comparing MIMIR and MAE Performance with EDM

In Table XVIII, we compare the performance of MIMIR and MAE on CIFAR-10 and Tiny-ImageNet. Both methods pre-train for 800 epochs and fine-tune for 100 epochs. MIMIR consistently outperforms MAE on both natural accuracy and adversarial robustness. These results support our design intuition that adversarial noise builds a more difficult task for Masked Image Modeling, which helps the ViT encoder learn more discriminative features.

In addition, we use the elucidating diffusion model (EDM) data as data augmentation. EDM generative data is usually used to improve the performance of adversarial training [71], [72], [73], [74]. Specifically, we use 5 million generated CIFAR-10 data and 1 million Tiny-ImageNet data provided by [71]. The EDM data is applied to experiments with CIFAR-10 and Tiny-ImageNet but not to ImageNet-1K (EDM data for ImageNet-1K are not provided in [71]).

E. Data Augmentation Evaluation

Figure 8 demonstrates the loss and accuracy while training with different augmentations. "no mix" refers to using only weak augmentation, including RandomResizedCrop and RandomHorizontalFlip. "+mix" refers to using MixUp (0.8) and CutMix (1.0). "+aug" refers to using MixUp (0.8), CutMix (1.0), and Randaugment (rand-m9-mstd0.5-inc1).

TABLE XVIII
NATURAL AND ADVERSARIAL ACCURACY ON CIFAR-10 AND TINY-IMAGENET TEST SET USING MIMIR AND MAE, PRE-TRAINING (800 EPOCHS) AND THEN FINE-TUNING (100 EPOCHS) USING PGD ADVERSARIAL TRAINING. WE USE EDM DATA FROM [71] AS DATA AUGMENTATION.

Dataset	Arch	Pre-train	Natural	PGD_{20}	AA
CIFAR-10	ViT-S	MAE	90.66	59.77	55.48
		MIMIR	**91.94**	**64.04**	**61.06**
	ViT-B	MAE	92.13	63.03	59.44
		MIMIR	**92.42**	**64.88**	**62.03**
Tiny-ImageNet	ViT-S	MAE	62.77	28.23	23.73
		MIMIR	**63.83**	**28.74**	**24.54**
	ViT-B	MAE	65.76	25.25	22.05
		MIMIR	**66.75**	**26.86**	**23.87**

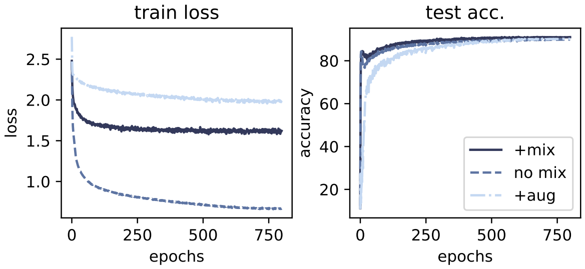

Fig. 8. The training results of using different data augmentations with 800 epochs.

F. Decoder Size

Table XIX provides the performance when pre-training with different decoder sizes. Specifically, the experiments are conducted on different numbers of decoder layers and different hidden sizes. In Table XIX, a deeper decoder may slightly increase the performance, but a larger hidden size may decrease the performance. Additionally, increasing the size of the decoder will also increase the training cost, so we prefer to use a small decoder.

G. Subset of training data

Table XX shows the performance of pre-training on a small subset of training data, which explores whether MIMIR still performs well at a lower cost. MIMIR performs well when it uses only 10% of the training data and can achieve near-full data performance using only 25% of the data.

TABLE XIX
THE PERFORMANCE AND PRE-TRAINING TIME-CONSUMPTION (HOURS) OF VIT-S WITH DIFFERENT DECODER SIZES (DEPTH AND HIDDEN SIZE).

Dataset	Layers	Hidden	Natural	PGD_{20}	Time
CIFAR-10	2	256	86.45	55.12	5.28
	2	512	85.45	51.57	6.76
	4	128	86.52	55.10	5.08
	6	128	87.37	56.93	6.01
Tiny-ImageNet	2	256	64.03	29.13	11.64
	2	512	63.11	28.65	14.02
	4	128	63.44	28.79	11.31
	6	128	64.39	28.61	12.68

TABLE XX
THE PERFORMANCE OF MIMIR PRE-TRAINING ON A SUBSET OF TRAINING DATA.

Dataset	Proportion	Natural	PGD_{20}
CIFAR-10	0.1	83.31	46.49
	0.25	85.75	52.61
	0.5	86.11	53.78
Tiny-ImageNet	0.1	57.92	24.61
	0.25	60.85	26.46
	0.5	62.44	28.13

TABLE XXI
THE PERFORMANCE OF VIT-S WHILE CALCULATING MI FOR MIMIR AT DIFFERENT LAYERS.

Dataset	Layer Index	Natural	PGD_{20}
CIFAR-10	5	86.45	54.51
	7	85.93	53.22
	9	86.39	53.41
	12	86.56	56.76
Tiny-ImageNet	5	63.63	28.67
	7	53.41	24.05
	9	61.79	18.49
	12	63.82	28.74

H. Layer-Wise MI

Table XXI shows the performance of using latent features from different layers to calculate the MI penalty in Equation 8.

We find a phenomenon of MI oscillation, which occurs in layers closer to the inputs. This is because the latent features in those layers do not include the additional normalization layer. In the standard ViT design, an additional normalization layer is included after the final transformer layers to enhance stability and performance. In MIMIR, we use the latent features after the final normalization to calculate the MI penalty, i.e., layer 12 in the case of ViT-S. The MI oscillation also decreases the performance of fine-tuned models, as shown in Table XXI.

I. Efficiency

We provide an analysis of the efficiency of MIMIR. Table XXII provides the total time consumption and memory usage of different adversarial training methods, which are evaluated on four A6000 GPUs. MIMIR is more efficient than 10-step PGD but slightly less efficient than FastAT, with higher robust accuracy than both 10-step PGD and FastAT. Further, we provide the training time of MAE in Table XXII, which shows that the extra training time consumption introduced by the calculation of MI between $x + \delta$ and z is small.

TABLE XXII

THE AVERAGE TIME CONSUMPTION ON 4 GPUS. THE "MEM." REFERS TO GPU MEMORY USAGE. THE TOTAL TIME IS ESTIMATED BASED ON THE TIME CONSUMPTION ON A SINGLE EPOCH. THE TRAINING SCHEDULE FOR PGD$_{10}$ AND FASTAT IS 300 EPOCHS. THE TRAINING SCHEDULE FOR MAE AND MIMIR IS 800 EPOCHS.

			CIFAR-10 [34]		**Tiny-ImageNet** [35]		**ImageNet-1K** [36]	
Architecture	#Parames (M)	Method	time[H]	mem.[GB]	time[H]	mem.[GB]	time[H]	mem.[GB]
ViT-S	21.34	PGD$_{10}$ AT	12.44	2.54×4	25.5	3.99×4	187.64	12.5×4
		FastAT	3.61	2.54×4	5.64	4.03×4	46.29	10.4×4
		MAE	3.58	3.24×4	7.33	3.27×4	59.91	11.1×4
		MIMIR	4.09	3.12×4	8.89	3.18×4	61.22	11.1×4
ViT-B	85.27	PGD$_{10}$ AT	30.1	5.39×4	85.18	8.30×4	451.39	22.1×4
		FastAT	10.23	5.36×4	15.02	8.34×4	113.44	19.8×4
		MAE	11.78	5.95×4	23.67	5.95×4	109.09	17.0×4
		MIMIR	13.11	6.08×4	27.11	6.11×4	113.31	17.0×4
ConViT-S	27.05	PGD$_{10}$ AT	36.88	6.64×4	74.75	12.19×4	552.21	32.5×4
		FastAT	8.88	5.86×4	15.27	10.62×4	119.27	26.4×4
		MAE	7.33	10.6×4	15.0	10.61×4	135.49	27.5×4
		MIMIR	10.0	10.4×4	20.0	10.54×4	135.8	28.3×4

J. Mutual Information and HSIC

MI measures the mutual dependence between two random variables, X and Y. It can be decomposed as:

$$\begin{aligned} I(X,Y) &= H(X) - H(X|Y), \\ &= H(Y) - H(Y|X), \\ &= H(X) + H(Y) - H(X,Y), \end{aligned} \tag{24}$$

where $H(X)$ and $H(Y)$ are the information entropies, $H(X|Y)$ and $H(Y|X)$ are the conditional entropies, and $H(X,Y)$ is the joint entropy of X and Y.

Unfortunately, estimating MI in high-dimensional space is a difficult task since it may involve a precise estimation of the underlying data distribution $P_{(X,Y)}$ or $P_{(X)}$ and $P_{(Y)}$. To address this issue, the deterministic information bottleneck (DIB) [48] uses the recently proposed matrix-based Rényi's α-entropy functional I_α [49], [50], which suggests similar quantities to $I(X,Y)$ in terms of the normalized eigenspectrum of the Hermitian matrix of the projected data in the reproducing kernel Hilbert space (RKHS), but avoids density estimation.

Specifically, given N pairs of samples $(x_i,y_i)_{i=1}^N$ (in our setup, N refers to the mini-batch size), we can obtain two Gram (or kernel) matrices K_x and K_y, for variables X and Y, respectively, with $(K_x)_{i,j}=\kappa_x(x_i,x_j)$, $(K_y)_{i,j}=\kappa_y(y_i,y_j)$, in which κ_x and κ_y are corresponding kernel functions. The information entropy of X can be expressed as:

$$\begin{aligned} H_\alpha(X) &= \frac{1}{1-\alpha}\log_2\left(\text{tr}(\tilde{K_x}^{\alpha})\right) \\ &= \frac{1}{1-\alpha}\log_2\left(\sum_{i=1}^{N}\lambda_i(\tilde{K_x})^{\alpha}\right), \end{aligned} \tag{25}$$

where $\tilde{K}$ is the normalized version of K, i.e., $\tilde{K}=K/\text{tr}(K)$, and $\lambda_i(\tilde{K})$ denotes the i-th eigenvalue of $\tilde{K}$.

Further, the joint entropy for X and Y can be expressed as:

$$H_\alpha(X,Y) = H_\alpha\left(\frac{K_x \circ K_y}{\text{tr}(K_x \circ K_y)}\right), \tag{26}$$

where $K_x \circ K_y$ denotes the Hadamard product between the matrices K_x and K_y.

Given Eqs. (25) and (26), the matrix-based Rényi's α-order mutual information $I_\alpha(X;Y)$ in analogy of Shannon's MI is given by:

$$I_\alpha(X;Y) = H_\alpha(X) + H_\alpha(Y) - H_\alpha(X,Y). \tag{27}$$

Throughout this paper, we use the radial basis function (RBF) kernel $\kappa(x_i,x_j)=\exp(-\frac{\|x_i-x_j\|^2}{2\sigma^2})$ with kernel width σ to obtain the Gram matrices.

The Hilbert–Schmidt Independence Criterion (HSIC) [51] is also a kernel-based dependence measure and is usually used as a surrogate of MI. Formally, the HSIC is defined as the squared norm of the cross-covariance operator $||C_{XY}||^2$:

$$\begin{aligned} &\text{HSIC}_{P_{X,Y}}(X,Y) \\ &= ||C_{XY}||^2 \\ &= \mathbb{E}_{xyx'y'}[\kappa_x(x,x')\kappa_{y'}(y,y')] \\ &+ \mathbb{E}_{xx'}[\kappa_x(x,x')]E_{yy'}[\kappa_y(y,y')] \\ &- 2\mathbb{E}_{xy}[\mathbb{E}_{x'}[\kappa_x(x,x')]\mathbb{E}_{y'}[\kappa_y(y,y')]], \end{aligned} \tag{28}$$

where κ_x and κ_y are kernel functions, $\mathbb{E}$ is the expectation, x' and y' are independent copies of x and y, respectively.

Given N pairs of samples $(x_i,y_i)_{i=1}^N$, the empirical estimator of HSIC is given by:

$$\widehat{\text{HSIC}}_{P_{X,Y}}(X,Y) = \frac{1}{N^2}\text{tr}\left(K_x H H_y H\right), \tag{29}$$

in which $(K_x)_{i,j}=\kappa_x(x_i,x_j)$, $(K_y)_{i,j}=\kappa_y(y_i,y_j)$, and $H=I-\frac{1}{N}\mathbb{1}\mathbb{1}^T$ is the centering matrix.

Artifact Appendix

A. Description & Requirements

MIMIR is a self-supervised pre-training strategy for adversarial robustness of Vision Transformers (ViTs). The training process consists of a pre-training stage and a fine-tuning stage. This artifact supports the experiments and findings presented in the paper by providing the necessary code and trained weights. We also provide scripts for setting up a Python environment and running experiments.

1) How to access: The artifact is available at the permanent repository: https://doi.org/10.5281/zenodo.17807275

2) Hardware dependencies: Training and evaluation require at least one CUDA-enabled GPU, but we strongly recommend using more GPUs. In our case, experiments using small datasets (CIFAR-10, Tiny-ImageNet) are performed on two GPUs (RTX A6000 or RTX A5000). Experiments using the large dataset (ImageNet-1K) are performed on eight GPUs (RTX A6000, RTX A5000, or H100).

3) Software dependencies: The artifact is tested on Ubuntu 22.04.5 LTS with Python 3.10.12 and CUDA 12.7. The training script is implemented with PyTorch 2.1.0. All package dependencies are listed in `requirements.txt`.

4) Benchmarks: We use three commonly used benchmark datasets to evaluate the artifact: CIFAR-10, Tiny-ImageNet, and ImageNet-1K. CIFAR-10 and Tiny-ImageNet are included in the artifact. ImageNet-1k requires accepting the terms of access.[6]

B. Artifact Installation & Configuration

To install necessary dependencies, ensure Python and CUDA are available. Then go to the root path and execute:

```
$ pip install -r requirements.txt
```

C. Experiment Workflow

The artifact requires three primary workflows: (1) Pre-training with MIMIR. (2) Fine-tuning the MIMIR-trained models. (3) Evaluating the fine-tuned models. The `bash` scripts corresponding to each workflow are provided in the artifact.

D. Major Claims

- (C1): MIMIR is effective for ViTs on CIFAR-10. Adversarial training on ViTs is known to be difficult in previous works. This statement is supported by E1, and the results are shown in Table I.
- (C2): MIMIR is effective while scaling up to ImageNet-1K. This statement is supported by E2, and the results are shown in Table III.
- (C3): MIMIR is effective against unforeseen attacks. This statement is supported by E3, and the results are shown in Table V.
- (C4): MIMIR is effective against adaptive attacks. This statement is supported by E4, and the results are shown in Table XI.

There are also other experiments, but they may take more than several days to complete. We include corresponding scripts to execute them in the artifact, but not in this "Major Claims".

E. Evaluation

Overall, the experiments involve four steps for training:

1) Activate the Python environment. By default, we use virtual Python environments, which can be created by the following command lines:

```
$ python3 -m venv your_env
$ source your_env/bin/activate
$ pip install -r requirements.txt
```

2) Pre-training is implemented in `pretrain.py`.
3) Fine-tuning is implemented in `finetune.py`.
4) Evaluation is implemented in `finetune.py`, and can be activated by the flag `--eval`. Evaluation for ImageNet-1K with the 5000 RobustBench testset is implemented in `eval_advmae/sub_imagenet_eval.py`.

- (E1): MIMIR training on CIFAR-10 with ViT-S.

```
$ cd scripts
$ bash train_cifar10.sh
```

- (E2): MIMIR training on ImageNet-1K with ViT-S.

```
$ cd scripts
$ bash train_imagenet.sh
```

- (E3): Evaluation against unforeseen attacks on ImageNet-1K.

```
$ cd scripts
$ bash eval_subimagenet.sh
$ bash eval_imagenet_c.sh
```

- (E4): Evaluation against adaptive attacks.

```
$ cd scripts
$ bash eval_cifar10_adap.sh
$ bash eval_tiny_adap.sh
```

[6] https://image-net.org/download.php