Advancing Deep Metric Learning With Adversarial Robustness

Inderjeet Singh[1] INDERJEET78@NEC.COM
Kazuya Kakizaki KAZUYA1210@NEC.COM
Toshinori Araki TOSHINORI_ARAKI@NEC.COM
NEC Corporation, Kawasaki, Kanagawa, Japan

Editors: Berrin Yanıkoğlu and Wray Buntine

Abstract

Deep Metric Learning (DML) is a prominent subfield of machine learning with extensive practical applications in learning visual similarities. However, DML systems are vulnerable to input distributions during inference that differ from the training data, such as adversarial examples (AXs). In this paper, we introduce MDProp, a framework that enhances the clean data performance and adversarial robustness of DML models by generating novel Multi-Targeted AXs and Unadversarial Examples, in addition to conventional single-targeted AXs, in the feature space. To handle the input distribution shift caused by the generated novel input distributions, MDProp scales the separate batch normalization layer strategy. Our comprehensive experimental analysis demonstrates that MDProp outperforms current state-of-the-art convolutional neural networks by up to 2.95% in terms of R@1 scores for clean data, while simultaneously improving adversarial robustness by up to 2.12 times. Additionally, MDProp achieves state-of-the-art results in data-scarce setting while utilizing only half of the training data. Implementation is available at https://github.com/intherejeet/MDProp.

Keywords: Deep Metric Learning; Image Retrieval; Adversarial Example; Adversarial Robustness.

1. Introduction

Deep Metric Learning (DML) has received significant attention recently due to its prevalent practical applications in establishing similarities between objects by learning distance metrics in the feature space of deep neural networks (DNNs). Despite recent advances in improving DML performance through model architecture, loss function, and data augmentation, little attention has been paid to improving DML performance using adversarial examples (AXs). These carefully crafted instances with small perturbations can reduce the performance and uncover the limited generalization ability of DML models (Su et al. (2019); Madry et al. (2017); Goodfellow et al. (2014); Kurakin et al. (2018b); Sharif et al. (2019); Singh et al. (2022, 2021)).

While some initial studies have focused on using adversarial examples directly as training data to improve the generalization of DNNs for adversarial inputs, this technique, called adversarial training (Goodfellow et al. (2014); Kurakin et al. (2018b); Madry et al. (2017);

[1]Currenty works at Fujitsu Research of Europe Ltd, Slough, UK

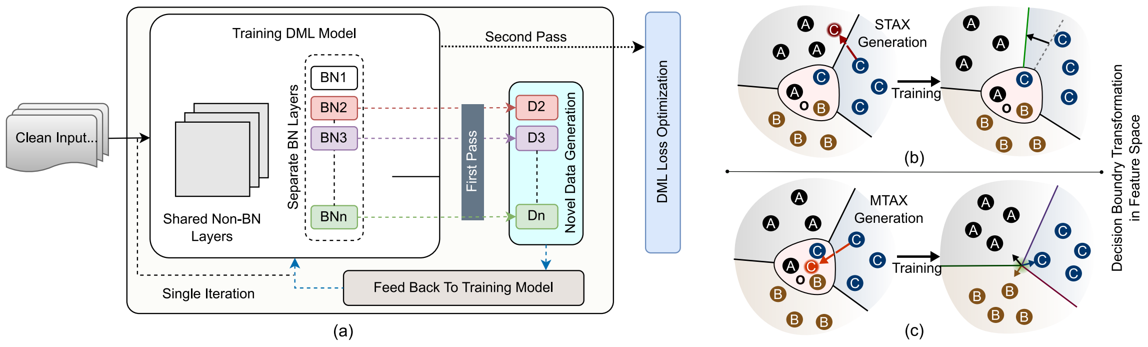

Figure 1: (a) Outline of the **MDProp** framework, which generates novel data distributions during the training of DML models in a disentangled learning framework with multiple BN layers. MDProp uses MTAXs and unadversarial examples in addition to STAXs, enhancing both accuracy and adversarial robustness of trained DML models. (b) Feature space of the DML models trained using only STAXs. STAXs are not designed to target the overlapped regions, leaving the overlap unresolved. (c) MTAXs in MDProp regularize the feature space overlap (region 'O') in DML models, resulting in improved generalization of the trained DML models.

Xie et al. (2019)), resulted in the degradation of clean data performance, which was later demonstrated as an inevitable accuracy-robustness tradeoff (Zhang et al. (2019)). However, Xie et al. (2020) challenged this tradeoff and proposed using separate batch normalization (BN) layers for adversarial data to improve clean data performance. Nonetheless, their method was limited to the *conventional AXs* in *non-DML end-to-end classification DNNs* only, and they did not focus on improving the *adversarial robustness*. We hypothesize that *in DML*, there is a scope for the generation of the novel kinds of adversarial and non-adversarial data in a first of its kind generalized framework to *simultaneously* enhance clean data performance and adversarial robustness.

In this paper, we introduce the ***M**ulti-**D**istribution **Prop**agation (MDProp) framework* (Fig. 1(a)) to enhance the image retrieval performance of DML models on inputs with different adversarial and non-adversarial distributions. MDProp develops novel *Multi-Targeted AXs (MTAXs)* and *Unadversarial Examples*, along with conventional single-targeted AXs (STAXs), in the feature space of a DML model, with each type of data following a distinct distribution that is different from the clean training data. The generated MTAXs simultaneously mimic the deep representations of multiple target classes, implying that they are situated inside multi-class overlap regions in the feature space. The use of MTAXs in training regularizes these overlapped feature space regions, as depicted in (b) and (c) of Fig. 1. The generated unadversarial examples following a different distribution expose the training model to novel intra-class input feature combinations, providing additional training data information, enabling improved generalization. To handle the input distribution shift, MDProp scales the separate BN layer strategy followed by AdvProp (Xie et al. (2020)).

When MDProp employs only conventional feature space STAX during training, it serves as a natural DML extension of the AdvProp.

To the best of our knowledge, this work represents the *first successful attempt in DML to simultaneously enhance model performance on different types of input distributions, by utilizing combinations of STAXs, MTAXs, and unadversarial examples.* We thoroughly evaluated the effectiveness of MDProp on standard DML benchmarks, including CUB200 (Wah et al. (2011)), CARS196 (Krause et al. (2013)), and SOP (Oh Song et al. (2016)), across multiple ResNet architectures (He et al. (2016)), as well as state-of-the-art distillation-based Convolutional Neural Networks (CNNs) from the S2SD method (Roth et al. (2021)), and various DML loss functions. Our results show that MDProp outperforms current state-of-the-art CNN-based methods by up to 2.95% in terms of R@1 scores for clean data, while simultaneously improving adversarial robustness by up to 39.09%, thereby establishing CNNs trained using MDProp as state-of-the-art in DML. Moreover, MDProp achieves state-of-the-art results in data-scarce setting *while utilizing only half of the training data*, owing to its local Lipschitzness enhancing capabilities and robust generalization abilities.

2. Preliminaries

DML. DML aims to find a distance metric $d_\theta : \psi \times \psi \mapsto \mathbb{R}$ on the feature space $\psi \subset \mathbb{R}^D$ of images $\mathcal{X}$ that best satisfy ranking losses (Wang et al. (2019); Wu et al. (2017); Deng et al. (2019); Roth et al. (2020)) defined for class labels $\mathcal{Y}$. An adversary can conveniently create AXs to significantly affect a DML model's job by designing AXs at feature-sapce (Sabour et al. (2015); Rozsa et al. (2017); Zhou et al. (2020); Tolias et al. (2019); Zhou et al. (2021)).

AX Data Augmentation And Adversarial Training. AXs added with adversarial noise δ bring additional features that help training DNN's parameters θ_c learn meaningful data representations (Tsipras et al. (2018)). Adversarial training (Goodfellow et al. (2014); Kurakin et al. (2018b); Madry et al. (2017); Shafahi et al. (2019); Andriushchenko and Flammarion (2020)) is a straightforward strategy that incorporates AXs during training to make DNNs robust against AXs and noisy inputs (Yin et al. (2019); Zhang and Zhu (2019)), solving the following saddle point objective with a loss $\mathcal{L}$:

$$\min_{\theta_c} \mathbb{E}_{(x,y)\sim\mathbb{D}} \left[\max_{\delta \in \mathcal{S}} \mathcal{L}\left(\theta_c, x+\delta, y\right) \right], \tag{1}$$

where $(x, y) \sim \mathbb{D}$ is the clean training data. δ is often crafted using first-order gradient-based methods (Madry et al. (2017); Goodfellow et al. (2014)).

AdvProp. To enhance image recognition accuracy in end-to-end *classification* DL setting, AdvProp (Xie et al. (2020)) proposed the use of AXs with separate BN layers to enable the disentangled learning. It was found that the *input distribution shift* due to AXs was causing a reduced clean data accuracy in adversarial training because BN is based on the single input distribution assumption (Ioffe and Szegedy (2015)). For a DL classifier with parameters θ_c, AdvProp optimized the following objective:

$$\arg\min_{\theta_c} \mathbb{E}_{(x,y)\sim\mathbb{D}} \left[\mathcal{L}\left(\theta_c, x, y\right) + \max_{\delta} \mathcal{L}\left(\theta_c, x+\delta, y\right) \right], \tag{2}$$

where (x, y) is a clean data with distribution $\mathbb{D}$, $\mathcal{L}$ is *classification* loss function, and δ is STAXs' noise.

Multi-targeted AXs. An AX generated to fool a target DML model by simultaneously imitating the deep features of multiple other identities, is called MTAX (Amada et al. (2021)). Let $f : \mathbb{R}^N \longrightarrow \mathbb{R}^D$ be the DML model with parameters θ. For MTAX generation, we craft the adversarial noise δ_f^m as

$$\delta_f^m = \underset{||\delta||_\infty \leq \epsilon}{\arg\min} \left[\frac{1}{T} \cdot \sum_{x^j \in \mathcal{S}_B} ||f(x^l + \delta) - f(x^j)||_p \right], \tag{3}$$

where x^l is the clean sample with identity l, $\mathcal{S}_B$ is the batch sampled from training data $\mathcal{X}_{train}$ and contains images of target identities such that $j \neq l \quad \forall x^j \in \mathcal{S}_B$, T is the number of impersonation targets, ϵ is the l_∞-norm constraint on the size of adversarial noise δ to achieve the imperceptibility objective. Finally, the crafted δ_f^m is added to the clean sample x^l and fools the target model f while holding Eq. 4.

$$d(f(x^l + \delta_f^m), f(x^j)) \leq d(f(x^l), f(x_r^l)) \quad \forall x^j \in \mathcal{S}, \tag{4}$$

where x_r^l is a gallery sample with same identity l as x^l, d is a distance metric, and $\mathcal{S} \subset \mathcal{S}_B$ with $|\mathcal{S}| > 1$.

Unadversarial Examples. Unadversarial examples (UXs), as opposed to AXs, are generated with the aim of *increasing* the model's confidence in the true class of the instance. Salman et al. (2021) first generated UXs for classification models using patches. However, a methodology to generate and leverage UXs to improve DML model training does not exist.

3. Method

We present MDProp, a generalized framework for enhancing the performance of DML models in image retrieval tasks with multiple input distributions. MDProp achieves this by generating MTAXs and unadversarial examples along with STAXs as training data that follow diverse distributions. The framework incorporates the idea of deep disentangled learning (Xie et al. (2020)) through multiple BN layers for each type of generated training data to effectively handle distribution shifts.

Let f_θ be a conventional DML model with trainable parameters $\theta = \{\theta_n, \theta_b\}$, where θ_b denotes the BN parameters and θ_n represents the remaining parameters. MDProp modifies the model architecture by incorporating additional BN layers to generate novel data with different distribution. During the forward training pass, we pass the generated data through the added BN layers while keeping the initial main BN layer reserved for the clean data pass.

Suppose the original training data follows distribution $\mathcal{D}_1$. MDProp generates $(K-1)$ sets of data following distributions $\mathcal{D}_2, \mathcal{D}_3, ...\mathcal{D}_K$, using a model $f_{\theta_n, \theta_b^1, \theta_b^2, ...\theta_b^K}$ with $(K-1)$ additional BN layers. The clean data is processed through f with the parameter set $\{\theta_n, \theta_b^1\}$, while the generated data with $\mathcal{D}_k$ distribution is generated and trained with the parameter

Algorithm 1 Multi-Distribution Propagation Framework for K input distributions

```
Input: Training data $(\mathcal{X}_{train}, \mathcal{Y}_{train})$, $f_{\theta_n, \theta_b^1, \theta_b^2 ... \theta_b^K}$, $\mathcal{B}$, $\phi_n$, $\phi_b$, $\mathcal{L}$,
    $\left\{\gamma^k, generator^k, \Lambda^k, T^k, \mathcal{L}^k\right\}_{k=2}^{K}$;
Initialize parameters of $f$ as $\theta_n, \theta_b^1, \theta_b^2 ... \theta_b^K \leftarrow \phi_n, \phi_b, \phi_b + \gamma^2, ... \phi_b + \gamma^K$;
for each training step do
    Sample batch $(X, Y) \in (\mathcal{X}_{train}, \mathcal{Y}_{train})$ with batch size $\mathcal{B}$;
    for k=2 to K do
        for batch $(X, Y)$ do
            Select target identity set $\mathbb{Y}^k = \{Y_1^{'}, Y_2^{'} ... Y_T^{'}\}$ s.t. $y_j^{'} \neq y_j$ and $(y_j^{'} \in Y_i^{'}, y_j \in Y)$,
              $\forall i \in \{1, 2 ... T\}$, $\forall j \in \{1, 2 ... \mathcal{B}\}$;
            Sample $\mathbb{X}^k = \left\{X_i^{'}\right\}_{i=1}^{T}$ s.t. $\forall\ i \in \{1, 2 ... T\}$, $X_i^{'}$'s has labels $Y_i^{'}$'s;
            Generate $X^k = \{x_i + \delta_i^m\}_{i=1}^{\mathcal{B}} = generator^k(f_{\theta_n, \theta_b^k}, \mathcal{L}^k, X, Y, \mathbb{X}^k, \Lambda^k)$;  ▷ Eq. 3, 4
        end
    end
    Compute $loss = \sum_{\substack{x_i \in X \\ y_i \in Y}} \mathcal{L}(\theta_n, \theta_b^1, x_i, y_i) + \sum_{k=2}^{K} \sum_{\substack{x_i \in X^k \\ y_i \in Y}} \mathcal{L}(\theta_n, \theta_b^k, x_i, y_i)$;
     ▷ While using separate BNs
    Minimize loss by performing back-propagation to $\left\{\theta_n, \theta_b^1, \theta_b^2 ... \theta_b^K\right\}$;
end
Output: Trained Network $f$ with Parameters $\left\{\theta_n, \theta_b^1, \theta_b^2 ... \theta_b^K\right\}$.
```

set $\{\theta_n, \theta_b^k\}$, for all $k \in 2, 3, ..K$. Finally, MDProp optimizes the training objective $\mathcal{Z}_2$ as

$$\mathcal{Z}_2 = \underset{\left\{\theta_n, \theta_b^1 ... \theta_b^K\right\}}{\arg\min}\ \mathbb{E}_{\left\{\substack{(x,y) \sim \mathbb{D}^k \\ \forall k \in \{1, 2 ... K\}}\right\}} \left[\mathcal{L}\left(\left\{\theta_n, \theta_b^k\right\}, x, y \right) \right], \tag{5}$$

where (x, y) is the training data, $\mathcal{L}$ is the DML training loss. MDProp is compatible with all popular DML loss functions. During inference, the auxiliary BN parameters $\{\theta_b^2, \theta_b^3 ... \theta_b^K\}$ are no longer required, and we only use $\theta_{test}^m = \{\theta_n, \theta_b^1\}$ parameters.

3.1. Multi-Distribution Data in MDProp

MDProp employs a distinct data generation process to produce data with varying distributions. Specifically, it generates MTAXs, unadversarial examples, and STAXs using first-order gradient-based optimization methods. MDProp crafts the perturbation δ_f for each clean image x for each type of generated data in the feature space of DML models. The clean images with their respective crafted noise δ_f enable the distribution shift while achieving the generation objective. MDProp is also compatible with conventional data augmentation methods (Cubuk et al. (2018); Lim et al. (2019)).

3.1.1. MTAX Generation

MDProp generates adversarial noise δ_f^m for the MTAX objective specified in Eq. 3 using first-order gradient-based optimization. Furthermore, we use separate BN layer models,

f_{θ_n,θ_b^1} and f_{θ_n,θ_b^2}, to extract the features of x^j and $(x^l + \delta)$, respectively. The different generation objectives of STAXs and MTAXs lead to distinct distributions, thus requiring separate BN layers. In our results section, we will demonstrate that MDProp achieves the best image retrieval performance when we use all three separate BN layers to train with MTAXs, STAXs, and clean data.

3.1.2. Why MTAXs?

DML models may suffer from feature space overlap (region 'O' in Fig. 1(b)) if a proper regularizer is not utilized during training, causing a reduction in test time performance. Feature space of a subset B of classes C is defined as having an *overlap* if a sample x_i^j satisfies:

$$d_\theta\left(f\left(x_i^j\right), f\left(\bar{x}^k\right)\right) \leq \tau \quad \forall k \in B \;\;\&\;\; k \neq j. \tag{6}$$

with $\bar{x}^k$ as the class center for class k and τ as the classification threshold, deep features $f(x_i^j)$ of an unperturbed input x_i^j with identity j result in false ranking predictions from DML models if they lie within the overlapped region, as given by Eq. 6 and Eq. 7.

$$\begin{aligned} d_\theta\left(f\left(x_i^j\right), f\left(\bar{x}^k\right)\right) &\leq d_\theta\left(f\left(x_i^j\right), f\left(x_g^j\right)\right) \\ & k \in B \;\;\&\;\; k \neq j. \end{aligned} \tag{7}$$

Here x_g^j is the *gallery* or *reference* image with identity j. Overlapped regions in the representation space are caused by high similarity between instances of different classes due to limited training data complexity and low discriminative power of the trained DML model.

MDProp leverages MTAXs as additional training data, with successful MTAXs generated during training residing in the overlapped regions. If computational budget restricts the attack generation, the generated MTAXs may be closer to these regions. The effective use of MTAXs in MDProp induces regularization, transforming the deep representation space to reduce or eliminate overlapped regions (Fig. 1), thereby improving generalization in the trained DML model.

3.1.3. Unadversarial Examples

We define an unadversarial example, x_{ux}, for a DML model f (f' for the model different BN layer) at the i^{th} training iteration as $x_{ux} = x + \delta_f^{ux}$. δ_f^{ux} is optimized based on the following objective.

$$\delta_f^{ux} = \underset{||\delta||_\infty \leq \epsilon}{\arg\min} \left[\frac{1}{|\mathcal{S}_U|} \cdot \sum_{x^j \in \mathcal{S}_U} ||f(x^l + \delta) - f^{'}(x_j^l)||_p\right], \tag{8}$$

where $\mathcal{S}_U \leftarrow \{x^l : x^l \in X_{train}, x^l \neq x_j^l, y_{x^l} = y_{x_j^l}\}$, $\mathcal{S}_U \geq 1$, first-order gradient-based optimization is used in our experiments.

The generation of unadversarial examples arises from the *interpolation* performed in the *feature space* due to the crafted noise introduced through the first-order optimization process for the unadversarial objective, at the *input space*. This feature space objective induces an *indirect mixing* of intra-identity features at the input space, which are subsequently utilized in the model's training process through the use of separate batch normalization layers, thereby rendering the transformation *homomorphic* in nature with regards to *intra-identity feature mixing*. As a result, this transformation introduces *novel feature combinations* during training, leading to an enhancement of generalization through the increased input data diversity.

3.1.4. AdvProp Extension

When utilizing a single additional BN layer and STAXs, MDProp represents *AdvProp-D*, the DML extension of the AdvProp method (Xie et al. (2020)). AdvProp-D generates AXs at the embedding space of the training model, where DML metrics are calculated. The training objective is optimized through Eq. 5.

3.2. Working of MDProp

We present MDProp in Algorithm 1 pseudo-code. It requires training data $(\mathcal{X}_{train}, \mathcal{Y}_{train})$, a DML model $f_{\theta_n,\theta_b^1,\theta_b^2...\theta_b^K}$ with default hyperparameters, training batch size $\mathcal{B}$, and pre-trained parameters $\{\phi_n, \phi_b\}$ for transfer learning. Additionally, MDProp uses noise set γ for initializing θ_b^k's, a set T of impersonation targets, a data generation recipe $generate^k$ with hyperparameters Λ, and loss functions $\mathcal{L}^k$ for training and data generation. The hyperparameters Λ determine the adversarial noise strength, the number of gradient update steps, and L_∞ size constraints in the adversarial data generation process.

MDProp starts by initializing the model parameters with ϕ_n, ϕ_b. It optionally uses noise γ while enabling transfer learning for the additional BN layers. A batch of clean data (X, Y) is sampled in each training step, followed by the generation of $K-1$ data batches using separate $generator^k$'s and BN layers. Finally, the loss $\mathcal{L}$ is calculated for all data batches and the process is repeated until a termination condition is met.

4. Experimental Setting

We perform a comprehensive assessment of *MDProp*'s performance using *current* baselines, DML architectures, benchmark datasets, and loss functions. Our datasets are *CUB200* (Wah et al. (2011)), *CARS196* (Krause et al. (2013)), and *Stanford Online Product (SOP)* (Oh Song et al. (2016)). We use *ResNet50* (He et al. (2016)), *ResNet18* (He et al. (2016)), and *ResNet152* (He et al. (2016)) architectures with pre-trained ImageNet (Deng et al. (2009)) parameters, and state-of-the-art S2SD method (Roth et al. (2021)), while maintaining an embedding size of 128. We use the Multisimilarity (Wang et al. (2019)) and ArcFace (Deng et al. (2019)) losses and keep hyperparameters consistent with Roth et al. (2021). For adversarial training, we use PGD (Madry et al. (2017)) and BIM (Kurakin et al. (2018a)) with different numbers of attack targets and set ϵ to 0.01 and 0.1. Our evaluation metrics are Recall@K (Jegou et al. (2010)), NMI (Manning et al. (2010)), and π_{ratio} (Roth et al. (2021)). To evaluate robustness, we generate single-targeted white-box AXs using PGD-20 (Madry et al. (2017)) with $\epsilon = 0.01$. The results for $\epsilon = 0.1$ are in the supplementary material along with more elaborate experimental details. Experiments were conducted on multiple NVIDIA Tesla V100 and Titan V GPUs in distributed training setting.

Met- -hod	T	CUB200 Data								CARS196 Data							
		Multisimilarity Loss				ArcFace Loss				Multisimilarity Loss				ArcFace Loss			
		R@1	R@4	NMI	π_{ratio}	R@1	R@4	NMI	π_{ratio}	R@1	R@4	NMI	π_{ratio}	R@1	R@4	NMI	π_{ratio}
ST	-	62.80 [±0.70]	83.70 [±0.54]	68.55 [±0.38]	1.007	62.22 [±0.01]	83.18 [±0.23]	67.79 [±0.42]	0.726	81.68 [±0.19]	93.47 [±0.27]	69.43 [±0.38]	1.129	79.17 [±0.73]	92.23 [±0.21]	66.99 [±0.04]	**0.661**
AT	1	61.73 [±0.71]	83.20 [±0.07]	68.04 [±0.51]	1.001	60.18 [±0.22]	82.61 [±0.31]	67.75 [±0.05]	0.721	80.02 [±0.42]	92.59 [±0.09]	68.56 [±0.06]	1.082	76.43 [±0.11]	91.14 [±0.06]	67.14 [±0.04]	0.686
AP$'$	1	63.69 [±0.13]	**84.47** [±0.36]	69.15 [±0.27]	0.985	63.23 [±0.09]	84.11 [±0.05]	69.83 [±0.50]	0.723	82.37 [±0.96]	93.54 [±0.51]	70.10 [±1.13]	1.074	79.62 [±0.23]	92.63 [±0.18]	69.31 [±0.59]	**0.681**
MP$'$	3	**64.71** [±0.41]	84.45 [±0.25]	**69.73** [±0.14]	**0.962**	**63.77** [±0.04]	**84.60** [±0.45]	**69.90** [±0.89]	**0.718**	**83.13** [±0.22]	**93.81** [±0.16]	**70.64** [±0.24]	**1.056**	**80.69** [±0.16]	**93.12** [±0.06]	**70.38** [±0.30]	0.689
MP$''$	1,5	**65.75** [±0.28]	**85.23** [±0.21]	**70.43** [±0.04]	**0.974**	**64.07** [±0.11]	**84.78** [±0.15]	**70.32** [±0.06]	**0.703**	**83.81** [±0.50]	**94.31** [±0.26]	**71.59** [±0.56]	**1.056**	**82.02** [±0.36]	**93.65** [±0.30]	**72.43** [±0.18]	0.697
		For Adversarial CUB200 Data								For Adversarial CARS196 Data							
ST	-	32.96 [±0.32]	64.40 [±0.35]	54.38 [±0.45]	1.429	38.45 [±1.19]	67.62 [±1.66]	55.92 [±0.46]	0.761	51.98 [±0.91]	79.99 [±0.80]	54.29 [±0.74]	1.455	34.82 [±1.14]	64.43 [±0.19]	42.85 [±0.20]	0.943
AT	1	38.88 [±0.60]	70.67 [±0.18]	58.33 [±0.21]	1.087	39.44 [±1.80]	70.04 [±0.51]	58.68 [±0.44]	0.743	52.42 [±0.11]	81.13 [±0.15]	56.52 [±0.60]	1.239	36.84 [±0.80]	67.87 [±0.62]	45.99 [±0.56]	0.829
AP$'$	1	**58.80** [±2.15]	**83.38** [±0.08]	62.21 [±0.29]	0.921	**51.24** [±0.96]	77.81 [±0.09]	**63.33** [±1.04]	0.712	**79.11** [±1.35]	**93.05** [±0.43]	70.87 [±0.99]	0.978	**65.67** [±1.24]	**87.11** [±0.83]	62.01 [±0.57]	**0.723**
MP$'$	3	57.06 [±1.15]	81.42 [±0.80]	**67.03** [±0.68]	**0.838**	50.96 [±0.28]	**77.84** [±0.42]	62.73 [±0.36]	**0.705**	78.18 [±0.52]	92.55 [±0.38]	**71.21** [±0.52]	**0.896**	64.76 [±0.71]	86.88 [±0.31]	**62.38** [±0.25]	0.726
MP$''$	1,5	**57.27** [±1.90]	**82.21** [±0.84]	**68.06** [±0.61]	**0.836**	**51.93** [±0.32]	**80.15** [±0.27]	**66.28** [±0.02]	**0.645**	**80.25** [±0.19]	**93.50** [±0.01]	**72.37** [±0.06]	**0.899**	**73.91** [±0.07]	**91.28** [±0.08]	**69.01** [±1.13]	**0.688**

Table 1: Image retrieval performance for *Standard Training* (ST), *Adversarial Training* (AT), *AdvProp-D* (AP$'$), and *MDProp* (MP) with one (MP$'$) and two (MP$''$) additional BN layers, on clean and STAX inputs from CUB200 (Wah et al. (2011)) and CARS196 (Krause et al. (2013)) datasets. ***Embedding dimensions were kept 128***, and T denotes the number of MTAX targets. Adversarial datasets were generated using *single-targeted white-box PGD-20* attacks with $\epsilon = 0.01$ on the test sets. Best and second best results per setup are denoted in **bluebold** and **bold**, respectively.

To assess improved generalization, we combined π_{ratio} with performance against MTAXs. In the *data-scarce* evaluation setting, we generated three training subsets per dataset by uniform class-wise sampling, creating subsets with fractions of 0.25, 0.50, and 0.75 of the original training data. We considered 2, 3, and 4 separate BN layers in MDProp, and used only the main BN layer during inference.

5. Results and Discussions

Clean Data Performance Gains. Tab. 1 presents results for the *CUB200* (Wah et al. (2011)) and *CARS196* (Krause et al. (2013)) datasets using Multisimilarity (Wang et al. (2019)) and ArcFace (Deng et al. (2019)) loss functions. Our proposed method, MDProp, *outperforms* standard and adversarial training baselines as well as AdvProp-D, with the highest image retrieval performance achieved using clean, STAX, and MTAX inputs and three BN layers. MDProp also outperforms AdvProp-D when trained using only MTAXs and clean data. Similar performance gains are observed with four separate BN layers (Sec. 4.4 in supplemental). The *lower* π_{ratio} *scores* for MDProp *validate* our hypothesis of regularization in DML models due to MTAXs. MDProp also shows better performance against MTAX inputs during *inference*.

For larger SOP data (Oh Song et al. (2016)), MDProp trained with a mix of STAXs, MTAXs, and clean data achieves the best performance for clean inputs (Tab. 3). The relatively lower gains on SOP data may be due to the presence of a large number of classes with already low π_{ratio} scores for vanilla training baselines, leading to ineffective adversarial target selection during the MTAX generation process in MDProp.

Adversarial Robustness Gains. MDProp not only boosts image retrieval performance on unperturbed inputs but also significantly strengthens the robustness of DML models against white-box adversarial attacks. Tab. 1, 2, and 3 reveal up to 86% higher recall scores, 47% higher NMI scores, and a 41% decrease in π_{ratio} scores compared to baselines for the adversarial inputs. Using both STAXs and MTAXs with two extra BN layers during training yields the most robust models, surpassing even the AdvProp-D case. The robustness improvement remains consistent even for the four separate BNs, highlighting the generalization benefit from incorporating MTAXs with additional BN layers.

Performance With Unadversarial Examples. The combination of unadversarial examples and clean data in MDProp with two separate BN layers improved the clean data R@1 score to 63.56% (Tab. 8 in supplemental) for the Multisimilarity loss setting used for Tab. 1, with modest improvement in π_{ratio} scores compared to the MTAX case. No further performance gains were observed with the combination of unadversarial examples and adversarial data in the MDProp setting, possibly due to incompatibility with adversarial data or overlap with the combination of clean and adversarial data during training. This issue remains an open research question.

No improvement was achieved using clean data augmentation techniques such as standard crop and affine transformations when separate BN layers were used. In some cases, separate BN layers even resulted in performance reduction on clean data.

Data-Scarce Setting Performance Gains. To investigate the superior generalization capability of MDProp, we compared it to the baseline Standard Training in the data-scarce setting on the CUB200 (Wah et al. (2011)) and CARS196 (Krause et al. (2013)) datasets. Fig. 2 presents a plot of the performance of MDProp and the baseline approach, with horizontal lines indicating MDProp's performance at 50% data usage. Our results demonstrate that *MDProp matches conventional DML training performance while using only half of the training data*, suggesting the usefulness of MDProp in practical scenarios where data availability is limited.

Performance Across Architectures. MDProp demonstrates exceptional clean data performance and adversarial robustness, as demonstrated in Tab. 2. This superiority is consistent across DL architectures of varying depth and even when combined with the S2SD method (Roth et al. (2021)). Notably, MDProp in the S2SD setting for the CUB200

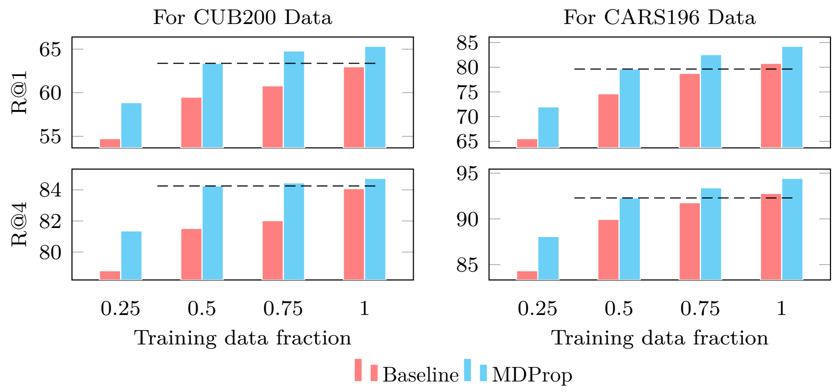

Figure 2: Performance comparison of MDProp and Standard Training in ***data-scarce*** setting evaluation on CUB200 (Wah et al. (2011)) and CARS196 (Krause et al. (2013)) datasets. The plot shows that MDProp achieves similar performance to conventional DML training while using only 50% of the training data, as indicated by the horizontal lines. This suggests that MDProp has the potential to improve the generalization abilities of DML models.

dataset outperforms the state-of-the-art distillation-based CNN training method (Roth et al. (2021)) by a significant margin of 1.39% on R@1 score, and also in NMI and π_{ratio} scores.

Evaluating Feature Space Overlap. Our results, summarized in Tab. 1, 2, and 3, demonstrate decreased π_{ratio} scores for MDProp in the majority of cases (also shown in the t-SNE plot of Fig. 2 in supplemental). The decreased π_{ratio} scores mean increased feature space sparseness contributing to improved performance that can be attributed to the use of MTAXs, as evidenced by our evaluation of the trained model's performance on MTAX inputs. We find that the optimal performance is achieved when MDProp is trained using a combination of both STAXs and MTAXs (see Section 4.1 in Supplemental), followed by MDProp trained using MTAXs and STAXs, respectively.

Scalability For Different Attack Target Counts. Our experiments with T values of 1, 2, 3, 5, and 10 reveal that increasing T improves performance on clean data up to a point where the hyperparameters of the predefined generation recipe provide *enough* semantic capability to the attack generation process. This results in the embedding space positions of the generated MTAXs shifting to the overlapping regions of the DML model during training (Sec. 4 in supplemental). MDProp using clean data and MTAXs achieved the best performance for $T = 3$, and MDProp using clean data, STAXs, and MTAXs performed best for $T = 5$. Smaller values $T = 2$ results in lesser performance improvements due to the reduced likelihood of finding highly overlapping embedding space regions.

BN Layer Separation Impact. In Tab. 1 and 2, we demonstrate that adversarial training, which does not utilize separate BN layers, increases adversarial robustness compared to standard training. However, this approach sacrifices performance on clean data. Experimenting with a DML model trained on a mix of clean and MTAX data, without separate

Met- -hod	T	ResNet50+S2SD Method								ResNet18				ResNet152			
		Clean CUB200 Data				*Adversarial* CUB200 Data				Clean CUB200 Data				Clean CUB200 Data			
		R@1	R@4	NMI	π_{ratio}	R@1	R@4	NMI	π_{ratio}	R@1	R@4	NMI	π_{ratio}	R@1	R@4	NMI	π_{ratio}
ST	-	67.69 [±0.13]	86.32 [±0.08]	71.46 [±0.13]	1.123	47.35 [±1.24]	76.08 [±0.64]	60.26 [±0.40]	1.393	58.81 [±0.52]	81.34 [±0.33]	66.12 [±0.45]	1.131	65.11 [±0.28]	84.64 [±0.10]	69.70 [±0.02]	0.967
AT	1	66.46 [±0.59]	85.63 [±0.12]	70.78 [±0.40]	1.092	45.13 [±1.09]	75.40 [±0.40]	60.89 [±0.25]	1.416	58.33 [±0.13]	81.15 [±0.11]	65.54 [±0.31]	1.093	64.98 [±0.47]	84.83 [±0.46]	70.56 [±0.14]	0.896
AP$'$	1	68.14 [±0.16]	86.45 [±0.05]	71.18 [±0.10]	**1.091**	**62.47** [±1.37]	84.18 [±0.59]	**69.64** [±0.11]	**1.102**	60.91 [±0.47]	82.52 [±0.44]	66.52 [±0.57]	**1.028**	66.95 [±0.04]	85.88 [±0.23]	71.72 [±0.21]	0.916
MP$'$	3	**68.76** [±0.24]	**86.47** [±0.27]	**71.78** [±0.29]	**1.106**	**62.47** [±0.13]	**84.66** [±0.84]	69.62 [±0.86]	1.109	**60.92** [±0.18]	**82.82** [±0.11]	**66.56** [±0.30]	**1.024**	**66.66** [±0.24]	**85.77** [±0.03]	**71.73** [±0.35]	**0.910**
MP$''$	1,5	**69.08** [±0.23]	**87.19** [±0.19]	**71.98** [±0.17]	1.252	**65.01** [±0.02]	**86.60** [±0.08]	**71.13** [±0.21]	**1.034**	**61.67** [±0.47]	**82.75** [±0.17]	**67.38** [±0.47]	1.091	**67.63** [±0.16]	**86.20** [±0.06]	**72.61** [±0.01]	**0.902**

Table 2: Comparison of AdvProp-D (AP$'$) and MDProp (MP) methods with Standard Training (ST) and Adversarial Training (AT) on *ResNet18*, *ResNet50* with *S2SD* (Roth et al. (2021)), and *ResNet152* architectures for image retrieval performance on *CUB200* (Wah et al. (2011)) dataset. The acronyms and other settings were identical to those in Tab. 1.

BNs, shows that this approach does not improve clean data performance and maintains similar robustness as adversarial training. This *supports* the hypothesis from AdvProp (Xie et al. (2020)) that separate BN layers are crucial in handling input distribution shifts.

Input Distribution Shifts By MTAXs. Gradient descent during MTAX generation is restricted to the overlapping embedding spaces of the model, while STAX generation benefits from a larger feasible solution space, leading to lower optimization complexity. Our hypothesis is that the unique generation process of MTAXs results in a distinct distribution compared to STAXs and clean data.

Our hypothesis was verified by reproducing the methodology of Xie et al. (2020). Specifically, we trained a ResNet50 model with Multisimilarity loss in the MDProp framework, using two additional BN layers for STAXs and MTAXs. Upon evaluation, we observed a *decrease* of 0.7% and 1.8% in R@1 scores for clean data when BN layers utilized for STAXs and MTAXs during training were used for inference in the model, respectively.

Additionally, Fig. 3 illustrates a clear and *significant pairwise variation* in the learned β and γ parameters of the BN layers used for different types of training data in MDProp. This difference in learned parameters further confirms that MTAXs follow a distinct distribution, requiring additional BN layers during training to mitigate input distribution shifts.

Training and Inference Complexities. The computational complexity of MDProp is twofold. During training, the complexity is $O(\text{Attack Iterations} \times (K-1) \times N) + O(K \times N)$, accounting for $K-1$ additional data sets generated via PGD attacks and K batch normalization layers. During inference, since MDProp uses BN layer used for the original training data only, the complexity remains $O(N)$, analogous to standard models, meaning no additional computational costs.

Method	T	Clean Data				Adversarial Data			
		R@1	R@4	NMI	π_{ratio}	R@1	R@4	NMI	π_{ratio}
ST	-	**78.09**	86.55	89.98	0.469	54.60	66.09	84.96	0.636
AP'	1	77.36	86.17	89.98	**0.410**	**71.96**	82.86	88.57	**0.422**
MP'	3	77.73	**86.98**	**89.99**	0.475	**72.95**	**83.90**	**88.82**	0.456
MP''	1,5	**78.70**	**87.19**	**90.27**	**0.452**	71.89	**83.00**	**88.62**	**0.438**

Table 3: Results for the SOP (Oh Song et al. (2016)) dataset while using *ResNet50* with *Multisimilarity* (Wang et al. (2019)) loss. Other acronyms and settings were the same as in Tab. 1.

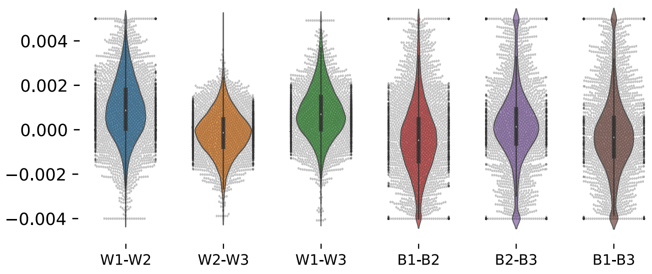

Figure 3: Visualization of layer-wise variations in the learned parameters of BN layers for various inputs in MDProp. The x-axis labels denote the weight (W) and bias (B) parameters, followed with numbers 1, 2, and 3 indicating the respective BN layers used for clean, STAX, and MTAX inputs. Differences are depicted at equal relative levels. The plot shows that every pair of BN layers exhibit significant differences among significant number of their learned parameters, indicating their adaptation to diverse input distributions.

6. Related Work

DML. Our work pushes the boundaries of standard DML techniques by leveraging adversarial data in a disentangled learning environment to achieve improved performance across multiple input distributions. The conventional DML approach focuses on ranking losses, data sampling, and data augmentation.

MDProp Advancement. Adversarial training has long been known to compromise clean data performance in DL models (Kurakin et al. (2018b); Madry et al. (2017); Xie et al. (2019); Shafahi et al. (2019); Andriushchenko and Flammarion (2020)). AdvProp (Xie et al. (2020)) and FastAdvProp (Mei et al. (2022)) have made strides in using separate BN layers for clean and adversarial data to improve clean data accuracy for end-to-end classification models. Raghunathan et al. (2020) proposed RST for semi-supervised learning. However, no prior work (Jiang et al. (2020); Ho and Nvasconcelos (2020); Chen et al. (2021); Mei et al.

(2022); Raghunathan et al. (2020)) has addressed the challenge of *simultaneously* improving clean and adversarial data performance for DML models in the image retrieval task.

This paper presents a unique approach to improve the image retrieval performance of DML models on both clean and adversarial data. Our MDProp framework generates and leverages novel adversarial data and unadversarial examples in the DML domain, leading to improved performance. By scaling the number of separate BN layers, MDProp outperforms state-of-the-art CNNs while using only half the training data in the data-scarce setting, demonstrating superior generalization. Furthermore, we present an efficient transfer learning approach for the auxiliary BN layers that effectively utilizes the pre-trained parameters of conventional architectures, reducing computational complexity. Our contributions represent a significant advancement in the field of DML.

7. Conclusion

In this paper, we introduced the *MDProp* framework to enhance the generalization of DML models in image retrieval tasks by generating and utilizing *MTAXs*, *Unadversarial Examples*, and conventional STAXs. Our evaluations show that MDProp outperforms state-of-the-art CNN-based methods by up to 2.95% in terms of R@1 scores for *clean data* and improves *adversarial robustness* by up to 2.12 times. Additionally, we demonstrate that MDProp achieves state-of-the-art results in *data-scarce setting*, making it a promising approach for scenarios where labeled data is limited. MDProp is a general framework that can be integrated into different DML architectures, loss functions, and distillation-based methods. Our work opens up opportunities for further research to enhance the performance and robustness of DML models.

References

Takuma Amada, Seng Pei Liew, Kazuya Kakizaki, and Toshinori Araki. Universal adversarial spoofing attacks against face recognition. In *2021 IEEE International Joint Conference on Biometrics (IJCB)*, pages 1–7. IEEE, 2021.

Maksym Andriushchenko and Nicolas Flammarion. Understanding and improving fast adversarial training. *Advances in Neural Information Processing Systems*, 33:16048–16059, 2020.

Xiangning Chen, Cihang Xie, Mingxing Tan, Li Zhang, Cho-Jui Hsieh, and Boqing Gong. Robust and accurate object detection via adversarial learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16622–16631, 2021.

Ekin D Cubuk, Barret Zoph, Dandelion Mane, Vijay Vasudevan, and Quoc V Le. Autoaugment: Learning augmentation policies from data. *arXiv preprint arXiv:1805.09501*, 2018.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

Jiankang Deng, Jia Guo, Niannan Xue, and Stefanos Zafeiriou. Arcface: Additive angular margin loss for deep face recognition. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4690–4699, 2019.

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

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

Chih-Hui Ho and Nuno Nvasconcelos. Contrastive learning with adversarial examples. *Advances in Neural Information Processing Systems*, 33:17081–17093, 2020.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pages 448–456. PMLR, 2015.

Herve Jegou, Matthijs Douze, and Cordelia Schmid. Product quantization for nearest neighbor search. *IEEE transactions on pattern analysis and machine intelligence*, 33(1): 117–128, 2010.

Ziyu Jiang, Tianlong Chen, Ting Chen, and Zhangyang Wang. Robust pre-training by adversarial contrastive learning. *Advances in Neural Information Processing Systems*, 33: 16199–16210, 2020.

Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3d object representations for fine-grained categorization. In *Proceedings of the IEEE international conference on computer vision workshops*, pages 554–561, 2013.

Alexey Kurakin, Ian J Goodfellow, and Samy Bengio. Adversarial examples in the physical world. In *Artificial intelligence safety and security*, pages 99–112. Chapman and Hall/CRC, 2018a.

Alexey Kurakin, Ian J Goodfellow, and Samy Bengio. Adversarial examples in the physical world. In *Artificial intelligence safety and security*, pages 99–112. Chapman and Hall/CRC, 2018b.

Sungbin Lim, Ildoo Kim, Taesup Kim, Chiheon Kim, and Sungwoong Kim. Fast autoaugment. *Advances in Neural Information Processing Systems*, 32, 2019.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017.

Christopher Manning, Prabhakar Raghavan, and Hinrich Schütze. Introduction to information retrieval. *Natural Language Engineering*, 16(1):100–103, 2010.

Jieru Mei, Yucheng Han, Yutong Bai, Yixiao Zhang, Yingwei Li, Xianhang Li, Alan Yuille, and Cihang Xie. Fast advprop. *arXiv preprint arXiv:2204.09838*, 2022.

Hyun Oh Song, Yu Xiang, Stefanie Jegelka, and Silvio Savarese. Deep metric learning via lifted structured feature embedding. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4004–4012, 2016.

Aditi Raghunathan, Sang Michael Xie, Fanny Yang, John Duchi, and Percy Liang. Understanding and mitigating the tradeoff between robustness and accuracy. *arXiv preprint arXiv:2002.10716*, 2020.

Karsten Roth, Timo Milbich, Samarth Sinha, Prateek Gupta, Bjorn Ommer, and Joseph Paul Cohen. Revisiting training strategies and generalization performance in deep metric learning. In *International Conference on Machine Learning*, pages 8242–8252. PMLR, 2020.

Karsten Roth, Timo Milbich, Bjorn Ommer, Joseph Paul Cohen, and Marzyeh Ghassemi. Simultaneous similarity-based self-distillation for deep metric learning. In *International Conference on Machine Learning*, pages 9095–9106. PMLR, 2021.

Andras Rozsa, Manuel Günther, and Terranee E Boult. Lots about attacking deep features. In *2017 IEEE International Joint Conference on Biometrics (IJCB)*, pages 168–176. IEEE, 2017.

Sara Sabour, Yanshuai Cao, Fartash Faghri, and David J Fleet. Adversarial manipulation of deep representations. *arXiv preprint arXiv:1511.05122*, 2015.

Hadi Salman, Andrew Ilyas, Logan Engstrom, Sai Vemprala, Aleksander Madry, and Ashish Kapoor. Unadversarial examples: Designing objects for robust vision. *Advances in Neural Information Processing Systems*, 34:15270–15284, 2021.

Ali Shafahi, Mahyar Najibi, Mohammad Amin Ghiasi, Zheng Xu, John Dickerson, Christoph Studer, Larry S Davis, Gavin Taylor, and Tom Goldstein. Adversarial training for free! *Advances in Neural Information Processing Systems*, 32, 2019.

Mahmood Sharif, Sruti Bhagavatula, Lujo Bauer, and Michael K Reiter. A general framework for adversarial examples with objectives. *ACM Transactions on Privacy and Security (TOPS)*, 22(3):1–30, 2019.

Inderjeet Singh, Satoru Momiyama, Kazuya Kakizaki, and Toshinori Araki. On brightness agnostic adversarial examples against face recognition systems. In *2021 International Conference of the Biometrics Special Interest Group (BIOSIG)*, pages 1–5. IEEE, 2021.

Inderjeet Singh, Toshinori Araki, and Kazuya Kakizaki. Powerful physical adversarial examples against practical face recognition systems. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 301–310, 2022.

Jiawei Su, Danilo Vasconcellos Vargas, and Kouichi Sakurai. One pixel attack for fooling deep neural networks. *IEEE Transactions on Evolutionary Computation*, 23(5):828–841, 2019.

Giorgos Tolias, Filip Radenovic, and Ondrej Chum. Targeted mismatch adversarial attack: Query with a flower to retrieve the tower. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5037–5046, 2019.

Dimitris Tsipras, Shibani Santurkar, Logan Engstrom, Alexander Turner, and Aleksander Madry. There is no free lunch in adversarial robustness (but there are unexpected benefits). *arXiv preprint arXiv:1805.12152*, 2(3), 2018.

Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. The caltech-ucsd birds-200-2011 dataset. 2011.

Xun Wang, Xintong Han, Weilin Huang, Dengke Dong, and Matthew R Scott. Multi-similarity loss with general pair weighting for deep metric learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5022–5030, 2019.

Chao-Yuan Wu, R Manmatha, Alexander J Smola, and Philipp Krahenbuhl. Sampling matters in deep embedding learning. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2840–2848, 2017.

Cihang Xie, Yuxin Wu, Laurens van der Maaten, Alan L Yuille, and Kaiming He. Feature denoising for improving adversarial robustness. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 501–509, 2019.

Cihang Xie, Mingxing Tan, Boqing Gong, Jiang Wang, Alan L Yuille, and Quoc V Le. Adversarial examples improve image recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 819–828, 2020.

Dong Yin, Raphael Gontijo Lopes, Jon Shlens, Ekin Dogus Cubuk, and Justin Gilmer. A fourier perspective on model robustness in computer vision. *Advances in Neural Information Processing Systems*, 32, 2019.

Hongyang Zhang, Yaodong Yu, Jiantao Jiao, Eric Xing, Laurent El Ghaoui, and Michael Jordan. Theoretically principled trade-off between robustness and accuracy. In *International conference on machine learning*, pages 7472–7482. PMLR, 2019.

Tianyuan Zhang and Zhanxing Zhu. Interpreting adversarially trained convolutional neural networks. In *International Conference on Machine Learning*, pages 7502–7511. PMLR, 2019.

Mo Zhou, Zhenxing Niu, Le Wang, Qilin Zhang, and Gang Hua. Adversarial ranking attack and defense. In *ECCV*, pages 781–799, 2020. ISBN 978-3-030-58568-6.

Mo Zhou, Le Wang, Zhenxing Niu, Qilin Zhang, Yinghui Xu, Nanning Zheng, and Gang Hua. Practical relative order attack in deep ranking. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 16413–16422, 2021.