ADVERSARIAL ROBUSTNESS FOR DEEP METRIC LEARNING

by

Ezgi Paket

B.S., Industrial Engineering, Istanbul Technical University, 2019

B.S., Computer Engineering, Istanbul Technical University, 2020

Submitted to the Institute for Graduate Studies in
Science and Engineering in partial fulfillment of
the requirements for the degree of
Master of Science

Graduate Program in Computer Engineering

Boğaziçi University

2024

ACKNOWLEDGEMENTS

I would first like to thank my thesis advisor, Assist. Prof. İnci Meliha Baytaş, for her guidance and endless support. Whenever I ran into a trouble spot, she steered me in the right direction with great motivation and belief. Her encouragement and advice carried me through all of the stages of this thesis.

I am profoundly grateful to my master thesis defense committee members, Prof. Dr. Bilge Günsel and Assoc. Prof. Emre Uğur, for their valuable suggestions.

I would also like to give special thanks to my family, friends, and DnA team at Eczacıbaşı Bilişim for their continuous love and encouragement during this journey. Their unwavering support kept me going through the most challenging days of my thesis writing process.

This study is funded by the Scientific and Technological Research Council of Turkey ARDEB 3501 under grant number 121E594. The experiments within this thesis are performed at the barbun-cuda and akya-cuda computing clusters of TUBITAK ULAKBIM High Performance and Grid Computing Center (TRUBA).

ABSTRACT

ADVERSARIAL ROBUSTNESS FOR DEEP METRIC LEARNING

Deep metric learning has been virtually used in various clustering, recognition, and retrieval frameworks. The goal of deep metric learning is to learn a non-linear projection that maps input samples into a latent space where the distance between similar and dissimilar samples is minimized and maximized, respectively. Convolutional neural networks used in deep metric learning for images are known to be vulnerable to adversarial attacks that alter the model's behavior by adding imperceptible perturbation to input images. Adversarial training, where adversarial samples are generated at each iteration, is one of the prominent techniques to defend deep metric learning models against adversarial attacks. Despite its popularity, adversarial training poses several drawbacks, such as increasing computational complexity and trade-off between robustness and generalization. Therefore, the quest for adversarial defense techniques that apply to various architectures without burdening the model training continues. This thesis proposes an adversarial defense framework for deep metric learning without adversarial sample generation during training. The proposed framework generates adversarial samples using a pre-trained network once and employs them to learn a more robust mapping. Thus, the training time of the deep metric learning model is not increased with adversarial attack generation. In addition, the proposed robust training is done with a lightweight architecture in the latent space to prevent overfitting to specific adversarial perturbations that deepen the gap between the metric learning performance with natural and adversarial samples. The proposed approach is evaluated with the most prominent benchmark datasets in metric learning, and the results indicated that the proposed robust metric learning framework enhances the adversarial retrieval and clustering without causing a significant decline in natural performances.

ÖZET

DERİN METRİK ÖĞRENMESİ İÇİN HASMANE DAYANIKLILIK

Derin metrik öğrenme, çeşitli kümeleme, tanıma ve erişim görevlerinde yaygın olarak kullanılmaktadır. Derin metrik öğrenmenin amacı, benzer örnekler arasındaki mesafeyi en aza indirgerken farklı örnekler arasındaki mesafeyi en üst düzeye çıkarabilmek için girdileri gizli bir uzaya eşleyen doğrusal olmayan bir projeksiyon öğrenmektir. Görüntüler için derin metrik öğrenmede kullanılan evrişimli sinir ağlarının, girdi görüntülerine algılanamayan bozulmalar ekleyerek modelin davranışını değiştiren hasmane saldırılara karşı savunmasız olduğu bilinmektedir. Her iterasyonda hasmane örneklerin üretildiği hasmane eğitim, derin metrik öğrenme modellerini hasmane saldırılara karşı savunmada öne çıkan tekniklerdendir. Hasmane eğitim popülaritesine rağmen, hesaplama karmaşıklığını artırma ve dayanıklılık ve genelleştirme arasında denge kurma gerekliliği gibi birkaç dezavantajı barındırmaktadır. Bu nedenle, model eğitimini zorlaştırmayan çeşitli mimarilere uygulanabilen hasmane savunma teknikleri arayışı devam etmektedir. Bu tez, eğitim sırasında hasmane örnek üretimi olmadan derin metrik öğrenme için bir hasmane dayanıklılık methodu önermektedir. Önerilen yaklaşım ile hasmane örnekler bir kez önceden eğitilmiş bir ağ kullanarak üretilir ve daha dayanıklı bir öğrenme için bu örnekler kullanılır. Böylece, derin metrik öğrenme modelinin eğitim süresi, hasmane saldırı üretimiyle artmaz. Ek olarak, önerilen dayanıklı eğitim, metrik öğrenme performansı ile doğal ve hasmane örnekler arasındaki farkı derinleştiren belirli hasmane bozulmalara aşırı uyumu önlemek için gizli uzayda küçük bir mimariyle yapılır. Önerilen yaklaşım, metrik öğrenmede en önemli referans veri kümeleriyle değerlendirilmiştir. Elde edilen sonuçlar, önerilen dayanıklı metrik öğrenme methodunun hasmane erişim ve kümelemeyi doğal performanslarda önemli bir düşüşe neden olmadan geliştirdiğini göstermiştir.

TABLE OF CONTENTS

ACKNOWLEDGEMENTS iii
ABSTRACT iv
ÖZET v
LIST OF FIGURES viii
LIST OF TABLES x
LIST OF SYMBOLS xii
LIST OF ACRONYMS/ABBREVIATIONS xiv
1. INTRODUCTION 1
1.1. Contributions of the Thesis 4
2. RELATED WORK 6
2.1. Adversarial Attacks 6
2.1.1. Fast Gradient Sign Method (FGSM) 6
2.1.2. Projected Gradient Descent (PGD) 7
2.2. Adversarial Robustness 8
2.3. Deep Metric Learning 10
2.4. Adversarial Defenses in Deep Metric Learning 14
3. METHOD 17
3.1. Embedding Module 18
3.2. Adversarial Sample Generation 19
3.3. Robust Metric Learning Module 20
4. EXPERIMENTS 24
4.1. Datasets 24
4.1.1. CUB200-2011 24
4.1.2. CARS196 24
4.1.3. Stanford Online Products (SOP) 25
4.2. Evaluation Metrics 25
4.2.1. Normalized Mutual Information 25
4.2.2. Recall@K 26

4.3. Implementation Details 27
4.3.1. Embedding Module and PGD Attack Generation 27
4.3.2. Robust Metric Learning Module 28
4.4. Baseline Studies 32
4.5. Quantitative Results 33
4.6. Qualitative Results 45
5. CONCLUSION 50
REFERENCES 52

LIST OF FIGURES

Figure 1.1. Top 5 nearest neighbors comparisons of ResNet-50 [12] embeddings in natural and adversarial settings. Each color represents the different class. 2

Figure 3.1. Proposed robust deep metric learning model. Embeddings of natural, $f(\mathbf{x};\theta) \in \mathbb{R}^d$, and adversarial images, $f\left(\mathbf{x}^{'};\theta\right) \in \mathbb{R}^d$, are extracted using embedding module. The embedding module is frozen, while the metric learning module is training. The outputs of the metric learning module, $(g(f(\mathbf{x});\phi) \in \mathbb{R}^{d^{'}}$ and $\left(g(f\left(\mathbf{x}^{'}\right);\phi\right) \in \mathbb{R}^{d^{'}}$, are provided to related loss function. 17

Figure 3.2. Adversarial metric learning framework. Training for T epochs, θ embedding module parameters, ϕ metric learning module parameters, τ learning rate, D dataset, n mini-batch size. 18

Figure 3.3. PGD attack generation [17] framework. D dataset, n mini-batch size, θ is the parameters of the embedding network, $\nabla_{x_{adv}}$ represents the gradient of the triplet loss, $\mathcal{L}_{\text{Triplet}}$, with respect to an anchor image $\mathbf{x}_{\text{adv}}$, ϵ controls amount of perturbation, the number of PGD steps is denoted by m, γ is the attack step size. 21

Figure 3.4. T-sne of natural and adversarial embeddings for a sample from CUB200-2011. Adversarial samples move away from their natural counterparts, while reducing the distance between the adversarial and natural samples from different categories. 22

Figure 4.1. Top 5 nearest neighbors comparisons of original ResNet-50 and RML embeddings in natural and adversarial settings for CARS196 dataset. 46

Figure 4.2. Top 5 nearest neighbors comparisons of original ResNet-50 and RML embeddings in natural and adversarial settings for CUB200-2011 dataset. 47

Figure 4.3. Top 5 nearest neighbors comparisons of ResNet-50 and RML embeddings in natural and adversarial settings for SOP dataset. . . . 48

LIST OF TABLES

Table 4.1. Performance comparison of different sampling strategies for CUB200-2011 dataset. 29

Table 4.2. Performance comparison of different sampling strategies for CARS196 dataset. 30

Table 4.3. Performance comparison of different sampling strategies for SOP dataset. 31

Table 4.4. Natural and adversarial performances of robust metric learning module trained with CUB200-2011 ResNet-50 embeddings. 34

Table 4.5. Natural and adversarial performances of robust metric learning module trained with CARS196 ResNet-50 embeddings. 36

Table 4.6. Natural and adversarial performances of robust metric learning module trained with SOP ResNet-50 embeddings. 38

Table 4.7. Natural and adversarial performances of robust metric learning module trained with CUB200-2011 ResNet-18 embeddings. 39

Table 4.8. Natural and adversarial performances of robust metric learning module trained with CARS196 ResNet-18 embeddings. 41

Table 4.9. Natural and adversarial performances of robust metric learning module trained with SOP ResNet-18 embeddings. 42

Table 4.10. Adversarial robustness of different approaches including the proposed RML against adversarial samples synthesized by attacking alignment loss. 43

Table 4.11. Training time analysis for the proposed approach. Training time of the first epoch is measured as 0.58 minutes, and it is multiplied by 100 epochs for the fine-tuning of the pre-trained ResNet-50 model naturally. Adversarial attack generation is completed in 12.7 minutes. Robust metric learning is applied for 2.7 minutes. 44

Table 4.12. Training time comparisons. Training time for an epoch is calculated as 13 minutes and it is multiplied by 100 epochs for an adversarial training. 44

LIST OF SYMBOLS

d	Dimension size
$\mathcal{D}$	Data distribution
D	Distance metric
f	DNN model function
g	Deep metric learning model function
$\mathbf{h}$	Embedding of an input image
$\mathbf{H}$	Entropy
$\mathcal{L}$	Loss function
m	Margin value
n	Mini-batch size
S	Acceptable domain range
t	Number of attack iterations
T	Number of epochs
$\mathbf{x}$	Natural input image
$\mathbf{x}_{\mathrm{a}}$	Anchor input image
$\mathbf{x}_{\mathrm{comp}}$	Input image to be compared
$\mathbf{x}_{\mathrm{n}}$	Negative input image
$\mathbf{x}_{\mathrm{p}}$	Positive input image
$\mathbf{x}^{'}$	Adversarial image
y	Target label
Y	Negative/positive pair control
α	Target angle
β	Balancing term for adversarial loss
γ	Attack step size
δ	Adversarial perturbation amount
θ	Parameters of neural networks
$\prod$	Projection operation

τ	Learning rate
Υ	Ground truth classes
ϕ	Parameters of deep metric learning model
Ω	Set of clusters

LIST OF ACRONYMS/ABBREVIATIONS

ACT	Anti-Collapse Triplet
ADML	Adversarial Deep Metric Learning
ARL	Adversarial Robust Learning
AT	Adversarial Training
CBML	Contrastive Bayesian Metric Learning
CCL	Center Contrastive Loss
CNN	Convolutional Neural Network
DNN	Deep Neural Network
EARDML	AT-Based Adversarial Deep Metric Learning Model
EE	Embedding Expansion
FGSM	Fast Gradient Sign Method
FRO-ML	Robenius Norm Regularization
FT	Fine-tuned Model
HM	Hardness Manipulation
MI	Mutual Information
MTAXs	Multi-targeted Adversarial Examples
NMI	Normalized Mutual Information
PGD	Projected Gradient Descent
RML	Robust Metric Learning Module
SCE	Spherical Embedding Constraint
SGD	Stochastic Gradient Descent
SOP	Stanford Online Products
SOTA	State-of-the-art
SPE-ML	Spectral Norm Regularization
STAXs	Single-targeted Adversarial Examples
TLA	Triplet Loss Adversarial
TRUBA	Turkish National e-Science e-Infrastructure
TUBITAK	Scientific and Technological Research Council of Turkey

ULAKBIM	High Performance and Grid Computing Center
XBM	Cross-batch Memory

1. INTRODUCTION

Deep metric learning aims to learn a non-linear projection to a new space by considering the expressive similarity and distance between the samples. It aims to extract powerful representations that minimize the distance between semantically similar samples while maximizing the distance between dissimilar samples. Therefore, it has been an essential component of various computer vision tasks, such as image classification [1], image retrieval [2–4] and face recognition [5, 6]. For instance, few-shot image classification [7] benefits from deep metric learning by classifying unseen samples according to their distances to a few seen samples in an embedding space. In retrieval problems, powerful distance metrics are learned with the aim of obtaining the nearest neighbors similar to the query image [8]. Moreover, individuals with similar face characteristics can be distinguished better using distance-based metric learning approaches [5, 6].

State-of-the-art (SOTA) metric learning for vision is modeled with Convolutional Neural Network (CNN) architectures [9, 10]. Even though CNN architectures facilitate accomplishing various complex tasks with, Szegedy et al. [11] discussed that they are vulnerable to perturbations imperceptible to humans. Attackers synthesize such perturbations to fool models during their decision-making processes. For instance, Figure 1.1 shows the most similar neighbors of the query image in natural and adversarial settings. To generate the adversarial images in this experiment, an iterative Projected Gradient Descent (PGD) attack is synthesized by attacking the triplet loss, which focuses on increasing the distance between similar samples while getting closer the dissimilar ones. As a result, while all of the closest five neighbors share the same white car category with the query in the natural setting, a query with adversarial perturbations deceives the model into retrieving closest neighbors from cars with different colors and styles.

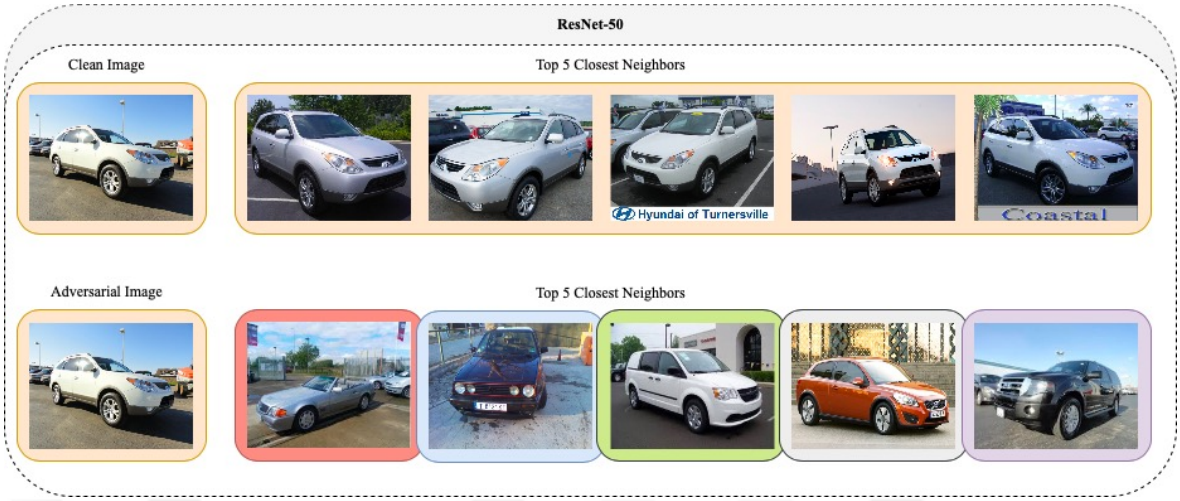


Figure 1.1. Top 5 nearest neighbors comparisons of ResNet-50 [12] embeddings in natural and adversarial settings. Each color represents the different class.

After Szegedy et al. [11] demonstrated intriguing characteristics of CNNs against adversarial attacks in object classification, vulnerability of deep networks used for different tasks and problems, including metric learning, has been investigated. Various studies have been conducted to enhance the robustness of models against adversarial attacks. Defensive distillation [13], deflecting adversarial attacks [14], adversarial sample purification [15], and regularization [16] can be the alternative methods to withstand malicious perturbations. However, like in other fields, adversarial training (AT) has also been the most preferred approach in metric learning due to its advanced capabilities.

AT refers to including adversarial sample generation in training to learn patterns in these attacks and to gain the capability of defending against them in real-world scenarios. In line with this idea, Madry et al. [17] posed the AT as a min-max optimization problem using an iterative Projected Gradient Descent (PGD) attack. AT performance is directly affected by the attack type and configurations during the training. For instance, when the most powerful attacks are applied, models may be prone to overfit these strong perturbation patterns. Furthermore, since labels are used in some approaches during the adversarial attack generation, models exploit the inherent regularities of adversarial images and perform better on adversarial samples than

the natural ones, which is called label leakage [18] problem. Therefore, several studies [19–21] are conducted to improve generalization and robustness by readjusting the adversarial attack generation process.

Despite the contributions of AT to model robustness, some concerns and limitations have been raised practice. One of these critical issues is increased computational costs due to the extra backpropagation steps required in attack generation during training. AT simultaneously solves two optimization problems: attack generation and the training of the complex architectures using these synthesized adversarial samples. Therefore, multiple forward and backward passes for these two optimization objectives cause a catastrophic rise in the training complexity.

Another drawback of AT is the trade-off between robustness and natural accuracy. Including adversarial samples during training might lead the deep model into learning and adapting the adversary patterns. In other words, while the model puts significant effort into learning representations invariant to adversarial perturbations, performance on natural samples might drop. However, a deep model with degraded natural performance would not serve the purpose, although it gains robustness.

AT evaluation performance heavily depends on the produced attack characteristics, and it cannot achieve desirable results if unseen data is forwarded through the network. For these reasons, training should consist of several diversified attacks to boost the generalization performance. However, it means that more and more training time is needed since training complexity is multiplied for each different attack manipulation. Besides, high resource capacities are necessary to perform these complicated training operations.

This thesis focuses on alleviating some of these drawbacks of AT and enhancing the adversarial robustness via a lightweight metric learning approach for images. The proposed robust metric learning model improves the defense mechanism against malicious attacks by minimizing the training time and maintaining the natural performance.

Contributions of the thesis are outlined in the following section.

1.1. Contributions of the Thesis

This thesis focuses on robustifying deep metric learning models against imperceptible perturbations by learning a mapping of the feature embeddings into a more robust higher or lower dimensional space. The key contributions that serve these ideas can be listed as follows:

- We introduce a lightweight, robust metric learning approach without implementing adversarial training. Training complexity and time, which arise from synthesizing adversaries at each iteration of adversarial training, are reduced by benefiting a black box approach. Our method produces and saves the adversarial samples one time before the robust metric learning training starts. Therefore, adversarial images are fed to the model in a black-box manner during metric learning.
- The proposed robust metric learning module does not rely on specific architectures; it can be easily combined with shallow or deep architectures. Besides, the proposed model considers a pre-trained architecture, fine-tunes it with the natural samples in the dataset of interest, and trains the robust metric learning model from scratch using the latent space of the fine-tuned architecture. Thus, the proposed framework can maintain the SOTA performance on the natural samples while introducing adversarial robustness.
- The proposed approach is evaluated with ResNet-18 [12] and ResNet-50 [12], which are commonly used in deep metric learning literature to investigate the role of depth in the problem. We observe consistent behavior of the proposed approach when the two architectures are used with various benchmark datasets.
- Compared to adversarial fine-tuning often used in deep metric learning literature, the proposed approach is better at alleviating the gap between the performances of the natural and robust models on natural samples for all the experiments with different architectures and benchmark datasets.

- Various metric learning objectives, such as contrastive, triplet, and angular loss functions, are utilized during the experiments to investigate the effect of the loss function on the adversarial robustness of metric learning. The performances of these loss functions are compared under different experimental scenarios to provide a clear understanding.
- The proposed approach is evaluated with the widely used benchmark datasets in metric learning studies, which are CUB200-2011 [22], CARS196 [23], and Stanford Online Products (SOP) [3]. The benchmark datasets pose various challenges regarding the number of categories, samples, and inter and intra-class similarities. The experimental results show that the proposed method can contribute to the robustness of the deep metric learning model for each benchmark dataset with different characteristics.

2. RELATED WORK

2.1. Adversarial Attacks

Adversarial attacks in computer vision tasks aim to fool deep learning models by adding human-imperceptible perturbations to input images. Such attacks can be categorized based on their generation purpose. Based on the attacker's knowledge of the target model, the three most prominent attack settings can be classified as white-box, gray-box, and black-box adversarial attacks. Under the white-box setting, attackers have full access to the weights of the trained model, the dataset, and the architecture [24]. Therefore, attackers can explore the most critical weaknesses of models and apply the perturbations more effectively by using these weaknesses. Fast Gradient Sign Method (FGSM) [25] and Projected Gradient Descent (PGD) [17] can be considered as some of the most popular white-box attacks.

Secondly, in the gray-box setting, the attacker has limited access to the details of the model. For instance, learned model weights might be unknown while the architecture of the target model is accessible [26]. Various publicly shared pre-trained models can help generate gray-box adversarial attacks. Lastly, the black-box setting constructs adversarial perturbations without the knowledge of the target model. However, the output of the target model can be used to generate a black-box [27] attack. The Fast Gradient Sign Method (FGSM) [25] and Projected Gradient Descent (PGD) [17], considered some of the most potent white-box attacks, are further explained in the following section.

2.1.1. Fast Gradient Sign Method (FGSM)

Fast Gradient Sign Method (FGSM) [25] applies the minimum amount of perturbation to input images for fooling the classification models. This attack computes the gradient of the loss function with respect to the input image once and perturbs the

input using the direction of the gradient in a single step. It increases adversarial loss in order to generate adversarial images, $\mathbf{x}^{'}_{FGSM}$, and can be demonstrated as

$$\mathbf{x}^{'}_{\text{FGSM}} = \mathbf{x} + \epsilon \cdot \text{sign}(\nabla_{\mathbf{x}} \mathcal{L}(\mathbf{x}, y, \boldsymbol{\theta})), \tag{2.1}$$

where $\boldsymbol{\theta}$ is the parameters of the neural network, ∇ represents the gradients of the adversarial loss $\mathcal{L}(\mathbf{x}, y, \boldsymbol{\theta})$ with respect to the natural image $\mathbf{x}$, ϵ controls the amount of perturbation, y denotes the target.

2.1.2. Projected Gradient Descent (PGD)

Unlike FGSM, Projected Gradient Descent (PGD) [17] adversarial attack is generated by iteratively perturbing the input using the gradients of the loss with respect to the input. Due to this iterative nature, PGD is considered more powerful and complex than other types of attacks. This attack can be produced as

$$\mathbf{x}^{t+1}_{\text{PGD}} = \prod_{\mathbf{x}+S} \left(\mathbf{x}^{t}_{\text{PGD}} + \gamma \cdot \text{sign} \left(\nabla_{\mathbf{x}^{t}_{\text{PGD}}} \mathcal{L} \left(\mathbf{x}, y, \boldsymbol{\theta} \right) \right) \right), \tag{2.2}$$

where the number of PGD steps is denoted by t, attack step size is represented by γ. Projection operation, $\prod_{\mathbf{x}+S}$, controls the maximum and minimum acceptable domain ranges.

One of the prominent challenges of iterative attacks is controlling the amount of perturbation. Since these attacks add perturbations cumulatively, they can exceed the input domain ranges. Clipping operations are applied during the PGD attack generation to keep the pixel values within the input domain and the perturbation amount within the set of allowed perturbations. As the strength of the gradient-based attack increases, the diversity of the attacks might decrease due to the clipping operations. Consequently, strong PGD attacks may become dependent on certain perturbations that hurt the generalizability of adversarial training, explained in the following section. Auto-Attack [28] proposed by Croce and Hein addressed this problem by combining PGD with white-box Fast Adaptive Boundary and black-box square attacks. Experimental results showed that the ensemble attack competitively decreases

the robust accuracies of SOTA models. Besides, the Auto-Attack eliminates the need for hyperparameter tuning in PGD by automatically optimizing the hyperparameters.

2.2. Adversarial Robustness

Before Szegedy's [11] first study about the adversarial samples, neural networks were only trained and evaluated on natural samples, which are not perturbed by any adversarial attack, to measure their performances. However, after the vulnerability of CNNs against adversarial attacks was revealed, a research field focusing on training more robust models against adversarial perturbations was born. The adversarial robustness studies [17], [25] report performances on natural and adversarially perturbed inputs. For instance, adversarial accuracy is used to explain a Deep Neural Network's (DNN) ability to maintain its natural model performances under adversarial attack settings. Since DNNs are heavily used in safety-critical applications in various domains, from health [29] to real-time driverless vehicles [30], evaluating adversarial robustness has gained significance.

Various defense techniques, including defensive distillation [13], [31], deflecting adversarial attacks [14], adversarial sample purification [15], and regularizing deep classifiers with loss functions [16], [32], are proposed in literature. In defensive distillation [13], [31], the outputs of the original DNN named teacher model are softened by using the temperature hyper-parameter. Then, the student model, is trained with these softened relationships. Since the student model consists of soft labels, it might provide smoother and better generalization performances. Prakash's [14] study redistributes the input image pixels to soften manually crafted adversarial perturbations. While some methods [15] project the adversarial input images to eliminate the effect of perturbations, various regularization parameters [16], [32] can be included in loss functions to obtain more stable training and better performance results.

Adversarial Training (AT) and its variants are fundamentally studied robust training techniques among the adversarial defense approaches. In AT approaches,

the training dataset might be augmented by natural and adversarial samples, or only adversarial samples might be used in training to increase the adversarial accuracy. AT benefits from different types of attacks, which might be gradient-based or generative-based attacks.

AT proposed by Goodfellow et al. [25] augments the training set with natural and adversarial samples generated with FGSM during training. The gradient of the cross-entropy loss is computed, and the loss function is composed of two cross-entropy terms, one computed with natural samples and the other with adversarial samples. The mathematical formulation of this approach can be expressed as

$$\mathcal{L}_{\text{Total}}(\boldsymbol{\theta}, \mathbf{x}, y) = \beta \cdot \mathcal{L}(\boldsymbol{\theta}, \mathbf{x}, y) + (1 - \beta) \cdot \mathcal{L}(\boldsymbol{\theta}, \mathbf{x} + \epsilon \cdot \text{sign}(\nabla_x \mathcal{L}(\boldsymbol{\theta}, \mathbf{x}, y)), y), \quad (2.3)$$

where $\mathbf{x}$ represents the natural samples, cross-entropy loss is defined as $\mathcal{L}$, $\boldsymbol{\theta}$ denotes the model parameters. β balances the effect of adversarial augmentations into the overall loss function.

AT by Madry et al. [17] trains deep convoluational neural networks with only adversarial samples generated by PGD attacks. Like Goodfellow's AT [25], the attack is computed with the cross-entropy loss during training. However, Madry's AT [17] trains with a single cross-entropy term computed with only adversarial samples, providing a higher adversarial test accuracy. This AT can be described as a minimax problem. While optimal perturbations are explored by maximizing the inner adversarial loss function, the outer part minimizes the adversarial loss function to obtain optimal model weights. This structure can be formulated as

$$\mathcal{L}_{\text{AT}}(\boldsymbol{\theta}, \mathbf{x}, y) = \min_{\boldsymbol{\theta}} \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}} \left[\max_{\boldsymbol{\delta}\in S} \mathcal{L}\left(f\left(\mathbf{x} + \boldsymbol{\delta}, \boldsymbol{\theta}\right), y\right) \right], \quad (2.4)$$

where natural image $\mathbf{x}$ and their target label y constitute the data distribution $\mathcal{D}$. Parameters of a DNN, $f(\cdot, \boldsymbol{\theta})$, are represented by $\boldsymbol{\theta}$. Adversarial perturbation amount, δ, is controlled by predetermined allowance limits S. Other AT approaches where state-of-the-art robustness is achieved with less number of attack steps are also proposed in the literature [21], [33, 34].

2.3. Deep Metric Learning

Deep metric learning aims to learn a non-linear projection to a new space where expressive similarity and distance between samples can be measured. Therefore, it has been an essential component of various computer vision tasks, such as image classification [1], image retrieval [2–4] and face recognition [5, 6]. CNNs for metric learning are trained to learn the low-dimensional feature representations of the high-dimensional input images such that the relationships between the input samples are preserved and enhanced in the low-dimensional space. For instance, if two samples belong to the same category, their distance in the CNN's latent space should be smaller than the distance between samples from different categories.

Deep metric learning studies can be divided into two subcategories according to their sample selection procedures: pair-based and proxy-based methods. Pair-based metric learning methods are heavily used in the literature. They focus on getting similar samples closer while pushing away the dissimilar items using the mini-batch pairs. Depending on the problem and architecture, triplets [5] or even quadruplets [35] can be included in different loss functions instead of using only the pairs. The most popular loss functions in the metric learning domain are further explained in detail below.

In contrastive loss, pair relationships are established between the anchor and comparison point to train the architectures. The loss function forces the distance between the positive pairs to be minimized, whereas the distance between the negative pairs to be larger than the positive distances. The contrastive loss [36] can be formulated as

$$\begin{aligned}\mathcal{L}_{\text{Contrastive}}\left(\mathbf{x}, \mathbf{x}_{\text{comp}}; \phi\right) = \min_{\phi} \frac{1}{N} \sum_{i=1}^{N} \Big[&(1-Y) D\left(g\left(\mathbf{x}^{\text{i}}; \phi\right), g\left(\mathbf{x}^{\text{i}}_{\text{comp}}; \phi\right)\right)^2 \\ &+ Y\left(m - D\left(g\left(\mathbf{x}^{\text{i}}; \phi\right), g\left(\mathbf{x}^{\text{i}}_{\text{comp}}; \phi\right)\right)^2\right)\Big],\end{aligned} \tag{2.5}$$

where $g\left(\cdot; \phi\right)$ denotes a DNN parametrized by ϕ, $\mathbf{x}$ is the input image and $\mathbf{x}_{\text{comp}}$ illustrates the comparison point. If positive pair distance D is computed, Y is marked

as 0. On the other hand, if negative pairs are trained, Y is selected as 1. A predetermined margin, m is set for the distance between negative pairs.

There are numerous studies in the literature to improve deep metric learning generalization performances. Contrastive loss is one of the promising pair-based loss functions to shed light on these problems. Wang et al. [37] proposed cross-batch memory (XBM) mechanism to address the constraint of being dependent only on the current mini-batch. XBM's enqueue-dequeue structured memory bank paves the way for updating contrastive loss using the information from both the previous and the current mini-batches. While XBM keeps each sample embedding directly, Cai et al. [38] introduced a Center Contrastive Loss (CCL) method to store only the representative class centers in its center bank. This study combines contrastive loss with center constraint to reduce intra-class variations. On the other hand, Contrastive Bayesian Metric Learning (CBML) developed by Kan et al. [39] achieves better generalization performances by reducing the semantic differences between the intermediate feature layer and class labels at the final output layer. Conditional relations from intermediate features to the last output layer are established and integrated with the contrastive loss function. Since the increasing number of conditions may boost the possibility of over-fitting, CBML also restricts the metric variances in an acceptable range.

One of the popular pair-based metric learning loss functions is triplet loss, which updates the gradients using triplets. The main difference between contrastive and triplet loss arises from the number of samples used for updating the gradients. Triplet and contrastive loss benefit from three and two samples, respectively. Triplets consist of anchor, positive and negative samples. In addition, during training, anchor and positive samples are brought closer while the anchor and negative points are pushed away from each other. Triplet loss [5] can be written as

$$\begin{aligned}\mathcal{L}_{\text{Triplet}}\left(\mathbf{x}_{\text{a}}, \mathbf{x}_{\text{p}}, \mathbf{x}_{\text{n}}; \phi\right) = \min_{\phi} \frac{1}{N} \sum_{i=1}^{N} \big[D\left(g\left(\mathbf{x}_{\text{a}}^{\text{i}}; \phi\right), g\left(\mathbf{x}_{\text{p}}^{\text{i}}; \phi\right)\right) \\ - D\left(g\left(\mathbf{x}_{\text{a}}^{\text{i}}; \phi\right), g\left(\mathbf{x}_{\text{n}}^{\text{i}}; \phi\right)\right) + m \big]_{+} ,\end{aligned} \tag{2.6}$$

where $\mathbf{x}_{\mathrm{a}}^{\mathrm{i}}$, $\mathbf{x}_{\mathrm{p}}^{\mathrm{i}}$, $\mathbf{x}_{\mathrm{n}}^{\mathrm{i}}$ denote anchor, positive and negative pairs, respectively. D demonstrates the distance between selected positive and negative pairs.

Recent studies using the triplet loss are concentrated on to improve generalization performances using some techniques like data augmentation and data splitting. Ko and Gu [40] proposed an Embedding Expansion (EE) technique to augment and enrich the data in an embedding space. These produced synthetic samples are given to triplet loss to be able to extract more meaningful and representative embeddings. From the different point of view, divide and conquer algorithm developed by Sanakoyeu et al. [4] concentrates on smaller parts of the existing images to eliminate unnecessary factors from the latent space representations. In the end, these divided parts are conquered and triplet loss updates the gradients using this conquered embedding.

Finally, unlike the contrastive and triplet loss, angular loss considers the angles between embeddings instead of the predetermined fixed distances. Different classes may have different intra and inter-class variations. Therefore, in some cases, using fixed margins for each class is not sufficient. Angular loss focuses on this problem during training. Besides, triplet loss does not directly control the relationship between positive and negative points. It relates these points with respect to the anchor sample. At this point, angular loss can be a promising alternative to adapt negative and positive sample gradients simultaneously by considering each other. Angular loss [41] is expressed as

$$\mathcal{L}_{\text{Angular}}\left(\mathbf{x}_{\mathrm{a}}, \mathbf{x}_{\mathrm{p}}, \mathbf{x}_{\mathrm{n}}; \phi\right) = \min_{\phi} \frac{1}{N} \sum_{x_a} \log \left(1 + \sum_{\substack{x_n \in B \\ y_n \neq y_a, y_p}} \exp(f_{a,p,n}) \right), \tag{2.7}$$

$$\begin{aligned} f_{a,p,n} = & \, 4 \tan^2 \alpha \left(g\left(\mathbf{x}_{\mathrm{a}}; \phi\right) + g\left(\mathbf{x}_{\mathrm{p}}; \phi\right) \right)^T g\left(\mathbf{x}_{\mathrm{n}}; \phi\right) \\ & - 2(1 + \tan^2 \alpha) g\left(\mathbf{x}_{\mathrm{a}}; \phi\right)^T g\left(\mathbf{x}_{\mathrm{p}}; \phi\right), \end{aligned} \tag{2.8}$$

where y represents class labels, α defines the target angle between sample points. $g\left(\cdot; \phi\right)$ denotes a DNN parametrized by ϕ. f shows a function to obtain the predetermined target angle α.

After the discovery of the angular loss function, several studies have considered angular space characteristics of their approaches to get a chance to improve their generalization performances. While some have integrated final embeddings directly into the angular space, others have defined their pair-based loss function margins using angular distances. For instance, Zhang et al. [42] have added Spherical Embedding Constraint (SCE), which directly provides angular embedding norm adjustment during the iterations. In this way, they can control hyper-sphere surface placements, while conventional loss functions can only arrange the magnitude and direction of the embeddings. On the other hand, Yu and Tao [43] have added a slack margin in an angular space to prevent over-fitting issues during the training.

While pair-based loss functions directly utilize sample data in pairs or triplets, proxy-based methods determine class-based representative points and update the sample points with respect to these proxies. These methods struggle for getting closer each data point with its proxy, whereas proxies of different classes are pushed away from the sample data. Computational burden is decreased via minimizing required number of comparisons. Early studies [44] were benefiting from only one proxy for each class. Therefore, if samples within the class do not have similar distributions, they cannot produce a representative proxy for that class. This problem is addressed by SoftTriple loss developed by Qian et al. [45], which uses multiple proxies to represent intra-class variances. While these one or multiple proxy-based methods take advantage of fast convergence, they do not have the ability to explore data-to-data relations. Kim et al. [40] introduced a Proxy-Anchor that combines proxy and pair-based losses. This method calculates the relative hardness of each sample to its proxy, and it updates both proxy and other relative points according to the magnitude of this hardness. Even though this model provides promising results, it requires the calculation of many data-to-data relations. Chan et al. [46] proposed a method using proxy-proxy relations, which reduces computational costs with a better generalization performance.

2.4. Adversarial Defenses in Deep Metric Learning

While enhancing the performance of deep metric learning models on natural samples is necessary, adversarial robustness should also be gained since deep metric learning is not inherently robust against adversarial attacks. Therefore, in recent years, some studies have started investigating this area to train more robust models in classification, clustering, and image retrieval tasks.

Firstly, Mao et al. [16] developed a Triplet Loss Adversarial (TLA) method to boost the classification accuracy of models in adversarial settings. For this aim, they trained the network architecture from scratch while regularizing cross-entropy loss with an adversarial triplet loss in the latent space. Their method attacks cross-entropy loss to produce PGD adversarial samples during the training, and these samples are included as anchor points in the triplet loss. Instead of regularizing the cross entropy with adversarial loss, Zhou et al. [47] proposed the Anti-Collapse Triplet (ACT) technique, which directly utilizes triplet loss to increase the robustness of models. They have generated positive and negative adversarial samples by collapsing them during training. Since adversarial attack struggles to reduce the distance between positive and negative adversarial samples, misleading gradient and insufficient mini-batch exploitation problems [48], which arise from arbitrary generation of adversaries, are addressed in a controlled way.

Later, many studies generated more informative adversarial samples to increase deep metric learning model robustness. They have changed prevalent adversarial loss functions with other objectives or adjusted them with upper and lower boundaries. Panum et al. [49] attacked contrastive and triplet loss functions instead of cross-entropy loss for producing PGD adversarial attack. This work fine-tunes the complex architectures with adversarial training and uses a manually crafted synthetic dataset that declines undesired shifts during the PGD attack generation. On the other hand, Zhou and Patel [50] introduced the Hardness Manipulation (HM) technique to control the adversary generation process with lower and upper bound limits during adversarial

training. HM focuses on directing the source hardness, the distance between the anchor and positive minus the distance between anchor and negative, to destination hardness while generating adversarial samples.

As similar to idea introduced in HM, Huai et al. [51] consisted of spectral norm regularization (SPE-ML) or frobenius norm regularization (FRO-ML) as an upper bound. Moreover, they benefit from pairwise adversarial loss instead of triplet loss and this loss function is restricted by regularized upper bound. Minimization of the upper bound contributes to minimization of the attacker's exploration space and adversarial robustness is increased. To be clear about the main idea, baseline pairwise adversarial loss function without upper bounds can be expressed as

$$\begin{aligned}\mathcal{L}_{\text{PairwiseAdv}}\left(\mathbf{x}, \mathbf{x}_{\text{comp}}; \phi\right) = \min_{\phi} \frac{2}{N(N-1)} \sum_{i=1}^{N} \max\{0, 1 \\ + Y(D(g\left(\mathbf{x}^{\text{i}}; \phi\right), g\left(\mathbf{x}^{\text{i}}_{\text{comp}}; \phi\right)) \ - m)\},\end{aligned} \tag{2.9}$$

where $Y \in \{-1, 1\}$. While it is denoted as -1 if the class labels of $\mathbf{x}$ and $\mathbf{x}_{\text{comp}}$ are different, it is represented with 1 if $\mathbf{x}$ and $\mathbf{x}_{\text{comp}}$ are samples from the class.

As mentioned in the previous works, pair-based triplet and contrastive losses are mainly used in the robust metric learning domain. Apart from this idea, Appiah et al. [52] interestingly combined the differentiable decision trees with distance-based pairwise metric learning. They claimed that differentiable decision trees explore both relative position information as done by pair-based losses and the absolute pattern of the whole dataset. Therefore, much more information is provided to DNNs for classification tasks. During training, white-box and black-box attacks are synthesized by getting the adversarial samples as close as possible to their benign inputs.

Data augmentation is a technique for enhancing the training dataset by adding variations to natural inputs. While this method is widely used for training DNNs, it has recently been adapted into a robust metric learning domain. Singh et al. [53] leverages single-targeted adversarial examples (STAXs) and multi-targeted adversarial

examples (MTAXs) to create inputs coming from multiple distributions. While STAXs are shifted through one distinct class, MTAXs are directed to an overlap area containing several classes. Therefore, MTAXs provide more information to clarify these overlap areas and generalize better on unseen data when compared with STAXs. After these adversarial samples are synthesized during the training, they are used in adversarial training mechanisms to boost performance against PGD attacks. This method can be combined with all pair-based or proxy-based metric learning loss functions.

As one more example of data augmentation usage in robust metric learning, Chen et al. [54] developed the adversarial robust learning (ARL) method by adapting both natural random and constrained adversarial augmentations into contrastive loss. ARL attacks to the contrastive loss for generating PGD attack during the adversarial training. Unlike previous approaches, this study calculates the distance between synthesized adversarial samples of different classes in the latent space. It minimizes this distance in the latent space to generalize better against adversarial perturbations.

In brief, there are varying approaches in the metric learning literature to increase the model robustness against adversarial samples. As a similarity of these methods, their approaches and our study configure the adversarial attack generation process in such a way that positive and negative samples are pulled together for deceiving the models. Although the primary purpose of adversarial attack generation is similar, special boundaries or loss functions can be adjusted depending on the method. In contrast to approaches using adversarial augmentation and adversarial training, the proposed method in this thesis contributes to adversarial robustness via training a lightweight neural network.

3. METHOD

This section explains our proposed model for improving the robustness of deep metric learning against adversarial attacks in detail. The proposed model consists of an embedding and a metric learning module, as illustrated in Figure 3.1. While the embedding module extracts the representative embeddings of input images using complex architectures, the robust metric learning (RML) module maps these embeddings into a new space where the semantic relationships encoded in the input embeddings are preserved but not affected by the adversarial perturbations leaked into the embeddings of the adversarial samples.

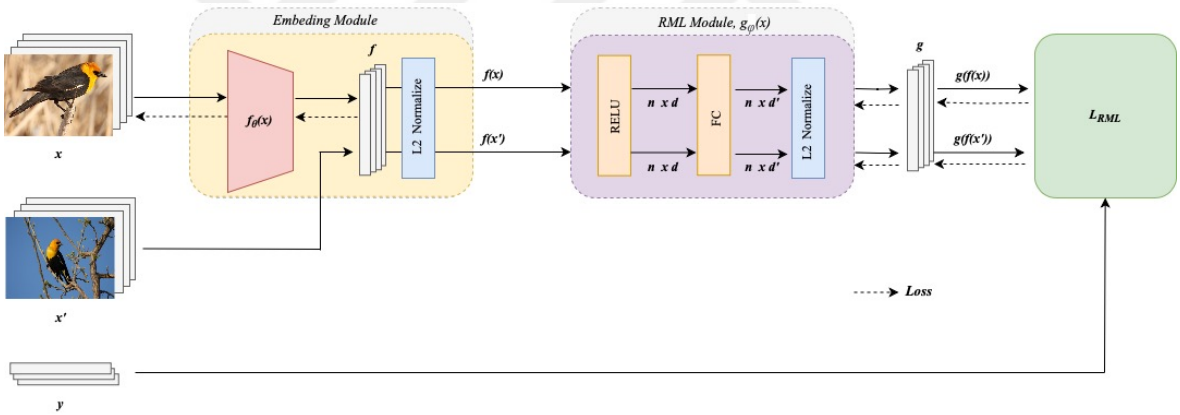


Figure 3.1. Proposed robust deep metric learning model. Embeddings of natural, $f(\mathbf{x};\theta) \in \mathbb{R}^d$, and adversarial images, $f\left(\mathbf{x}^{'};\theta\right) \in \mathbb{R}^d$, are extracted using embedding module. The embedding module is frozen, while the metric learning module is training. The outputs of the metric learning module, $(g(f(\mathbf{x});\phi) \in \mathbb{R}^{d^{'}}$ and $\left(g(f\left(\mathbf{x}^{'}\right);\phi\right) \in \mathbb{R}^{d^{'}}$, are provided to related loss function.

It can be emphasized that the above-mentioned proposed architecture is trained by a four-step framework, as shown in Figure 3.2. To be clear about the architecture implementation, details of the embedding module, fine-tuning of an embedding module, adversarial sample generation, and robust metric learning module are presented in the following sections.

Require: Natural data: $\mathcal{D}_{\text{nat}} = \{(\mathbf{x}_1, y_1), \cdots, (\mathbf{x}_N, y_N)\}$,
adversarial data: $\mathcal{D}_{\text{adv}}$,
data embeddings: h
Ensure:
Fine-tuning: $f \leftarrow \text{FinetuneResNet}(D_{nat})$
Adversarial attack generation: $\mathcal{D}_{\text{adv}} \leftarrow \text{PGDAttack}(f(D_{\text{nat}}, \theta))$
Embedding module: $\mathbf{h}_{\text{nat}}, \mathbf{h}_{\text{adv}} \leftarrow \text{GetEmbedding}(D_{\text{nat}}, D_{\text{adv}})$
Triplet sampling: anchor: $\mathbf{h}_{\text{adv}}$, positive: $\mathbf{h}_{\text{p}}$, negative: $\mathbf{h}_{\text{n}}$
for t = 1, ... , T **do**
  for mini-batch $\{\mathbf{x}_i, \boldsymbol{y}_i\}_{i=1}^{n} \sim$ D **do**
    model update:
    $\phi \leftarrow \phi - \tau \cdot \frac{1}{n} \sum_{i=1}^{n} \nabla_\theta \mathcal{L}_{\text{RML}} \left(g\left(\mathbf{h}_{\text{adv}}^{\text{i}}; \phi\right), g\left(\mathbf{h}_{\text{p}}^{\text{i}}; \phi\right), g\left(\mathbf{h}_{\text{n}}^{\text{i}}; \phi\right)\right)$
  end for
end for

Figure 3.2. Adversarial metric learning framework. Training for T epochs, θ embedding module parameters, ϕ metric learning module parameters, τ learning rate, D dataset, n mini-batch size.

3.1. Embedding Module

The embedding module in Figure 3.1 is comprised of fine-tuning a pre-trained DNN and extracting the image embeddings from the fine-tuned architecture.

In recent years, adversarial robustness in object recognition tasks has reached a robust accuracy of around 70% against powerful Auto-Attack [28] as demonstrated in RobustBench [55] leader board for adversarial robustness. The prominent approach in the best-performing methods is using a vast number of synthetic images during adversarial training. Since the reason for vulnerability against adversarial attacks can also be posed as an overfitting problem, using millions of synthetic samples enhances the model's generalization property. However, the computational cost of generating and training with such large-scale synthetic datasets requires high-capacity hardware

and memory. Therefore, this thesis aims to introduce robustness to a DNN trained or fine-tuned for metric learning without training deep architectures with adversarial sample generation and synthetic samples from scratch.

Under the low data regime, embeddings can be extracted directly from a pre-trained network. However, fine-tuning might be essential due to the data shift between the source and target datasets. For instance, deep metric learning literature often prefers fine-tuning ResNet [12] architectures trained on ImageNet [56] dataset [57–59]. Therefore, pre-trained ResNet50 and ResNet18 architectures [12] are used in the embedding module in this thesis. Fine-tuning can be done with a retrieval or a classification objective. In this thesis, the cross-entropy loss is used during fine-tuning of the embedding network inspired by Boudiaf et al. [58], who suggest that fine-tuning with cross-entropy loss is more effective and efficient than pair-based metric learning objectives. Defining optimization bounds in cross-entropy loss boosts the performance. Since mining and forming triplets are not necessary, fine-tuning with cross-entropy loss is considered more practical.

After completing the fine-tuning process for each dataset, embeddings are extracted, normalized and saved to be used in the following robust metric learning module. Moreover, the role of embedding extraction network for PGD attack generation is explained in the following adversarial sample generation section in detail.

3.2. Adversarial Sample Generation

Adversarial training can be defined as the most popular technique to defend deep learning models against adversarial attacks. Many studies in the metric learning domain also try to improve adversarial model performances using AT technique. Even though AT provides promising robustness performances, several backpropagation steps required for adversarial sample generation like demonstrated in the Equation (2.1) and Equation (2.2). These backpropagation steps cause an excessive computational burden on models, especially for the large scale datasets.

This study aims to transform the learned embedding space vulnerable to adversarial perturbations into a more robust space. Apart from the previous approaches in the metric learning domain, we propose generating adversarial samples after training the embedding module. Generated adversarial samples are saved in order to train them in the next metric learning module. The computational costs and training time are reduced since the deep embedding module is not also trained with adversarial samples.

PGD is one of the most effective adversarial attacks because of its iterative nature. Therefore, this thesis uses the PGD attack as the primary attack in robust deep metric learning. Adversarial samples are generated with the PGD formulation in Equation (2.2). On the other hand, unlike the object recognition task, the gradient of the triplet loss with respect to the anchor images is used as the perturbation direction. Thus, the adversarial sample generation process is explained in Figure 3.3 in detail.

3.3. Robust Metric Learning Module

This module describes the robust metric learning module in Figure 3.1 with model update sub-task from Figure 3.2. The architecture of the proposed model, pair-based loss functions, and sampling strategies are explained further within this module.

Adversarial attacks push similar samples away from each other while reducing the distance between the adversarial sample and a sample from a different category. While such perturbations can force embeddings of the adversarial samples from the same class to cluster together far away from their natural counterparts, they can also distribute the adversarial embeddings across the space without any high-level structure. To be clear, Figure 3.4 visualizes outputs of an embedding module for randomly selected natural and adversarial samples from CUB200-2011 dataset. It is apparent that adversarial samples are moving away from their natural ones, while they are getting closer to another natural dissimilar samples. As a result, the proposed robust metric learning framework focuses on restoring the semantic relationships between adversarial and natural samples while leveraging the latent space of a well-trained embedding network.

Require: Natural data: $\mathcal{D}_{\text{nat}} = \{(\mathbf{x}_1, y_1), \cdots, (\mathbf{x}_N, y_N)\}$,
adversarial data: $\mathcal{D}_{\text{adv}} = \{(\mathbf{x}^1_{adv}, y_1), \cdots, (\mathbf{x}^N_{adv}, y_N)\}$
Ensure:
for mini-batch $\{\mathbf{x}_i, \boldsymbol{y}_i\}_{i=1}^n \sim$ D **do**
$\mathbf{x}_{\text{a}}, \mathbf{x}_{\text{p}}, \mathbf{x}_{\text{n}} \leftarrow$ anchor, positive, negative images
$\mathbf{x}_{\text{adv}} \leftarrow \mathbf{x}_{\text{a}}$
for m = 1, ... , M **do**
$\mathbf{h}_{\text{adv}}, \mathbf{h}_{\text{p}}, \mathbf{h}_{\text{n}} \leftarrow f(\mathbf{x}_{\text{adv}}; \theta), f(\mathbf{x}_{\text{p}}; \theta), f(\mathbf{x}_{\text{n}}; \theta)$
$\mathbf{x}_{\text{adv}} = (\mathbf{x}_{\text{adv}} + \gamma \cdot \text{sign}(\nabla_{x_{adv}} \mathcal{L}_{\text{Triplet}}(\mathbf{h}_{\text{adv}}, \mathbf{h}_{\text{p}}, \mathbf{h}_{\text{n}})))$
$\boldsymbol{\delta} = \max(\min(\mathbf{x}_{\text{adv}} - \mathbf{x}_{\text{a}}, \epsilon), -\epsilon)$
$\mathbf{x}_{\text{adv}} = \mathbf{x}_{\text{a}} + \boldsymbol{\delta}$
end for
end for

Figure 3.3. PGD attack generation [17] framework. D dataset, n mini-batch size, θ is the parameters of the embedding network, $\nabla_{x_{adv}}$ represents the gradient of the triplet loss, $\mathcal{L}_{\text{Triplet}}$, with respect to an anchor image $\mathbf{x}_{\text{adv}}$, ϵ controls amount of perturbation, the number of PGD steps is denoted by m, γ is the attack step size.

A lightweight metric learning module is trained with the embeddings of natural and adversarial images. As shown in Figure 3.1, the metric learning module is a multi-layer perceptron with ReLU hidden layers that map the original embeddings of natural, $f(\mathbf{x}; \theta) \in \mathbb{R}^d$, and adversarial images, $f(\mathbf{x}'; \theta) \in \mathbb{R}^d$, into a lower or higher dimensional space where $(g(f(\mathbf{x}); \phi) \in \mathbb{R}^{d'}$, $(g(f(\mathbf{x}'); \phi) \in \mathbb{R}^{d'}$ and $d' \neq d$. Therefore, first, the embeddings of natural and adversarial images that are not robust against adversarial attacks are extracted from the embedding module. Then, the robust metric learning module transforms ResNet embeddings into a new space by learning the semantic similarities between natural and adversarial samples. These embeddings in the new latent space are then fed into various loss functions used in the metric learning domain to compare their evaluation performances.

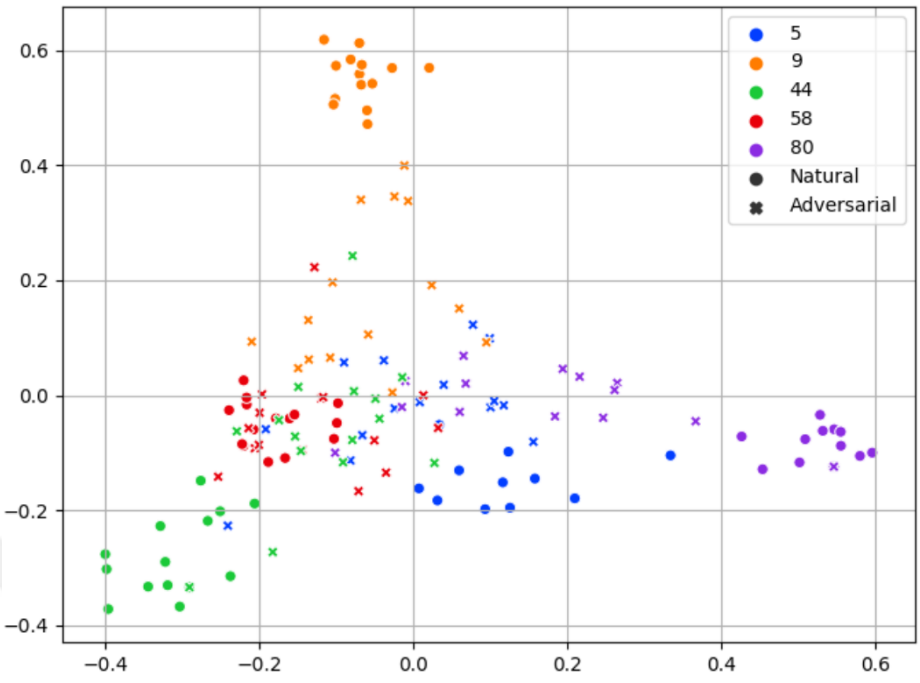


Figure 3.4. T-sne of natural and adversarial embeddings for a sample from CUB200-2011. Adversarial samples move away from their natural counterparts, while reducing the distance between the adversarial and natural samples from different categories.

In this thesis, pair-based loss functions, which are contrastive loss in Equation (2.5), triplet loss in Equation (2.6), and angular loss in Equation (2.7) are revised for robust adversarial metric learning. Particularly, anchor point is selected from adversarial images, whereas natural samples are assigned to comparison points. Revised robust metric learning contrastive loss can be minimized as

$$\begin{aligned}\mathcal{L}^{\text{RML}}_{\text{Contrastive}}\left(\mathbf{h}_{\text{adv}}, \mathbf{h}_{\text{comp}}; \phi\right) = \frac{1}{N}\sum_{i=1}^{N}\Big[&(1-Y)D\left(g\left(\mathbf{h}^{\text{i}}_{\text{adv}}; \phi\right), g\left(\mathbf{h}^{\text{i}}_{\text{comp}}; \phi\right)\right)^2 \\ &+Y\left(m - D\left(g\left(\mathbf{h}^{\text{i}}_{\text{adv}}; \phi\right), g\left(\mathbf{h}^{\text{i}}_{\text{comp}}; \phi\right)\right)^2\right)\Big],\end{aligned} \tag{3.1}$$

where $\mathbf{h}^{\text{i}}_{\text{adv}}$ represents the adversarial anchor, $\mathbf{h}^{\text{i}}_{\text{comp}}$ illustrates the comparison point and $\mathbf{h} = f\left(\mathbf{x}; \theta\right) \in \mathbb{R}^d$.

Secondly, the robust metric learning module is trained with the input triplets, where the anchor is sampled from the adversarial embeddings. A positive pair is formed

between adversarial embedding and the natural embedding of another image with the same label. On the other hand, the negative pair is constructed with the embedding of a randomly sampled image from a different category than the anchor image. Triplet loss used in the robust metric learning module can be expressed as

$$\begin{aligned}\mathcal{L}_{\text{Triplet}}^{\text{RML}}\left(\mathbf{h}_{\text{adv}}, \mathbf{h}_{\text{p}}, \mathbf{h}_{\text{n}}; \phi\right) = \min_{\phi} \frac{1}{N} \sum_{i=1}^{N} \big[D\left(g\left(\mathbf{h}_{\text{adv}}^{\text{i}}; \phi\right), g\left(\mathbf{h}_{\text{p}}^{\text{i}}; \phi\right)\right) \\ - D\left(g\left(\mathbf{h}_{\text{adv}}^{\text{i}}; \phi\right), g\left(\mathbf{h}_{\text{n}}^{\text{i}}; \phi\right)\right) + m \big]_{+},\end{aligned} \tag{3.2}$$

where $\mathbf{h}_{\text{p}}^{\text{i}}$, $\mathbf{h}_{\text{n}}^{\text{i}}$ denote positive and negative learned embeddings, respectively.

Finally, angular loss explained in Equation (2.7) is adjusted to our proposed adversarial metric learning module. Angular loss can be written as

$$\mathcal{L}_{\text{Angular}}^{\text{RML}}\left(\mathbf{h}_{\text{adv}}, \mathbf{h}_{\text{p}}, \mathbf{h}_{\text{n}}; \phi\right) = \min_{\phi} \frac{1}{N} \sum_{x_{adv}} \left\{ \log \left(1 + \sum_{\substack{x_n \in B \\ y_n \neq y_{adv}, y_p}} \exp(f_{adv,p,n}) \right) \right\}, \tag{3.3}$$

$$\begin{aligned}f_{a,p,n} = 4 \tan^2 \alpha (g\left(\mathbf{h}_{\text{adv}}; \phi\right) + g\left(\mathbf{h}_{\text{p}}; \phi\right))^T g\left(\mathbf{h}_{\text{n}}; \phi\right) \\ - 2(1 + \tan^2 \alpha) g\left(\mathbf{h}_{\text{adv}}; \phi\right)^T g\left(\mathbf{h}_{\text{p}}; \phi\right),\end{aligned} \tag{3.4}$$

where $\mathbf{h}_{\text{adv}}^{\text{i}}$, $\mathbf{h}_{\text{p}}^{\text{i}}$, $\mathbf{h}_{\text{n}}^{\text{i}}$ denote anchor, positive and negative learned embeddings, respectively. y represents class labels, α defines the target angle between sample points.

Apart from the selection of proper metric learning loss functions, sampling [60] the input points should also be carefully implemented. One of the essentials is the class-balanced sampling [59]. Some datasets may have an imbalance in the number of images per class. Deep learning models tend to overfit the over-represented classes during training. Therefore, minority classes might not be prioritized during gradient updates in latent space, and the evaluation performance of these minority classes degrades. As a result, the importance of class-balanced sampling increases and generalization performances can be enhanced using this technique.

4. EXPERIMENTS

In this section, datasets used in the experiments, implementation details, evaluation metrics and experimental results are presented. Experiments are conducted on TRUBA akya-cuda, barbun-cuda servers. Akya-cuda, barbun-cuda servers consist of Nvidia Tesla V100 GPU and Nvidia P100 GPU cards, respectively.

4.1. Datasets

CUB200-2011 [22], CARS196 [23] and Stanford Online Products (SOP) [3] are the benchmark datasets widely used in deep metric learning. Since all of these datasets use entirely different classes between training and test sets, they force the models to learn a generalizable metric of similarity rather than only memorizing particular items.

4.1.1. CUB200-2011

Caltech-UCSD Birds-200-2011 (CUB-200-2011) [22] can be defined as a modified version of the CUB-200 dataset. It is mainly used for fine-grained categorization tasks in computer vision. CUB-200-2011 consists of 200 bird classes with 11,788 images in total. While training data contains the first 100 classes with 5,864 images, the test set has the other 100 classes with 5,924 images. Rich annotations like bounding boxes, part locations of birds, and binary attribute labels are also provided in order to pave the way for the analysis of various computer vision tasks. Besides, since the distinction between categories may be very subtle in fine-grained image classification tasks, this data set can be challenging.

4.1.2. CARS196

The dataset [23] includes 16,185 car images from 196 different classes. High-quality images are captured from several angles and locations under good lighting

effects. The variety in images guarantees that models have better training gradient updates to learn more distinctive features. While the train set has the first 98 types of cars with 8,144 images, the test set includes the last 98 classes with 8,041 images. Although this data set is primarily used for classification and retrieval of automobiles according to their models, bounding boxes are also placed for further investigations in object detection domain.

4.1.3. Stanford Online Products (SOP)

Images of vast range of products from bicycle to kettle are collected from online shopping websites. In total, SOP dataset has 22,634 classes with 120,053 images. It includes 59,551 images from 11,318 classes for training and 60,502 images from the remaining 11,157 classes for testing. This dataset has different characteristics from the previous ones. Even though it is a larger dataset with respect to CUB200-2011 and CARS196, the number of images per class is very small due to the existence of a huge amount of sub-categories. Therefore, the SOP dataset constitutes a challenging data distribution [3].

4.2. Evaluation Metrics

Evaluating learned embeddings is crucial to determining a model's performance in deep metric learning. In deep metric learning research, customized metrics that are more in line with the goals of metric learning are preferred rather than merely depending on standard classification accuracy. Normalized Mutual Information (NMI) and Recall@K are well-known and widely used metrics.

4.2.1. Normalized Mutual Information

Mutual Information (MI) determines how closely the clustering outcomes match the underlying data structure. The number of clusters may affect the performance of mutual information scores without a normalization. Therefore, it is beneficial to use

Normalized Mutual Information (NMI) [61] that ranges from 0 to 1. While 1 denotes the highest clustering score, 0 represents the independence between clusters. During the evaluation process, all samples are assigned to a cluster that has the nearest center point using K-Means. Set of clusters $\boldsymbol{\Omega} = \{w_1, ..., w_k\}$ and ground truth classes $\boldsymbol{\Upsilon} = \{v_1, ..., v_k\}$ are determined, where w_i denotes the samples that belong to the ith cluster, and v_j represents the set of samples with a true label of j. The normalized mutual information is computed as

$$NMI(\boldsymbol{\Omega}, \boldsymbol{\Upsilon}) = \frac{2I(\boldsymbol{\Omega}, \boldsymbol{\Upsilon})}{H(\boldsymbol{\Omega}) + H(\boldsymbol{\Upsilon})}, \tag{4.1}$$

where $I(\boldsymbol{\Omega}, \boldsymbol{\Upsilon})$ represents the information shared between predicted cluster assignments and their ground truth classes, $H(\boldsymbol{\Omega})$ and $H(\boldsymbol{\Upsilon})$ are the entropies of clusters and ground truth classes, respectively.

4.2.2. Recall@K

Recall@K [62] is a commonly used metric to measure retrieval quality. It is defined as the ratio of the number of recommended items within the top K relevant results and the total amount of relevant items. If top K samples include X relevant items among all Y relevant items, it can be formulated as $Recall@K = X \ / \ Y$.

One may learn how the quality of retrieval changes as more results are taken into account by investigating Recall@K for various values of K. Recall@1 concentrates on the top result, whereas Recall@10 offers an expanded overview of the top 10 results. Therefore, different K values are applied during experiments depending on the size of the dataset. While Recall@1, 2, 4, 8 are preferred for CUB200-2011 and CARS196, Recall@1, 10, 100, 1000 values are used for the more extensive dataset SOP. After proper K values are assigned for experiments, cosine similarities are computed to determine these samples.

4.3. Implementation Details

This section explains architecture types, data augmentation techniques, training parameters of embedding, and robust metric learning modules in detail. Moreover, sampling strategies for each dataset are clarified in the implementation details of the robust metric learning module. All models are implemented with the PyTorch library.

4.3.1. Embedding Module and PGD Attack Generation

For the embedding module, pre-trained ResNet-18 and ResNet-50 models were fine-tuned separately for each benchmark dataset to obtain improved natural metric learning performances and more meaningful representations in the latent space [58]. Before training, pre-processing and data augmentations were applied to datasets. Since each dataset has different characteristics, their pre-processing and data augmentations were also specialized. The dataset-specific pre-processing steps for training are outlined below.

- CUB200-2011: Resizing to 256x256, color jitter (0.25, 0.25, 0.25, 0), randomized crop 224x224 with scaling (0.16, 1) and ratio (4/3, 3/4).
- CARS196: Resizing to 224x224, color jitter (0.3, 0.3, 0.3, 0.1), randomized crop 224x224 with scaling (0.16, 1), and ratio (1, 1).
- SOP: Resizing to 256x256, randomized crop 224x224 with scaling (0.16, 1) and ratio (4/3, 3/4).

The same test data transformations, resizing to 256x256 and center crop to 224x224, are applied to all three datasets. All pre-trained models such as ResNet-18 and ResNet-50 in PyTorch expects at least 224x224 pixels.

After the pre-processing operations, the CUB200-2011 dataset is trained for 30 epochs with a learning rate of 0.02. While 100 epochs and a learning rate of 0.05 are required for the best performance of CARS196, the SOP dataset is trained with 100

epochs and a learning rate of 0.003. The model weights are saved for both ResNet-18 and ResNet-50 architectures after fine-tuning. It is essential to emphasize that these complex DNNs in the embedding module are only naturally trained, and the embedding model's training is not complicated with white-box attack generations at each iteration.

After the fine-tuning step, adversarial attacks are generated by a 5-step and 7-step PGD attack, denoted by PGD-5 and PGD-7, for each model and dataset. Epsilon and gamma values of PGD attacks are chosen in agreement with the literature to be used in our experiments. The attack budget ϵ is set to 0.01, and the step size γ is set to 0.004 for PGD-5 for all datasets [49]. Moreover, γ is also experimented with 0.003 for SOP dataset [63]. For PGD-7, ϵ is set to 0.0314 and step size is used as 0.007 for CARS196 and CUB200-2011 datasets [63]. Since our proposed method has focused on improving adversarial robustness in clustering and image retrieval tasks, we have attacked triplet loss with a margin value 1.0 during the PGD attacks. The adversarial images are generated one time and saved in the memory. Unlike adversarial training, perturbed images are not generated repeatedly during the training. However, they are incorporated in the robust metric learning in a black-box manner to reduce the robust training time. Natural, $\mathbf{x}$, and adversarial inputs, $\mathbf{x}^{'}$, are forwarded through the fine-tuned model $\mathbf{h} = f(\mathbf{x}; \theta) \in \mathbb{R}^d$, where resulting embedding size is $d = 2048$ for ResNet-50 and $d = 512$ for ResNet-18 models. Then, these embeddings are normalized before they are fed to a robust metric learning module.

4.3.2. Robust Metric Learning Module

The robust metric learning module maps embeddings into different dimensional latent spaces to increase semantic similarities between the natural and adversarial samples of the same class while decreasing the similarity between the separate classes.

As mentioned in Chapter 3, some datasets might not be equally distributed for each class. While class j may include several numbers of images, other class k may contain one or two input data. This imbalance in datasets causes a decline in model

generalization performances. Therefore, proper mini-batch class sampling for each dataset is determined in the following experiments. To compare the results of different sampling strategies, 2048 dimensional ResNet-50 embeddings, $\mathbf{h} = f(\mathbf{x}; \theta) \in \mathbb{R}^{2048}$ are used to train the robust metric learning module with the triplet loss function. In the Table 4.1, Table 4.2, and Table 4.3, class # and images/class # denote number of classes and number of images per class in a mini-batch, respectively. It can be realized that each class has the same amount of data in the mini-batch, and gradient updates are distributed equally thanks to these sampling strategies.

Table 4.1. Performance comparison of different sampling strategies for CUB200-2011 dataset.

CUB200-2011								
Strategy Name	Class#	Images/Class#	Batch Size	NMI	R@1	R@2	R@4	R@8
Natural Samples								
S1	3	25	75	67.8	86.3	88.9	91.7	94.5
S2	5	15	75	68.0	86.6	89.0	92.0	94.5
S3	15	5	75	68.9	87.0	**89.4**	92.2	94.7
S4	100	5	500	**70.2**	**87.1**	89.2	**92.5**	**95.0**
PGD-5 ($\epsilon = 0.01$)								
S1	3	25	75	27.0	21.3	27.7	36.1	46.4
S2	5	15	75	**28.1**	22.2	28.7	37.9	48.6
S3	15	5	75	27.8	**22.6**	**29.7**	**38.9**	**49.9**
S4	100	5	500	25.4	17.5	23.8	32.2	43.2

Although maximum NMI and recall@K values for natural samples of CUB200-2011 are observed in the strategy S4 that includes 100 distinct classes with batch size 500 in the training dataset, the lowest adversarial performances are also obtained in this setting as given in Table 4.1. On the other hand, in the latent space of adversarial robust metric learning, strategy S3 that consists of 15 distinct classes with batch size 75 outperforms the other strategies in terms of all recall@K values, while the second-best

natural performances are also achieved. Since the main purpose is to find a strategy maximizing both natural and adversarial performances simultaneously, S3 is selected as the champion for the CUB200-2011 dataset.

In Table 4.2, the CARS196 dataset is evaluated with different sampling strategies. Similar to CUB200-2011, natural performances are boosted by strategy S4 when all 98 distinct classes are included in a mini-batch with batch size 490. However, in contrast to CUB200-2011, the maximum adversarial performance scores except the NMI can be observed again in this sampling strategy. Therefore, batch size 490 with 98 distinct classes is applied to all further investigations for the CARS196 dataset.

Table 4.2. Performance comparison of different sampling strategies for CARS196 dataset.

CARS196								
Strategy Name	Class#	Images/Class#	Batch Size	NMI	R@1	R@2	R@4	R@8
Natural Samples								
S1	3	25	75	67.8	92.8	94.5	96.2	97.7
S2	5	15	75	68.8	93.0	94.7	96.4	97.6
S3	15	5	75	70.9	93.5	94.9	96.8	98.0
S4	98	5	490	**72.5**	**94.0**	**95.4**	**97.1**	**98.1**
PGD-5 ($\epsilon = 0.01$)								
S1	3	25	75	26.7	30.4	38.8	49.1	60.7
S2	5	15	75	27.2	32.4	40.6	51.7	63.4
S3	15	5	75	**28.1**	37.6	46.9	57.5	68.4
S4	98	5	490	27.4	**39.5**	**48.1**	**58.1**	**69.0**

Fewer experiments have been conducted on the SOP dataset compared to other datasets. The first reason for this is the long training times due to the large scale of the dataset. Secondly, each class has a very low number of samples in the SOP dataset. The scarcity of images per class also reduces the number of experimental alternatives.

As seen in Table 4.3, sampling strategy S2 that consists of 40 classes with batch size 80 is superior to the others by considering both natural and adversarial results.

Table 4.3. Performance comparison of different sampling strategies for SOP dataset.

SOP								
Str. Name	Class#	Images/Class#	Batch Size	NMI	R@1	R@10	R@100	R@1000
Natural Samples								
S1	15	5	75	**93.4**	89.4	93.1	96.1	**98.4**
S2	40	2	80	**93.4**	**89.6**	**93.2**	**96.3**	**98.4**
PGD-5 ($\epsilon = 0.01$)								
S1	15	5	75	87.9	74.9	81.6	87.4	93.1
S2	40	2	80	**88.1**	**75.2**	**81.9**	**87.7**	**93.5**

After sampling strategies are determined, natural and adversarial embeddings are fed through the robust metric learning module. The resulting dimension size in the latent space may vary depending on the architecture selection. For instance, original ResNet-50 embedding size, $d = 2048$, can be decreased to $d^{'} = 512$ or can be increased to $d^{'} = 4096$ during the experiments. On the other hand, the original ResNet-18 embedding size, $d = 512$, can be mapped to several output dimensions such as $d^{'} \in [128, 256, 1024]$ with the aim of contributing model robustness. Some dimensions of ResNet representations might not be robust against adversarial attacks [64]. Therefore, our study tries to achieve more robust representations in the latent space by minimizing the dimension size. On the other hand, significant robustness features, which are not captured by the original ResNet embeddings, are investigated by mapping to the larger space.

During the training of robust metric learning, the loss functions are optimized by stochastic gradient descent (SGD) with a learning rate of 0.01 for all datasets. When the resulting dimension $d^{'}$ for the CUB200-2011 dataset is set to 512, all loss functions are trained for ten epochs. The margin value for triplet loss is 0.05, whereas alpha

for angular loss is set to 20. However, the triplet loss function with a final dimension size $d^{'} = 4096$ is trained for 20 epochs to reach the maximum robustness. Considering the CARS196 dataset with the final dimension size $d^{'} = 512$, triplet, contrastive, and angular loss functions are updated for 30, 30, and 20 epochs, respectively. If the resulting dimension is set to $d^{'} = 4096$, 50 epochs are needed for training. The alpha of angular loss is assigned to 30. For the largest scale dataset, SOP, 5, 10, and 10 epochs are utilized for triplet, contrastive, and angular losses, respectively. Like the CARS196 dataset, the alpha value of angular loss is set to 30.

When the ResNet-18 architecture is utilized in the embedding module, some RML training parameters mentioned above are slightly updated. CUB200-2011 dataset embeddings are mapped to lower dimensional space using triplet loss for 20 epochs, whereas 30 epochs are utilized for increasing input dimension from $d = 512$ to output dimension size $d' = 1024$. The alpha of angular loss is set to 20. Unlike the other training setups, the CARS196 dataset gets better updates when the learning rate of SGD is 0.1. All model updates are completed using 10 epochs except the triplet loss function with final dimension size $d = 1024$, which is trained by 20 epochs. The best-performing alpha for this dataset is selected as 20. Finally, the RML module learns for 10 epochs during the training of all SOP dataset models, and angular loss is executed by using an alpha value of 20.

4.4. Baseline Studies

Panum et al. [49] fine-tunes the complex architectures with adversarial training to achieve more robust representations in the latent space. This work attacks to contrastive and triplet loss functions instead of cross-entropy loss for producing PGD adversarial attack. They have improved the adversarial robustness of models in metric learning for retrieval tasks and evaluated the model performances against the PGD-5 for all datasets. The attack budget ϵ and step size γ are set to 0.01 and 0.004, respevtively.

Wu and Huang [63] have benefited from alignment (A) and uniformity loss (U) separately instead of triplet loss (T) in their ADML adversarial training framework. While alignment loss controls the intra-class distances of positive pairs, inter-class distances of negative pairs are considered in uniformity loss. They have compared the robustness performance against a 7-step PGD attack with an ϵ of 0.0314 and step size γ of 0.007 for CUB200-2011 and CARS196 datasets. On the other hand, a 5-step PGD attack with an ϵ of 0.01 and step size γ of 0.003 is generated for the SOP dataset.

4.5. Quantitative Results

CUB200-2011, CARS196, and SOP are widely used benchmark datasets in metric learning domain. Therefore, we have conducted our experiments using these benchmark datasets. In all tables below, dimension ($d^{'}$) represents the output dimension size of related model. Each dataset is experimented with ResNet-50 and ResNet-18 embeddings, seperately. To be clearer about all models in the first column of the tables, they are listed with their explanations:

- ResNet: The standard pre-trained model [12] on ImageNet.
- FT ResNet: Fine-tuned (FT) pre-trained model [58] on our specific datasets with the original embedding size.
- FT ResNet w/o RML: Performance of fine-tuned pre-trained model on our specific datasets, when they are mapped to new latent space without a RML module.
- EARDML$_{\text{Contrastive}}$: AT-based adversarial deep metric learning model trained by contrastive loss [49].
- EARDML$_{\text{Triplet}}$: AT-based adversarial deep metric learning model trained by triplet loss [49].
- RML$_{\text{Contrastive}}$: The proposed RML performance using the contrastive loss.
- RML$_{\text{Angular}}$: The proposed RML performance using the angular loss.
- RML$_{\text{Triplet}}$: The proposed RML performance using the triplet loss.
- ADML + T: AT-based adversarial deep metric learning model trained by triplet loss [63].

- ADML + U: AT-based adversarial deep metric learning model trained by uniformity loss [63].
- ADML + A: AT-based adversarial deep metric learning model trained by alignment loss [63].

Table 4.4. Natural and adversarial performances of robust metric learning module trained with CUB200-2011 ResNet-50 embeddings.

CUB200-2011						
Model	Dimension (d')	NMI	R@1	R@2	R@4	R@8
Natural Samples						
ResNet-50 [12]	2048	57.8	47.5	61.6	73.0	83.9
FT ResNet-50 [58]	2048	**71.4**	**87.7**	**90.2**	**93.2**	**95.3**
FT ResNet-50 w/o RML	512	69.8	86.7	89.2	92.0	94.9
$\text{EARDML}_{\text{Contrastive}}$ [49]	128	-	58.2	-	-	-
$\text{EARDML}_{\text{Triplet}}$ [49]	128	-	53.4	-	-	-
$\text{RML}_{\text{Contrastive}}$	512	64.5	85.1	87.7	90.6	93.6
$\text{RML}_{\text{Angular}}$	512	70.1	87.0	89.6	92.5	94.9
$\text{RML}_{\text{Triplet}}$	512	68.9	87.0	89.4	92.2	94.7
$\text{RML}_{\text{Triplet}}$	4096	70.5	86.9	89.2	92.4	95.0
PGD-5 ($\epsilon = 0.01$)						
ResNet-50 [12]	2048	27.5	12.5	20.1	29.6	42.3
FT ResNet-50 [58]	2048	23.8	17.1	23.1	31.4	42.3
FT ResNet-50 w/o RML	512	23.3	15.5	21.3	28.8	39.4
$\text{EARDML}_{\text{Contrastive}}$ [49]	128	-	20.3	-	-	-
$\text{EARDML}_{\text{Triplet}}$ [49]	128	-	16.9	-	-	-
$\text{RML}_{\text{Contrastive}}$	512	**27.9**	19.7	26.4	34.2	45.3
$\text{RML}_{\text{Angular}}$	512	24.0	17.5	23.4	31.0	42.1
$\text{RML}_{\text{Triplet}}$	512	27.8	22.6	29.7	38.9	49.9
$\text{RML}_{\text{Triplet}}$	4096	27.7	**24.6**	**32.8**	**41.4**	**52.3**

Firstly, ResNet-50 embeddings are utilized in RML module as demonstrated in Table 4.4, Table 4.5, and Table 4.6. All experiments are conducted under natural and adversarial attack settings. While the upper half of the tables represents the model performances against natural samples, the below part displays the adversarial robustness of models under the PGD attack.

When all of these tables are analyzed, it can be observed that although the ResNet models have satisfactory results with natural embeddings, their performance degrades significantly against adversarial perturbations. The performance degradations may differ depending on the dataset characteristics. The CUB200-2011 classes include cars from the same type, whereas the colors and background features may be very different between the instances within the same class. Besides, bird images in the CUB200-2011 are captured at varying background environments. Therefore, it can be said that these two datasets have high intra-class variability. However, the SOP dataset consists of a lower number of images per class, and these images are captured by taking photos of the same product from different perspectives in the same background. This setup decreases the intra-class variability. For these reasons, the success rate of the adversarial attack is higher for the CUB200-2011 and CARS196 datasets and lower for the SOP dataset.

In Table 4.4, it can be seen that the FT ResNet-50 model outperforms the standard pre-trained ResNet-50 in natural and adversarial settings. Therefore, our proposed RML module benefits from the fine-tuned embeddings for all experiments. FT ResNet-50 metrics tend to decline when they are forwarded through the neural nets without any RML optimization. For instance, recall@1 decreases from 17.1% to 15.5%, and recall@2 declines from 23.1% to 21.3%. In contrast, the proposed RML module with all different loss functions improves the semantic relations between natural and adversarial samples while keeping the natural performances close. Despite enhancements provided by angular and contrastive loss, triplet loss with RML can be determined as the best-performing model. To illustrate, $\text{RML}_{\text{Triplet}}$ with $d^{'} = 512$ increases the adversarial recall@1 from 17.1 to 22.6, while preserving the natural recall@1 around 86.94%.

Table 4.5. Natural and adversarial performances of robust metric learning module trained with CARS196 ResNet-50 embeddings.

CARS196						
Model	Dimension (d')	NMI	R@1	R@2	R@4	R@8
Natural Samples						
ResNet-50 [12]	2048	42.0	44.2	56.7	68.1	78.6
FT ResNet-50 [58]	2048	**74.2**	**96.1**	**96.2**	**97.6**	**98.6**
FT ResNet-50 w/o RML	512	73.4	94.4	95.6	97.1	98.3
EARDML$_{\text{Contrastive}}$ [49]	128	-	72.1	-	-	-
EARDML$_{\text{Triplet}}$ [49]	128	-	71.9	-	-	-
RML$_{\text{Contrastive}}$	512	68.8	93.3	95.0	96.7	98.0
RML$_{\text{Angular}}$	512	56.5	85.4	87.4	90.4	93.6
RML$_{\text{Triplet}}$	512	72.5	94.0	95.4	97.1	98.1
RML$_{\text{Triplet}}$	4096	73.5	94.6	95.9	97.4	98.5
PGD-5 ($\epsilon = 0.01$)						
ResNet-50 [12]	2048	18.5	10.8	16.2	23.8	33.4
FT ResNet-50 [58]	2048	19.4	27.5	35.2	44.8	56.1
FT ResNet-50 w/o RML	512	18.7	24.0	31.3	40.5	51.6
EARDML$_{\text{Contrastive}}$ [49]	128	-	35.7	-	-	-
EARDML$_{\text{Triplet}}$ [49]	128	-	36.2	-	-	-
RML$_{\text{Contrastive}}$	512	27.5	38.1	47.4	57.2	67.7
RML$_{\text{Angular}}$	512	**27.7**	15.8	22.8	32.3	44.3
RML$_{\text{Triplet}}$	512	27.4	39.5	48.1	58.1	69.0
RML$_{\text{Triplet}}$	4096	26.3	**40.6**	**49.6**	**59.5**	**70.3**

When proposed RML module performances are compared with the EARDML models as shown in Table 4.4, it is clear that our best-performing RML$_{\text{Triplet}}$ is more robust than the EARDML$_{\text{Contrastive}}$ and EARDML$_{\text{Triplet}}$ models under the PGD attack. RML achieves these better performances without adversarial training. It does not attack the complex architectures for each update; therefore, training times and

complexity of training are reduced. Moreover, it is crucial to emphasize that since EARDML models benefit from adversarial training, they are prone to overfit adversarial perturbations in the training set. In parallel to this idea, EARDML natural recall@1 performances are at very low levels, like 58.2% and 53.4%, while the RML natural performances are around 87.0%.

In Table 4.5, it is obvious that the CUB200-2011 dataset is also affected significantly by the PGD attack. The natural NMI of FT ResNet-50 embeddings is decreasing from 74.2% to 19.4% under the attack. Table 4.5 displays that $\text{RML}_{\text{Contrastive}}$ and $\text{RML}_{\text{Triplet}}$ methods contribute to the adversarial robustness of models in terms of all evaluation metrics, similar to the CUB200-2011 dataset. In contrast, $\text{RML}_{\text{Angular}}$ is not able to increase recall values of FT ResNet-50 embeddings in an adversarial setting. While $\text{RML}_{\text{Angular}}$ can be expressed as the common least performing method for CUB200-2011 and CARS196 datasets, $\text{RML}_{\text{Triplet}}$ with $d' = 4096$ can be explored as the most defensive method for these two datasets.

EARDML natural recall@1 performance also significantly degrades from 71.9% to 36.2% when the adversarial attack is applied, as illustrated in Table 4.5. This vulnerability of EARDML can be improved by $\text{RML}_{\text{Contrastive}}$ and $\text{RML}_{\text{Triplet}}$ modules. They have 38.1% and 40.6% recall@1 scores against the PGD attack, respectively. Apart from the adversarial robustness, $\text{RML}_{\text{Triplet}}$ has nearly 20% higher natural performance score than $\text{EARDML}_{\text{Triplet}}$.

The last experiments with ResNet-50 embeddings are conducted for the SOP dataset as demonstrated in Table 4.6. Unlike the CUB200-2011 and CARS196 datasets, adversarial attacks have less capability to reduce the recall performances of the SOP dataset. To illustrate, the PGD attack decreases recall@1 from 91.4% to 69.6% for the FT ResNet-50 model implementation of the SOP, while it causes a decline from 96.1% to 27.5% for CARS196. As explained before, this situation arises from the different intra-class variabilities of different datasets.

Table 4.6. Natural and adversarial performances of robust metric learning module trained with SOP ResNet-50 embeddings.

SOP						
Model	Dimension ($d^{'}$)	NMI	R@1	R@10	R@100	R@1000
Natural Samples						
ResNet-50 [12]	2048	86.2	54.3	70.6	83.7	94.5
FT ResNet-50 [58]	2048	**94.2**	**91.4**	**94.8**	**97.3**	**99.0**
FT ResNet-50 w/o RML	512	93.8	90.7	94.2	96.8	98.5
$\text{EARDML}_{\text{Contrastive}}$ [49]	128	-	66.7	-	-	-
$\text{EARDML}_{\text{Triplet}}$ [49]	128	-	64.0	-	-	-
$\text{RML}_{\text{Contrastive}}$	512	93.2	88.6	92.4	95.6	98.0
$\text{RML}_{\text{Angular}}$	512	91.3	82.8	87.5	92.4	96.7
$\text{RML}_{\text{Triplet}}$	512	93.4	89.6	93.2	96.3	98.4
PGD-5 ($\epsilon = 0.01$)						
ResNet-50 [12]	2048	80.8	21.2	35.3	54.8	78.3
FT ResNet-50 [58]	2048	86.4	69.6	77.4	84.4	90.9
FT ResNet-50 w/o RML	512	86.0	66.8	75.1	82.4	89.4
$\text{EARDML}_{\text{Contrastive}}$ [49]	128	-	53.6	-	-	-
$\text{EARDML}_{\text{Triplet}}$ [49]	128	-	39.3	-	-	-
$\text{RML}_{\text{Contrastive}}$	512	**89.1**	**77.7**	**83.1**	**88.3**	93.4
$\text{RML}_{\text{Angular}}$	512	87.5	67.7	77.7	85.6	93.0
$\text{RML}_{\text{Triplet}}$	512	88.1	75.2	81.9	87.7	**93.5**

The robustness of FT ResNet-50 embeddings can be boosted from 69.6% to 77.7% by the proposed $\text{RML}_{\text{Contrastive}}$ module regarding the recall@1. $\text{RML}_{\text{Triplet}}$ also improves the robustness to PGD perturbations up to 75.2%. Similar to previous insights from different datasets, RML modules perform better than EARDML methods in natural and adversarial settings while simultaneously decreasing the training time and complexity.

Table 4.7. Natural and adversarial performances of robust metric learning module trained with CUB200-2011 ResNet-18 embeddings.

CUB200-2011						
Model	Dimension (d')	NMI	R@1	R@2	R@4	R@8
Natural Samples						
FT ResNet-18 [58]	512	**66.3**	**85.1**	**87.8**	**91.2**	**94.3**
FT ResNet-18 w/o RML	128	59.8	81.2	83.9	87.9	91.6
FT ResNet-18 w/o RML	256	61.8	82.4	85.2	89.0	92.6
FT ResNet-18 w/o RML	1024	63.7	83.4	86.0	89.7	93.3
$\text{RML}_{\text{Contrastive}}$	1024	61.3	82.3	84.7	88.6	92.4
$\text{RML}_{\text{Angular}}$	1024	58.9	81.0	83.7	87.7	91.4
$\text{RML}_{\text{Triplet}}$	128	58.8	80.1	82.8	86.4	90.7
$\text{RML}_{\text{Triplet}}$	256	59.7	81.4	84.4	87.9	91.8
$\text{RML}_{\text{Triplet}}$	1024	61.0	81.9	84.5	88.6	92.6
PGD-5 ($\epsilon = 0.01$)						
FT ResNet-18 [58]	512	22.1	15.5	21.4	29.4	39.5
FT ResNet-18 w/o RML	128	21.7	9.3	13.5	20.6	29.5
FT ResNet-18 w/o RML	256	21.5	10.8	15.5	22.5	31.7
FT ResNet-18 w/o RML	1024	21.8	12.4	17.8	25.3	34.6
$\text{RML}_{\text{Contrastive}}$	1024	**26.1**	**21.6**	**28.9**	**38.0**	**48.4**
$\text{RML}_{\text{Angular}}$	1024	22.6	12.9	17.9	24.8	34.3
$\text{RML}_{\text{Triplet}}$	128	24.8	15.5	21.6	29.6	40.6
$\text{RML}_{\text{Triplet}}$	256	25.3	18.3	24.9	33.3	44.2
$\text{RML}_{\text{Triplet}}$	1024	25.6	20.3	27.0	35.5	46.6

After the proposed RML module has proved its success with ResNet-50 embeddings, we tested the performance of RML with embeddings learned from a simpler architecture. For this aim, ResNet-18 is utilized for the experiments conducted in Table 4.7, Table 4.8, and Table 4.9. ResNet-50 consists of a deeper architecture to learn discriminative features better for complex tasks. On the other hand, ResNet-18 sheds

light on simpler tasks due to its narrow architecture. However, it can be trained with less computational resources.

Initially, in Table 4.7, the original FT ResNet-18 embedding size $d^{'} = 512$ is mapped to 128, 256, and 1024 without implementing any optimization for CUB200-2011 dataset. It can be seen that a minor decrease in the performance is observed when the final dimension size is set to 1024. Therefore, all loss functions are tested under the scenario of mapping the original embeddings into $d^{'} = 1024$. Moreover, since triplet loss has generally outperformed the other loss functions in the previous experiments, $\text{RML}_{\text{Triplet}}$ module is also tested with varying final output dimension sizes. Table 4.7 displays that $\text{RML}_{\text{Contrastive}}$ natural performances have a slight decrease from original FT ResNet-18 performances, and it boosts the adversarial robustness under the PGD attack. For instance, $\text{RML}_{\text{Contrastive}}$ increases the recall@1 value of original FT ResNet-18 from 15.5% to 21.6% while preserving natural recall@1 at 82.3%. In contrast to the success of contrastive and triplet losses, angular loss is not able to provide robustness to original embeddings, similar to previous results seen in the ResNet-50 experiments.

FT ResNet-18 embeddings for the CARS196 dataset lose the least amount of natural and adversarial knowledge, when their final output dimension size $d^{'}$ is mapped from original dimension size 512 to 1024 as displayed in Table 4.8. After exploring this insight, all loss functions are evaluated with this dimension size, similar to the CUB200-2011 experimental setup. It can be analyzed that $\text{RML}_{\text{Contrastive}}$ and $\text{RML}_{\text{Triplet}}$ show nearly equal performances under the natural and adversarial settings. In general, while $\text{RML}_{\text{Contrastive}}$ demonstrates slightly better performance in recall metrics with natural inputs, $\text{RML}_{\text{Triplet}}$ has a minor improvement over the $\text{RML}_{\text{Contrastive}}$ module during the adversarial attack setting. Therefore, there can be a trade-off between natural and adversarial performances depending on the purpose of the users.

Like in the previous SOP experiments conducted with ResNet-50 embeddings, PGD attack can decrease the performance of FT ResNet-18 embeddings from 89.9% to 53.3% as shown in the Table 4.9, while this decline amount is more significant with

the other datasets. Although the attack success rate is lower on the SOP dataset, the proposed RML module also improves these performances. For instance, the adversarial robustness of FT ResNet-18 embeddings can be boosted using $\text{RML}_{\text{Triplet}}$ module, which increases recall@1 from 53.3% to 58.7%, and recall@1000 from 86.7% to 90.2%.

Table 4.8. Natural and adversarial performances of robust metric learning module trained with CARS196 ResNet-18 embeddings.

CARS196						
Model	Dimension (d')	NMI	R@1	R@2	R@4	R@8
Natural Samples						
FT ResNet-18 [58]	512	**67.2**	**92.4**	**94.0**	**96.2**	**97.8**
FT ResNet-18 w/o RML	128	58.5	86.9	89.3	92.3	95.1
FT ResNet-18 w/o RML	256	61.0	89.1	91.1	94.0	96.1
FT ResNet-18 w/o RML	1024	63.2	90.3	92.4	95.0	96.9
$\text{RML}_{\text{Contrastive}}$	1024	61.7	89.8	92.0	94.4	96.5
$\text{RML}_{\text{Angular}}$	1024	56.5	85.7	88.0	91.2	93.8
$\text{RML}_{\text{Triplet}}$	128	60.2	87.0	89.2	92.4	95.0
$\text{RML}_{\text{Triplet}}$	256	61.9	88.7	90.9	93.7	96.0
$\text{RML}_{\text{Triplet}}$	1024	61.9	89.5	91.5	94.1	96.1
PGD-5 ($\epsilon = 0.01$)						
FT ResNet-18 [58]	512	17.7	12.9	19.5	28.6	40.9
FT ResNet-18 w/o RML	128	17.0	7.7	12.4	18.8	28.3
FT ResNet-18 w/o RML	256	17.4	9.8	15.2	22.2	32.4
FT ResNet-18 w/o RML	1024	17.3	10.6	16.2	23.5	34.2
$\text{RML}_{\text{Contrastive}}$	1024	**22.1**	**16.5**	23.8	33.3	45.5
$\text{RML}_{\text{Angular}}$	1024	17.9	8.4	12.9	19.9	29.2
$\text{RML}_{\text{Triplet}}$	128	22.0	11.9	17.8	26.4	37.5
$\text{RML}_{\text{Triplet}}$	256	21.6	13.9	20.5	29.3	40.4
$\text{RML}_{\text{Triplet}}$	1024	**22.1**	16.1	**23.9**	**33.5**	**45.6**

Table 4.9. Natural and adversarial performances of robust metric learning module trained with SOP ResNet-18 embeddings.

SOP						
Model	Dimension ($d^{'}$)	NMI	R@1	R@10	R@100	R@1000
Natural Samples						
FT ResNet-18 [58]	512	**93.7**	**89.9**	**93.7**	**97.0**	**98.8**
FT ResNet-18 w/o RML	128	92.7	87.3	91.4	94.9	97.4
FT ResNet-18 w/o RML	256	93.0	88.2	92.3	95.6	97.9
$\text{RML}_{\text{Contrastive}}$	1024	93.1	88.4	92.4	95.9	98.2
$\text{RML}_{\text{Angular}}$	1024	92.3	86.6	90.7	94.4	97.2
$\text{RML}_{\text{Triplet}}$	128	92.7	87.1	91.3	95.1	98.0
$\text{RML}_{\text{Triplet}}$	256	92.8	87.8	91.9	95.5	98.1
$\text{RML}_{\text{Triplet}}$	1024	93.0	88.2	92.3	95.8	98.2
PGD-5 ($\epsilon = 0.01$)						
FT ResNet-18 [58]	512	84.4	53.3	64.9	76.1	86.7
FT ResNet-18 w/o RML	128	83.3	43.0	55.0	67.5	80.9
FT ResNet-18 w/o RML	256	83.7	46.5	58.8	70.7	83.1
$\text{RML}_{\text{Contrastive}}$	1024	85.3	58.1	69.8	79.8	88.6
$\text{RML}_{\text{Angular}}$	1024	82.7	46.2	57.8	68.7	80.8
$\text{RML}_{\text{Triplet}}$	128	85.2	54.2	67.8	79.6	89.6
$\text{RML}_{\text{Triplet}}$	256	85.4	56.1	69.3	80.4	89.9
$\text{RML}_{\text{Triplet}}$	1024	**85.7**	**58.7**	**70.9**	**81.1**	**90.2**

In the experiments above, we have compared the adversarial robustness of several approaches against the 5-step PGD attack with an ϵ of 0.01 and step size γ of 0.004. In addition, it is significant to show the robustness of these approaches under different adversarial attack settings. If the model provides promising performances under different types of attacks, the model's robustness might be considered generalizable under varying scenarios. For this purpose, a 7-step PGD attack with an ϵ of 0.0314 and step size γ of 0.007 is used for CUB200-2011 and CARS196 datasets. On the other hand,

a 5-step PGD attack with an ϵ of 0.01 and step size γ of 0.003 is generated for the SOP dataset with the aim of comparing the results of the proposed method with the adversarial deep metric learning (ADML) [63] approach. We should also note that generalizability to various kinds of adversarial attacks is extremely challenging to attain since it is possible to propose new attacks. For this reason, our goal is to improve the generalizability for commonly used attacks in the robust metric learning domain rather than claiming to be generalizable to any kind of adversarial attack.

In Table 4.10, adversarial samples generated by attacking the alignment loss are evaluated using various ADML and the proposed RML frameworks. Table 4.10 demonstrates that our approach outperforms the ADML approaches regarding recall@1 and NMI values for all datasets except the NMI result of CUB200-2011.

Table 4.10. Adversarial robustness of different approaches including the proposed RML against adversarial samples synthesized by attacking alignment loss.

	CUB200-2011		CARS196		SOP	
Models	R@1	NMI	R@1	NMI	R@1	NMI
ADML + T [63]	11.58	25.3	25.4	21.2	10.7	80.2
ADML + A [63]	17.4	**29.2**	40.0	26.1	14.0	80.4
ADML + U [63]	15.1	27.9	33.1	24.5	11.3	80.3
RML	**24.0**	27.4	**50.9**	**35.3**	**73.0**	**88.0**

After the robustness performance of our method is evaluated under varying architectures and attack settings, training time investigations are conducted for the adversarial training and robust metric learning approaches using the CARS196 dataset. Table 4.11 displays the list of operations and their durations for the proposed method implementation. Firstly, for the fine-tuning of the pre-trained ResNet-50 model naturally, the training time of the first epoch is calculated as 0.58 minutes, and it is multiplied by 100 epochs, which is advised by Boudiaf et al. [58]. Secondly, adversarial attack generation is completed once, and adversarial samples are saved to be utilized in

the following framework. This operation lasts for 12.7 minutes. Lastly, robust metric learning is implemented for 2.7 minutes.

For the adversarial training approach, training time for an epoch is calculated as 13 minutes and it is multiplied by 100 epochs. When the overall training time needed for our model and the adversarial training is compared based on the results reported in Table 4.12, it is apparent that adversarial training has higher training time due to its nature of complexity, which arises from attack generation and the training of the complex architectures using these synthesized adversarial samples for each iteration.

Table 4.11. Training time analysis for the proposed approach. Training time of the first epoch is measured as 0.58 minutes, and it is multiplied by 100 epochs for the fine-tuning of the pre-trained ResNet-50 model naturally. Adversarial attack generation is completed in 12.7 minutes. Robust metric learning is applied for 2.7 minutes.

Operations	**Required Time (minute)**
FT ResNet-50	58.0
Attack Generation	12.7
Adversarial Metric Learning	2.7

Table 4.12. Training time comparisons. Training time for an epoch is calculated as 13 minutes and it is multiplied by 100 epochs for an adversarial training.

Methods	**Required Time (hour)**
Adversarial Training	21.7
Our Approach	1.2

4.6. Qualitative Results

Metric learning has an essential role in information retrieval tasks, and the retrieval performances can be explained better using qualitative approaches. This section visualizes randomly selected natural and adversarial performances of original ResNet-50 embeddings and the best-performing RML representations. For this aim, query images for natural and adversarial settings are determined, and the k-nearest neighbors algorithm finds the top 5 closest neighbors to these query images. While query natural and adversarial images are demonstrated on the left side in all figures, the top 5 nearest neighbors are displayed on the right. The maximum similarity with the query is represented by the leftmost image of the neighbors, and this similarity decreases from left to right. Since the ranking of classes has great importance in metric learning problems, each sample is also surrounded by its class color to analyze its order among all neighbors.

In Figure 4.1, original ResNet-50 embeddings and the proposed best-performing RML method performances are reported in natural and adversarial settings for CARS196 dataset. It can be realized that first, second and the fifth natural neighbors belong to the same class with the query natural image by considering ResNet-50 embeddings. However, when the PGD attack is applied to query sample, we cannot observe any instances from the same class with query image in the top 5 closest neighbors. It proves that ResNet-50 model is vulnerable to adversarial attacks in retrieval tasks.

As shown in Figure 4.1, RML keeps the natural performances near the ResNet-50 model performances. Similar to ResNet-50, RML also includes the first, second, and fifth natural neighbors belonging to the same class with the query image. It performs superior to the ResNet-50 model in an adversarial setting while having the first two neighbors from the same class with the query image. Although ResNet-50 causes significant variance in adversarial neighbor classes, mismatched third and fifth adversarial neighbors of RML are in the same category with some instances from natural neighbors.

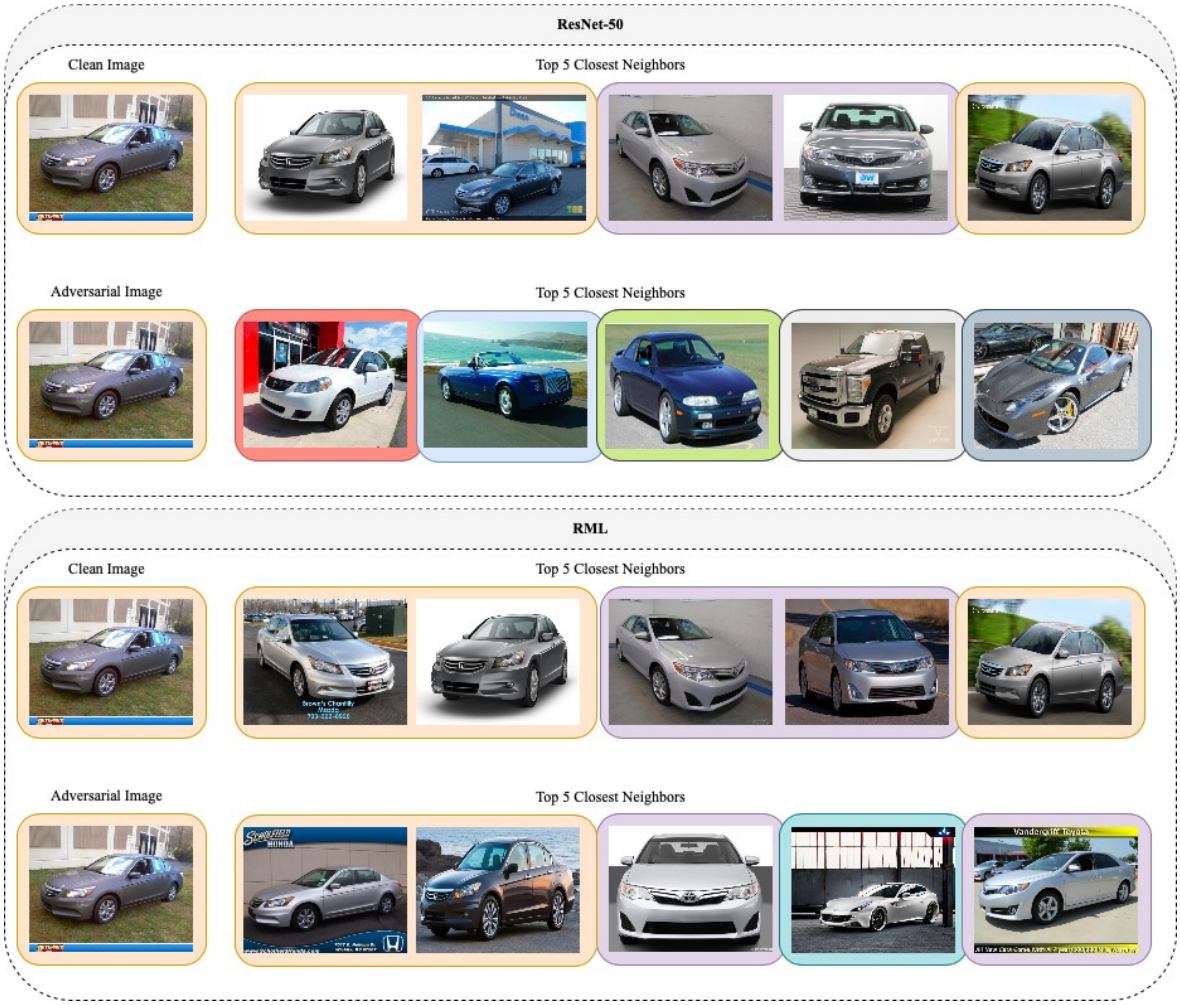


Figure 4.1. Top 5 nearest neighbors comparisons of original ResNet-50 and RML embeddings in natural and adversarial settings for CARS196 dataset.

Figure 4.2 demonstrates the natural and adversarial performance of randomly selected query images from the CUB200-2011 dataset using ResNet-50 and RML embeddings. When the natural query image is forwarded through the model, all the top 5 closest responses are obtained from the same class with the query sample in the ResNet-50 embedding space. Therefore, it is clear that natural performances are at promising levels. However, like the CARS196 dataset, the CUB200-2011 dataset significantly suffers from adversarial perturbations. It can be seen that there is no matching class with the query sample among the top 5 closest neighbors.

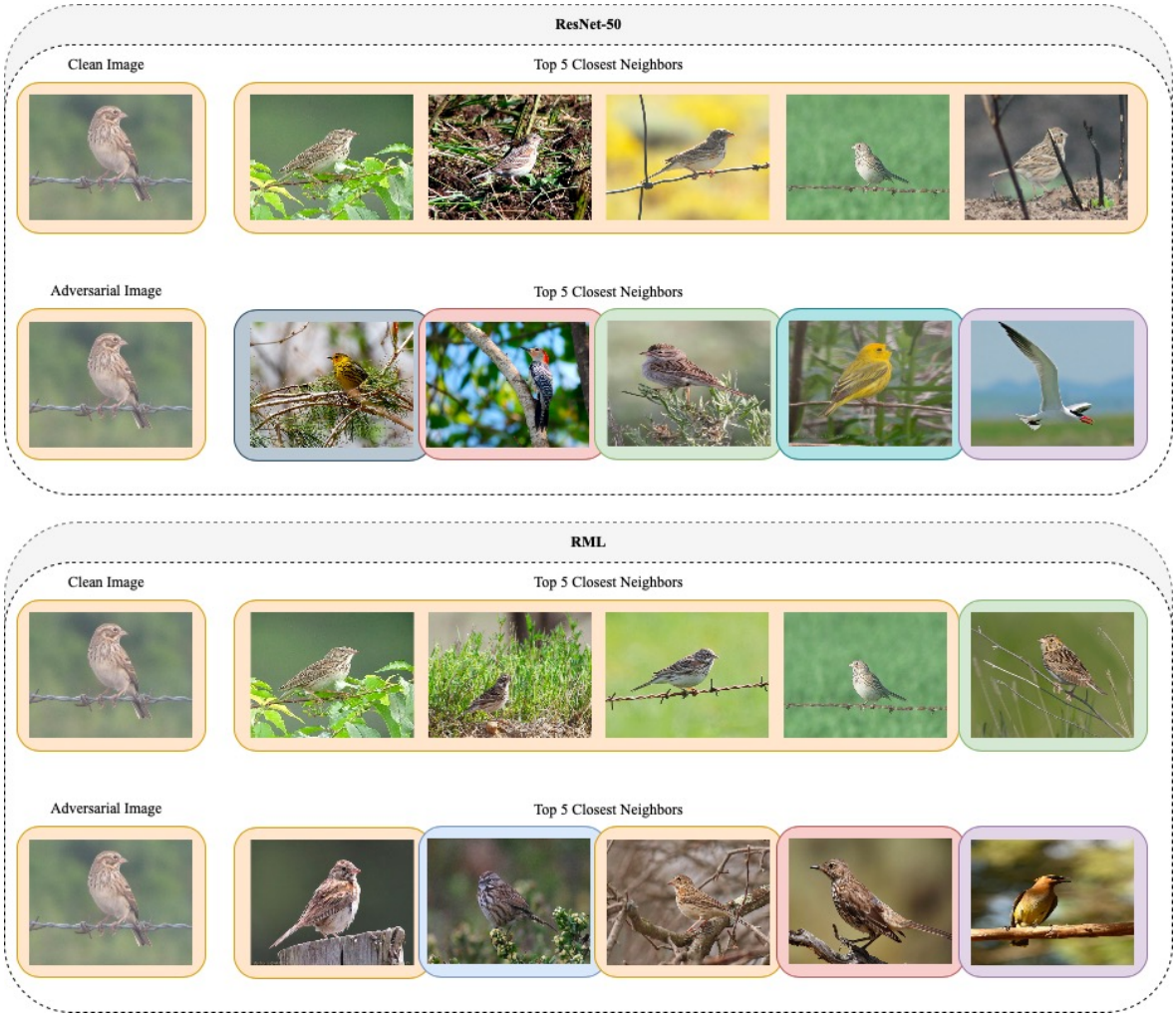


Figure 4.2. Top 5 nearest neighbors comparisons of original ResNet-50 and RML embeddings in natural and adversarial settings for CUB200-2011 dataset.

To address the vulnerability of the ResNet-50 model against the adversarial attacks, the proposed RML module is implemented to the CUB200-2011 dataset, and their results are shared in the lower part of the Figure 4.2. When compared with ResNet-50 performances above, RML shows a minor decrease in the natural setting. The fifth closest image is defined from one of the other classes. On the other hand, as a contribution of the RML module, it boosts the adversarial retrieval performance. While no instance shares the same class as the adversarial query sample in the ResNet-50 experiment, the first and third neighbors are obtained from the category of the anchor image in RML implementations.

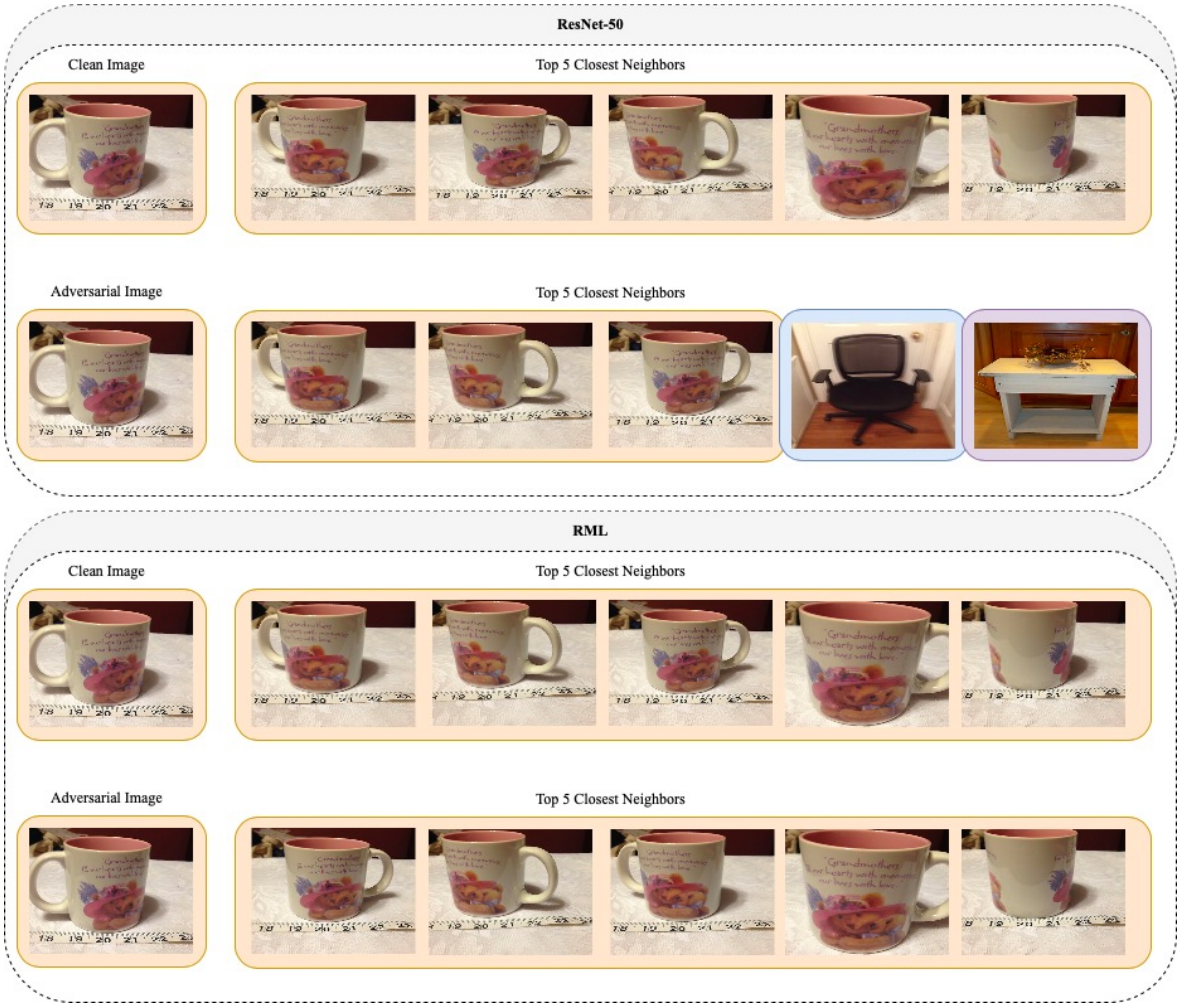


Figure 4.3. Top 5 nearest neighbors comparisons of ResNet-50 and RML embeddings in natural and adversarial settings for SOP dataset.

In Figure 4.3, retrieval performances of natural and PGD-attacked query images are tested with ResNet-50 and RML module embeddings. As we mentioned in previous sections, qualitative evaluations also prove that the SOP dataset is less vulnerable to adversarial attacks when compared with other benchmark datasets. For instance, in experiments with ResNet-50 embeddings, while all of the closest adversarial instances are collected from different classes for the CARS196 dataset, only the fourth and fifth samples belong to another category from the query instance for the SOP dataset.

Even though the SOP dataset has better adversarial robustness against adversarial perturbations, the proposed RML module paves the way for more enhanced robustness, as illustrated in the RML results section of Figure 4.3. The latent space

representations obtained from the RML module provide all neighbors matching with query sample class in natural and adversarial settings. There is no mismatching class in the outputs.

5. CONCLUSION

Carefully designed adversarial perturbations can deceive deep neural networks. In numerous studies, adversarial training is utilized to generate more robust models against these adversarial samples. Even though adversarial training is a widely used and capable technique, it is significant to consider that it also has several drawbacks that must be addressed.

In this thesis, to alleviate the problems in robust metric learning arising from adversarial training, we proposed a robust metric learning model that maps the feature embeddings in the latent space into higher or lower dimensions. Semantic similarities are kept and enhanced during the non-linear projection to get similar samples closer while increasing the distances between different classes. The essential contributions of the thesis can be summarized as follows:

- Adversarial training requires synthesizing adversarial instances within the inner optimization function at each iteration. Furthermore, these newly produced instances are fed to the deep architectures for each iteration to utilize the outer optimization objective. Conversely, in this thesis, adversarial samples are generated once and saved to be utilized in the following metric learning module in a black-box manner. Thus, the training time and complexity are reduced while improving and sometimes preserving the state-of-the-art robustness of models.
- Capturing adversaries within the feature embeddings requires deeper architectures with varying parameters. However, the natural model, which expects more simple neural nets, might be adversely affected and cause overfitting issues. The proposed lightweight metric learning module prevents such problems and maintains natural performances similar to original embeddings.
- Finally, the proposed robust metric learning framework can be considered model-agnostic in the sense that the robust metric learning module is adaptable to different deep backbone architectures. The proposed framework does not require

a specific architecture. Therefore, one can train or fine-tune any off-the-shelf or original architecture for metric learning and then improve its robustness following the proposed robust metric learning module.

This proposed approach experimented with contrastive, triplet, and angular loss functions, which are commonly preferred pair-based losses in the metric learning research area. Moreover, clustering and retrieval performances are evaluated using the three most widely used benchmark datasets in this domain: CUB200-2011, CARS196, and SOP. Experimental results showed that our technique can be more robust against adversarial perturbations than the original embedding performances at the latent space of any architecture. This claim is also tested and we empirically showed the results under different architectural setups, including the deeper and simpler ones.

Our study is evaluated against only a PGD attack, one of the most powerful iterative attacks in the literature. As a future work, model performances can be tested under various attack configurations. For instance, Auto-Attack [28] could be a promising alternative for this challenge since it has been a widespread ensemble attack, and the robustness against this attack is now heavily increased by benefiting from millions of synthetic images like studies listed in RobustBench [55] benchmark list. Metric learning might decrease the need for these vast amounts of synthetic data.

We have improved the robustness by directly forwarding the existing natural and adversarial embeddings into the metric learning module. This method can achieve better performances by applying several data augmentation methods in a latent space of feature embeddings. Exploring these proper data augmentation techniques for each dataset can be the further research area to extend this study.

REFERENCES

1. Mensink, T., J. Verbeek, F. Perronnin and G. Csurka, "Metric Learning for Large Scale Image Classification: Generalizing to New Classes at Near-zero Cost", *European Conference on Computer Vision*, Florance, Italy, pp. 488–501, 2012.

2. Wang, J., Y. Song, T. Leung, C. Rosenberg, J. Wang, J. Philbin, B. Chen and Y. Wu, "Learning Fine-grained Image Similarity with Deep Ranking", *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Columbus, OH, USA, pp. 1386–1393, 2014.

3. Song, H. O., Y. Xiang, S. Jegelka and S. Savarese, "Deep Metric Learning via Lifted Structured Feature Embedding", *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Las Vegas, NV, USA, pp. 4004–4012, 2016.

4. Sanakoyeu, A., V. Tschernezki, U. Buchler and B. Ommer, "Divide and Conquer the Embedding Space for Metric Learning", *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Long Beach, CA, USA, pp. 471–480, 2019.

5. Schroff, F., D. Kalenichenko and J. Philbin, "Facenet: A Unified Embedding for Face Recognition and Clustering", *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Boston, MA, USA, pp. 815–823, 2015.

6. Wang, H., Y. Wang, Z. Zhou, X. Ji, D. Gong, J. Zhou, Z. Li and W. Liu, "Cosface: Large Margin Cosine Loss for Deep Face Recognition", *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Salt Lake City, UT, USA, pp. 5265–5274, 2018.

7. Li, X., X. Yang, Z. Ma and J.-H. Xue, "Deep Metric Learning for Few-shot Image

Classification: A Review of Recent Developments", *Pattern Recognition*, Vol. 138, p. 109381, 2023.

8. Yang, J., D. She, Y.-K. Lai and M.-H. Yang, "Retrieving and Classifying Affective Images via Deep Metric Learning", *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 32, New Orleans, Lousiana, USA, pp. 491–498, 2018.

9. Ermolov, A., L. Mirvakhabova, V. Khrulkov, N. Sebe and I. Oseledets, "Hyperbolic Vision Transformers: Combining Improvements in Metric Learning", *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, New Orleans, LA, USA, pp. 7409–7419, 2022.

10. Roth, K., T. Milbich, B. Ommer, J. P. Cohen and M. Ghassemi, "Simultaneous Similarity-based Self-distillation for Deep Metric Learning", *International Conference on Machine Learning*, Virtual, pp. 9095–9106, 2021.

11. Szegedy, C., W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow and R. Fergus, "Intriguing Properties of Neural Networks", ArXiv:1312.6199 [cs], 2013.

12. He, K., X. Zhang, S. Ren and J. Sun, "Deep Residual Learning for Image Recognition", *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Las Vegas, NV, USA, pp. 770–778, 2016.

13. Papernot, N., P. McDaniel, X. Wu, S. Jha and A. Swami, "Distillation as a Defense to Adversarial Perturbations Against Deep Neural Networks", *IEEE Symposium on Security and Privacy (SP)*, San Jose, CA, USA, pp. 582–597, 2016.

14. Prakash, A., N. Moran, S. Garber, A. DiLillo and J. Storer, "Deflecting Adversarial Attacks with Pixel Deflection", *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Salt Lake City, UT, USA, pp. 8571–8580, 2018.

15. Hwang, U., J. Park, H. Jang, S. Yoon and N. I. Cho, "Puvae: A Variational Autoencoder to Purify Adversarial Examples", ArXiv:1903.00585 [cs], 2019.

16. Mao, C., Z. Zhong, J. Yang, C. Vondrick and B. Ray, "Metric Learning for Adversarial Robustness", *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, Vancouver, Canada, pp. 480–491, 2019.

17. Madry, A., A. Makelov, L. Schmidt, D. Tsipras and A. Vladu, "Towards Deep Learning Models Resistant to Adversarial Attacks", *6th International Conference on Learning Representations*, Vancouver, BC, Canada, p. 1706, 2018.

18. Kurakin, A., I. Goodfellow and S. Bengio, "Adversarial Machine Learning at Scale", ArXiv preprint arXiv:1611.01236 [cs], 2016.

19. Shaham, U., Y. Yamada and S. Negahban, "Understanding Adversarial Training: Increasing Local Stability of Supervised Models through Robust Optimization", *Neurocomputing*, Vol. 307, No. 0925-2312, pp. 195–204, 2018.

20. Jang, Y., T. Zhao, S. Hong and H. Lee, "Adversarial Defense via Learning to Generate Diverse Attacks", *Proceedings of the IEEE/CVF International Conference on Computer Vision*, Seoul, Korea (South), pp. 2740–2749, 2019.

21. Zhang, H. and J. Wang, "Defense Against Adversarial Attacks Using Feature Scattering-based Adversarial Training", *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, Vancouver, Canada, pp. 1831–1841, 2019.

22. Wah, C., S. Branson, P. Welinder, P. Perona and S. Belongie, *Caltech-ucsd Birds 200*, Tech. Rep. CNS-TR-2011-001, California Institute of Technology, California, 2011.

23. Krause, J., M. Stark, J. Deng and L. Fei-Fei, "3D Object Representations for Fine-grained Categorization", *Proceedings of the IEEE International Conference on Computer Vision Workshops*, Sydney, Australia, pp. 554–561, 2013.

24. Chakraborty, A., M. Alam, V. Dey, A. Chattopadhyay and D. Mukhopadhyay,

"Adversarial Attacks and Defences: A Survey", ArXiv:1810.00069 [cs], 2018.

25. Goodfellow, I. J., J. Shlens and C. Szegedy, "Explaining and Harnessing Adversarial Examples", *3rd International Conference on Learning Representations*, San Diego, CA, USA, p. 1412, 2015.

26. Ren, K., T. Zheng, Z. Qin and X. Liu, "Adversarial Attacks and Defenses in Deep Learning", *Engineering*, Vol. 6, No. 3, pp. 346–360, 2020.

27. Yuan, X., P. He, Q. Zhu and X. Li, "Adversarial Examples: Attacks and Defenses for Deep Learning", *IEEE Transactions on Neural Networks and Learning Systems*, Vol. 30, No. 9, pp. 2805–2824, 2019.

28. Croce, F. and M. Hein, "Reliable Evaluation of Adversarial Robustness with an Ensemble of Diverse Parameter-free Attacks", *International Conference on Machine Learning*, Virtual, pp. 2206–2216, 2020.

29. Ghaffari, N. L., D. Truhn, G. P. Veldhuizen, T. Han, M. van Treeck, R. D. Buelow, R. Langer, B. Dislich, P. Boor and V. Schulz, "Adversarial Attacks and Adversarial Robustness in Computational Pathology", *Nature Communications*, Vol. 13, No. 1, p. 5711, 2022.

30. Rossolini, G., F. Nesti, G. D'Amico, S. Nair, A. Biondi and G. Buttazzo, "On the Real-world Adversarial Robustness of Real-time Semantic Segmentation Models for Autonomous Driving", ArXiv preprint arXiv:2201.01850 [cs], 2023.

31. Goldblum, M., L. Fowl, S. Feizi and T. Goldstein, "Adversarially Robust Distillation", *Proceedings of the AAAI Conference on Artificial Intelligence*, New York, NY, USA, pp. 3996–4003, 2020.

32. Jagatap, G., A. Joshi, A. B. Chowdhury, S. Garg and C. Hegde, "Adversarially Robust Learning via Entropic Regularization", ArXiv:2008.12338 [cs], 2020.

33. Wang, J. and H. Zhang, "Bilateral Adversarial Training: Towards Fast Training of More Robust Models Against Adversarial Attacks", *Proceedings of the IEEE/CVF International Conference on Computer Vision*, Seoul, Korea (South), pp. 6629–6638, 2019.

34. Baytaş, İ. M. and D. Deb, "Robustness-via-synthesis: Robust Training with Generative Adversarial Perturbations", *Neurocomputing*, Vol. 516, No. C, pp. 49–60, 2023.

35. Chen, W., X. Chen, J. Zhang and K. Huang, "Beyond Triplet Loss: A Deep Quadruplet Network for Person Re-identification", *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Honolulu, HI, USA, pp. 403–412, 2017.

36. Hadsell, R., S. Chopra and Y. LeCun, "Dimensionality Reduction by Learning an Invariant Mapping", *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, Vol. 2, New York, NY, USA, pp. 1735–1742, 2006.

37. Wang, X., H. Zhang, W. Huang and M. R. Scott, "Cross-batch Memory for Embedding Learning", *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Seattle, WA, USA, pp. 6388–6397, 2020.

38. Cai, B., P. Xiong and S. Tian, "Center Contrastive Loss for Metric Learning", ArXiv:2308.00458 [cs], 2023.

39. Kan, S., Z. He, Y. Cen, Y. Li, V. Mladenovic and Z. He, "Contrastive Bayesian Analysis for Deep Metric Learning", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 45, No. 6, pp. 7220–7238, 2023.

40. Ko, B. and G. Gu, "Embedding Expansion: Augmentation in Embedding Space for Deep Metric Learning", *Proceedings of the IEEE/CVF Conference on Computer*

Vision and Pattern Recognition, Seattle, WA, USA, pp. 7255–7264, 2020.

41. Wang, J., F. Zhou, S. Wen, X. Liu and Y. Lin, "Deep Metric Learning with Angular Loss", *Proceedings of the IEEE International Conference on Computer Vision*, Venice, Italy, pp. 2593–2601, 2017.

42. Zhang, D., Y. Li and Z. Zhang, "Deep Metric Learning with Spherical Embedding", *Proceedings of the 34th International Conference on Neural Information Processing Systems*, Vancouver, Canada, pp. 18772–18783, 2020.

43. Yu, B. and D. Tao, "Deep Metric Learning with Tuplet Margin Loss", *Proceedings of the IEEE/CVF International Conference on Computer Vision*, Seoul, Korea (South), pp. 6490–6499, 2019.

44. Movshovitz-Attias, Y., A. Toshev, T. K. Leung, S. Ioffe and S. Singh, "No Fuss Distance Metric Learning Using Proxies", *Proceedings of the IEEE International Conference on Computer Vision*, Venice, Italy, pp. 360–368, 2017.

45. Qian, Q., L. Shang, B. Sun, J. Hu, H. Li and R. Jin, "Softtriple Loss: Deep Metric Learning without Triplet Sampling", *Proceedings of the IEEE/CVF International Conference on Computer Vision*, Seoul, Korea (South), pp. 6450–6458, 2019.

46. Chan, P. P., S. Li, J. Deng and D. S. Yeung, "Multi-proxy Based Deep Metric Learning", *Information Sciences*, Vol. 643, No. C, p. 119120, 2023.

47. Zhou, M., L. Wang, Z. Niu, Q. Zhang, N. Zheng and G. Hua, "Adversarial Attack and Defense in Deep Ranking", ArXiv:2106.03614 [cs], 2021.

48. Zhou, M., Z. Niu, L. Wang, Q. Zhang and G. Hua, "Adversarial Ranking Attack and Defense", *Computer Vision–ECCV: European Conference*, Glasgow, UK, pp. 781–799, 2020.

49. Panum, T. K., Z. Wang, P. Kan, E. Fernandes and S. Jha, "Exploring Adversarial

Robustness of Deep Metric Learning", ArXiv:2102.07265 [cs], 2021.

50. Zhou, M. and V. M. Patel, "Enhancing Adversarial Robustness for Deep Metric Learning", *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, New Orleans, LA, USA, pp. 15325–15334, 2022.

51. Huai, M., T. Zheng, C. Miao, L. Yao and A. Zhang, "On the Robustness of Metric Learning: An Adversarial Perspective", *ACM Transactions on Knowledge Discovery from Data (TKDD)*, Vol. 16, No. 5, pp. 1–25, 2022.

52. Appiah, B., Z. Qin, A. M. Abra and A. J. A. Kanpogninge, "Decision Tree Pairwise Metric Learning against Adversarial Attacks", *Computers & Security*, Vol. 106, No. 8, p. 102268, 2021.

53. Singh, I., K. Kakizaki and T. Araki, "Advancing Deep Metric Learning Through Multiple Batch Norms and Multi-Targeted Adversarial Examples", ArXiv:2211.16253 [cs], 2022.

54. Chen, C., D. Ye, Y. He, L. Tang and Y. Xu, "Improving Adversarial Robustness with Adversarial Augmentations", *IEEE Internet of Things Journal*, Vol. Early Access, p. 1, 2023.

55. Croce, F., M. Andriushchenko, V. Sehwag, E. Debenedetti, N. Flammarion, M. Chiang, P. Mittal and M. Hein, "RobustBench: A Standardized Adversarial Robustness Benchmark", ArXiv:2010.09670 [cs], 2020.

56. Russakovsky, O., J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla and M. Bernstein, "Imagenet Large Scale Visual Recognition Challenge", *International Journal of Computer Vision*, Vol. 115, No. 3, pp. 211–252, 2015.

57. Musgrave, K., S. Belongie and S.-N. Lim, "A Metric Learning Reality Check", *Computer Vision–ECCV: European Conference*, Glasgow, UK, pp. 681–699, 2020.

58. Boudiaf, M., J. Rony, I. M. Ziko, E. Granger, M. Pedersoli, P. Piantanida and I. B. Ayed, "A Unifying Mutual Information View of Metric Learning: Cross-entropy vs. Pairwise Losses", *Computer Vision–ECCV: European Conference*, Glasgow, UK, pp. 548–564, 2020.

59. Zhai, A. and H.-Y. Wu, "Classification is a Strong Baseline for Deep Metric Learning", *British Machine Vision Conference*, Cardiff, UK, p. 91, 2018.

60. Wu, C.-Y., R. Manmatha, A. J. Smola and P. Krahenbuhl, "Sampling Matters in Deep Embedding Learning", *Proceedings of the IEEE International Conference on Computer Vision*, Venice, Italy, pp. 2840–2848, 2017.

61. Duan, Y., W. Zheng, X. Lin, J. Lu and J. Zhou, "Deep Adversarial Metric Learning", *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Salt Lake City, UT, USA, pp. 2780–2789, 2018.

62. Jegou, H., M. Douze and C. Schmid, "Product Quantization for Nearest Neighbor Search", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 33, No. 1, pp. 117–128, 2010.

63. Wu, Y. and H. Huang, "Understanding Metric Learning on Unit Hypersphere and Generating Better Examples for Adversarial Training", 2022, `https://openreview.net/forum?id=DkeCkhLIVGZ`, accessed on November 5, 2023.

64. Ilyas, A., S. Santurkar, D. Tsipras, L. Engstrom, B. Tran and A. Madry, "Adversarial Examples are not Bugs, They are Features", *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, Vancouver, Canada, pp. 125–136, 2019.