

# UC San Diego

## UC San Diego Electronic Theses and Dissertations

### Title

Learning Toward Object Invariance Across Modalities

### Permalink

<https://escholarship.org/uc/item/68z9m9mq>

### Author

Ho, Chih-Hui

### Publication Date

2024

Peer reviewed|Thesis/dissertation

UNIVERSITY OF CALIFORNIA SAN DIEGO

Learning Toward Object Invariance Across Modalities

A dissertation submitted in partial satisfaction of the  
requirements for the degree Doctor of Philosophy

in

Electrical Engineering (Intelligent Systems, Robotics, and Control)

by

Chih-Hui Ho

Committee in charge:

Professor Nuno Vasconcelos, Chair  
Professor Nikolay Atanasov  
Professor Henrik Christensen  
Professor Xiaolong Wang  
Professor Michael Yip

2024

Copyright

Chih-Hui Ho, 2024

All rights reserved.

The Dissertation of Chih-Hui Ho is approved, and it is acceptable in quality and form for publication on microfilm and electronically.

University of California San Diego

2024

## DEDICATION

To my family.

## EPIGRAPH

Work hard. Dream big.  
Don't take shortcuts. Do the right thing.

*-Bernardo Hees*

## TABLE OF CONTENTS

Dissertation Approval Page .....	iii
Dedication .....	iv
Epigraph .....	v
Table of Contents .....	vi
List of Figures .....	ix
List of Tables .....	xi
Acknowledgements .....	xiii
Vita .....	xv
Abstract of the Dissertation .....	xvii
<b>Chapter 1 Introduction .....</b>	<b>1</b>
1.1 Object Invariance Across Modalities .....	2
1.2 Limitations of Existing Methods .....	2
1.3 Contributions of the Thesis .....	5
1.3.1 Multiview Dataset and Visual Invariant Model .....	6
1.3.2 Supervised Visual Invariant Model .....	7
1.3.3 Self-Supervised Visual Invariant Model .....	7
1.3.4 Object Invariance at Semantic Level .....	8
1.3.5 Outlier of Object Invariance .....	9
1.4 Organization of the Thesis .....	9
<b>Chapter 2 Catastrophic Child’s Play: Easy to Perform, Hard to Defend Adversarial Attacks .....</b>	<b>11</b>
2.1 Introduction .....	12
2.2 Prior work .....	15
2.3 Using pose to attack recognition networks .....	17
2.3.1 Camera shake and pose manipulations .....	18
2.3.2 Characterizing indistinguishable perturbations .....	21
2.3.3 Attacks and defenses .....	23
2.4 Experiments .....	24
2.5 Conclusion .....	27
<b>Chapter 3 PIEs: Pose Invariant Embeddings .....</b>	<b>29</b>
3.1 Introduction .....	30
3.2 Related work .....	33

3.3	Bringing object invariants to the real world . . . . .	36
3.3.1	New view-based and mutiview embeddings . . . . .	36
3.3.2	The need for invariant embeddings . . . . .	37
3.3.3	Pose invariant proxy embedding . . . . .	38
3.3.4	Properties of pose invariant distance . . . . .	39
3.3.5	Generating pose invariant embeddings . . . . .	40
3.3.6	Learning and inference . . . . .	41
3.4	Pose invariance dataset . . . . .	42
3.5	Experiments . . . . .	43
3.5.1	Experimental setup . . . . .	43
3.5.2	Joint classification and retrieval . . . . .	44
3.5.3	Comparison to the state of the art . . . . .	47
3.6	Conclusion . . . . .	49
Chapter 4	Exploit Clues from Views: Self-Supervised and Regularized Learning for Multiview Object Recognition . . . . .	50
4.1	Introduction . . . . .	51
4.2	Related work . . . . .	54
4.2.1	Multiview recognition . . . . .	54
4.2.2	Self-supervised Learning . . . . .	55
4.2.3	Network randomization . . . . .	57
4.3	Multiview Self Supervised Learning . . . . .	57
4.3.1	Light Weight Unsupervised Multiview Object Recognition . . . . .	57
4.3.2	Modeling . . . . .	58
4.3.3	Randomization . . . . .	58
4.3.4	Multiview embeddings . . . . .	60
4.3.5	Multiview consistency regularization . . . . .	61
4.3.6	Scalable Implementation . . . . .	61
4.4	Experiments . . . . .	63
4.4.1	Dataset . . . . .	63
4.4.2	Baselines . . . . .	65
4.4.3	Classification . . . . .	66
4.4.4	Retrieval and Clustering . . . . .	66
4.4.5	Few-shot object recognition . . . . .	67
4.4.6	Dependence on number of objects and views . . . . .	67
4.4.7	Trade-off of training with labels . . . . .	67
4.5	Conclusion . . . . .	69
Chapter 5	ProTeCt: Prompt Tuning for Taxonomic Open Set Classification . . . . .	71
5.1	Introduction . . . . .	72
5.2	Related Work . . . . .	76
5.3	Preliminaries . . . . .	77
5.4	Taxonomic Open Set Classification . . . . .	79
5.5	Prompt Tuning for Hierarchical Consistency . . . . .	82



5.6	Experiments .....	85
5.6.1	TOS Classification Performance .....	88
5.6.2	Domain Generalization of TOS Classification .....	89
5.6.3	Ablation Study and Visualization .....	89
5.7	Conclusion .....	91
Chapter 6	Toward Unsupervised Realistic Visual Question Answering .....	93
6.1	Introduction .....	94
6.2	Related Work .....	97
6.3	RGQA Dataset .....	99
6.3.1	Dataset Curation .....	99
6.3.2	Dataset Analysis .....	101
6.3.3	Evaluation Metrics .....	103
6.4	Unsupervised RVQA Learning .....	104
6.4.1	Unsupervised RVQA .....	104
6.4.2	Training with Pseudo UQ .....	104
6.4.3	RoI Mixup .....	106
6.4.4	Model Ensembling .....	106
6.5	Experiments .....	107
6.5.1	Experimental Set-up .....	107
6.5.2	Quantitative Results .....	109
6.5.3	Qualitative results .....	111
6.5.4	Human Evaluation .....	113
6.6	Conclusion .....	113
Chapter 7	Discussion and Conclusion .....	115
Bibliography	.....	118

## LIST OF FIGURES

Figure 1.1.	Different text queries are given to the deep model. ....	3
Figure 1.2.	Inconsistent prediction of deep model when the object view point changes. ....	3
Figure 1.3.	The existing visual language model fails to provide consistent prediction when questions of different granularity are given. ....	4
Figure 1.4.	Unanswerable questions (UQs) cannot be answered by inspecting the image. ....	5
Figure 1.5.	Three aspects of studying object invariance, including visual invariance, semantic invariance and outlier detection. ....	6
Figure 2.1.	(a) Drone capturing images during flight. (b) Example images collected per viewing angle. (c) Examples of varying levels of camera shake as the drone hovers. ....	18
Figure 2.2.	Turk experiment. ....	20
Figure 2.3.	Per class RR for CS/PV SIP attacks. ....	25
Figure 3.1.	Taxonomy of embeddings learned by different methods according to different level of invariance. ....	30
Figure 3.2.	Embeddings produced by methods at the three levels of invariance of Figure 3.1. In all plots, there are three classes, two objects per class, and each dot represents the embedding of an image. ....	36
Figure 3.3.	Examples of the 8 viewpoints of ObjectPI, for 2 objects. ....	42
Figure 3.4.	TSNE visualization of proxy based embeddings on ObjectPI. Each dot is an object view, objects are identified by color, and their shape descriptors by 'x's. ....	43
Figure 3.5.	Classification accuracy of proxy based embedding on ObjectPI as a function of number of views at inference time. ....	47
Figure 4.1.	Lightweight unsupervised multiview object recognition. A household robot collects multiple object views by moving around, aggregating a multiview object database without view labels. ....	51
Figure 4.2.	Regularization of a self-supervised embedding by prototype randomization. ....	56
Figure 4.3.	TSNE visualization of <b>unseen class</b> embeddings. ....	64

Figure 4.4.	(a) Convergence rate of proposed methods and triplet loss. (b) Effect of different randomization threshold on unseen class accuracy. ....	68
Figure 4.5.	(a) Accuracy (represented by color) of VISPE on unseen classes, as a function of views per object and objects per class in training set. (b) Trade-off of training with labels as a function of object per class. ....	69
Figure 5.1.	(Top) An example of class hierarchy, where CLIP predicts the tiger image as “person” at the internal hierarchy level. ....	73
Figure 5.2.	(Left) Multiple possible label sets are available in a class hierarchy. ....	82
Figure 5.3.	Relative gain/loss after adding ProTeCt. (Left) HCA ; (Right) $Acc_{leaf}$ . ...	86
Figure 5.4.	Ablation of (a) tree dropout rate $\beta$ , (b) NCL strength $\lambda$ and (c) CLIP ViT B32 architecture. ....	86
Figure 5.5.	ProTeCt correctly predicts examples from ImageNet (a,b) and its variants (c,d) at all levels. [GT, Prediction] shows the groundtruth and incorrect prediction by vanilla prompt tuning. ....	87
Figure 6.1.	Realistic VQA. In VQA, a vision system answers a question by inspection of an image. ....	94
Figure 6.2.	Examples of CLIP based (a,b,e,f) and Perturbation (PT) based UQs (c,d,g,h) in RGQA. ....	97
Figure 6.3.	CLIP image-question similarity distribution of both AQs and UQs. ....	101
Figure 6.4.	Comparison between ROC curve (green; right axis) and ACC-FPR curve (orange; left axis). See text for details. ....	102
Figure 6.5.	Illustration of the pseudo UQ and RoI Mixup. The right table shows the label for different visual question inputs. ....	104
Figure 6.6.	Left: RVQA architecture ablation. Right: Human evaluation. ....	110
Figure 6.7.	Qualitative examples for a threshold such that all models achieve 55% accuracy. ....	112
Figure 6.8.	Left: confidence scores of MSP, RP, and Ens methods for 500 random samples. ....	113

## LIST OF TABLES

Table 2.1.	Turker imperceptibility rates for true positive (TP), camera shake (CS), pose variation (PV), and different object (DO) pairs. . . . .	20
Table 2.2.	RRs for CS and PV SIP attacks, under different defense methods and datasets. RRs are averaged over AlexNet, ResNet34 and VGG16. . . . .	25
Table 2.3.	Examples of IPs and SIPs, for CS and PV perturbations, that fool many classifiers. . . . .	26
Table 3.1.	Proxy based methods on ObjectPI. $\alpha = \beta = 1$ for PI-Proxy. . . . .	43
Table 3.2.	Comparison with state of the art methods on ModelNet dataset for 5 different tasks on VGG16. . . . .	46
Table 3.3.	Comparison with state of the art methods on MIRO dataset for 5 different tasks on VGG16. . . . .	46
Table 3.4.	Comparison with state of the art methods on ObjectPI for 5 different tasks on VGG16. . . . .	46
Table 4.1.	KNN classification results for various baselines, solving different surrogate tasks. RSPE outperforms all self-supervised learning methods, VGG16 pretrained model and instance classifiers. . . . .	63
Table 4.2.	Left: Recall @ k and NMI on Modelnet unseen classes. Right: low shot accuracy for k labeled images. . . . .	68
Table 5.1.	TOS classification performance of CLIP-based classifiers. . . . .	74
Table 5.2.	TOS performance with/without ProTeCt on Cifar100 ( $\lambda = 0.5$ ), SUN ( $\lambda = 0.5$ ) and ImageNet ( $\lambda = 1$ ) dataset. $\beta = 0.1$ for all datasets. . . . .	86
Table 5.3.	The gain of hierarchical consistency after adding ProTeCt generalizes across datasets in unseen domains. All methods are fine-tuned on ImageNet and evaluated on its 4 variants. . . . .	87
Table 5.4.	Comparison of CoOp with/without ProTeCt on FGVC Aircraft [131] dataset. . . . .	89
Table 5.5.	CoOp ablation on Cifar100 dataset. Both DTL and NCL loss improve the hierarchical consistency. . . . .	90
Table 5.6.	Improving other adapter-based tuning methods, including CLIP-Adapter and CLIP+LORA with ProTeCt on Cifar100. . . . .	91

Table 6.1. Comparison to previous datasets..... 99

Table 6.2. RVQA comparison of recent VQA models, using MSP for the UQ detector g. 107

Table 6.3. Comparison between different RVQA approaches on AUAF. Cells with light cyan background denote training with pseudo UQs. .... 107

## ACKNOWLEDGEMENTS

PhD is a special journey that I could not finish along by myself. This seven year experience in UCSD has left a deep impression in my mind. I would like to express my deep gratitude to those who accompanied me to make this achievement possible.

First, I would like to express my deepest appreciation to my Ph.D. advisor Professor Nuno Vasconcelos. I would not become who I am today without his guidance. He is definitely one of the most hard working professors that I ever met. I still remember that we said good night to each other on overleaf 3 AM in the morning. His critical thinking, professional writing and research principle helps me established my solid research foundation. I also love his sense of humor :). I am also extremely grateful to Professors Henrik Christensen, Xiaolong Wang, Nikolay Atanasov and Michael Yip for being my committee members. This thesis will not be completed without their mentorship and constructive feedback.

I am also thankful to all my mentors and collaborators of my two internship experiences, Srikar Appalaraju, Bhavan Jasani, R. Manmatha, Zhaowei Cai at Amazon, and Kuan-Chuan Peng at Mitsubishi Electric Research Laboratories (MERL). It is my pleasure to receive their invaluable guidance and insightful critiques. These internships provides the opportunity to collaborate with the top researchers and experience the lifestyle of working in industry.

I am also grateful to my lab mates of SVCL, Tz-Ying Wu, Yi Li, Jiacheng Cheng, Pedro Morgado, Zhiyuan Hu, Brandon Leung, Amir Persekian, Jia Wan, Jiteng Mu, Bo Liu, Zhaowei Cai, Yusheng Li, Xudong Wang, Zhihang Ren, Pin-Ying Wu, Yen Chang, Erik Sandstrom, David Orozco, Sean Kamano, Po Hsiang Huang, Yuwei Zhang, Dacheng Li, Jayi Wang, Yixuan Huang, Deepak Sridhar, Alakh Desai, Jake Pollard, Edward Yang. It is also my pleasure to collaborate with SungBal Seo, NaYeon Kim, YouSuk Bae, JJ Hwang, Shiwei Jin, An Le and Professor Truong Nguyen for over two years. I would also like to thank Simeng Zhang and Professor Paul Siegel for the opportunity of cross-domain collaboration. My life in UCSD would not be so fantastic without them.

Lastly and most importantly, I would love to express my deepest gratitude to my parents

for their unconditional support, both financially and mentally, throughout my PhD study. The most unique thanks is given to my wife, who has accompanied me through all the up-and-downs and the covid periods. I would not be able to survive through my PhD without her support. I would also like to thank my younger brother for spreading his humor and love when I was depressed. I am also grateful for the support from my aunt and my family-in-law during the PhD journey.

Chapter 2 is, in full, based on the material as they appear in the publications of “Catastrophic Child’s Play: Easy to Perform, Hard to Defend Adversarial Attacks”, Chih-Hui Ho\*, Brandon Leung\*, Erik Sandstrom, Yen Chang, and Nuno Vasconcelos, In Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2019 . The dissertation author was the primary investigator and author of this paper.

Chapter 3 is, in full, based on the material as they appear in the publications of “PIEs: Pose Invariant Embeddings”, Chih-Hui Ho, Pedro Morgado, Amir Persekian, and Nuno Vasconcelos, In Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2019 . The dissertation author was the primary investigator and author of this paper.

Chapter 4 is, in full, based on the material as they appear in the publications of “Exploit Clues from Views: Self-Supervised and Regularized Learning for Multiview Object Recognition”, Chih-Hui Ho, Bo Liu, Tz-Ying Wu, and Nuno Vasconcelos, In Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2020 . The dissertation author was the primary investigator and author of this paper.

Chapter 5 is, in full, based on the material as they appear in the publications of “ProTeCt: Prompt Tuning for Hierarchical Consistency”, Chih-Hui Ho\*, Tz-Ying Wu\*, and Nuno Vasconcelos, In Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2024 . The dissertation author was the primary investigator and author of this paper.

Chapter 6 is, in full, based on the material as they appear in the publications of “Toward Unsupervised Realistic Visual Question Answering”, Chih-Hui Ho\*, Yuwei Zhang\*, and Nuno Vasconcelos, International Conference on Computer Vision (ICCV), 2023 . The dissertation author was the primary investigator and author of this paper.

## VITA

- 2016 B.S. in Electrical Engineering and Computer Science, National Chiao Tung University, Taiwan
- 2019 M.S. in Computer Science, University of California San Diego, United States
- 2024 Ph. D. in Electrical Engineering (Intelligent Systems, Robotics & Control), University of California San Diego, United States

## PUBLICATIONS

**Chih-Hui Ho\***, Kuan-Chuan Peng, and Nuno Vasconcelos, “Long-Tailed Anomaly Detection with Learnable Class Names”, *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024

**Chih-Hui Ho\***, Tz-Ying Wu\*, and Nuno Vasconcelos, “ProTeCt: Prompt Tuning for Hierarchical Consistency”, *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024

**Chih-Hui Ho\***, Yuwei Zhang\*, and Nuno Vasconcelos, “Toward Unsupervised Realistic Visual Question Answering”, *International Conference on Computer Vision (ICCV)*, 2023

**Chih-Hui Ho**, SungBal Seo, NaYeon Kim, Pin-Ying Wu, YouSuk Bae, Nuno Vasconcelos, “Unsupervised PCB Anomaly Segmentation with Foundational Model”, *Electronic Imaging (EI)*, Oral, 2023.

**Chih-Hui Ho**, and Nuno Vasconcelos, “DISCO: Adversarial Defense with Local Implicit Functions”, *Conference on Neural Information Processing Systems (NeurIPS)*, 2022

**Chih-Hui Ho**, Srikar Appalaraju, Bhavan Jasani, R. Manmatha, and Nuno Vasconcelos, “YORO - Lightweight End to End Visual Grounding”, *European Conference On Computer Vision Workshop (ECCVW)*, 2022

Brandon Leung, **Chih-Hui Ho**, and Nuno Vasconcelos, “Black-Box Test-Time Shape REFINEMENT for Single View 3D Reconstruction”, *Conference on Computer Vision and Pattern Recognition Workshop (CVPRW)*, 2022

**Chih-Hui Ho**, Ziheng Huang, NaYeon Kim, YouSuk Bae, and Nuno Vasconcelos, “Tire Defect Detection with Limited Annotation”, *Electronic Imaging (EI)*, Oral, 2022.

**Chih-Hui Ho**, and Nuno Vasconcelos, “Contrastive Learning with Adversarial Examples”, *Conference on Neural Information Processing Systems (NeurIPS)*, 2020



Tz-Ying Wu, Pedro Morgado, Pei Wang, **Chih-Hui Ho**, and Nuno Vasconcelos, “Solving Long-tailed Recognition with Deep Realistic Taxonomic Classifier”, *European Conference On Computer Vision (ECCV)*, 2020

**Chih-Hui Ho**, Bo Liu, Tz-Ying Wu, and Nuno Vasconcelos, “Exploit Clues from Views: Self-Supervised and Regularized Learning for Multiview Object Recognition”, *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020

**Chih-Hui Ho**, Pedro Morgado, Amir Persekian, and Nuno Vasconcelos, “PIEs: Pose Invariant Embeddings”, *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019

**Chih-Hui Ho\***, Brandon Leung\*, Erik Sandstrom, Yen Chang, and Nuno Vasconcelos, “Catastrophic Child’s Play: Easy to Perform, Hard to Defend Adversarial Attacks”, *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019

## ABSTRACT OF THE DISSERTATION

Learning Toward Object Invariance Across Modalities

by

Chih-Hui Ho

Doctor of Philosophy in Electrical Engineering (Intelligent Systems, Robotics, and Control)

University of California San Diego, 2024

Professor Nuno Vasconcelos, Chair

Humans possess a remarkable ability to identify and pinpoint objects within complex environments, even from partial views or descriptions. For example, when we are asked to “grab a green mug next to coffee machine”, we can easily identify the green mug regardless of what view of the green mug is presented to us and distinguish it from other objects. This requires the model to comprehend the instruction, pick up the correct object, and be invariant with object views.

In this thesis, I start investigating object invariance from systematically analyzing what are the challenging views of an object such that the existing models fail to recognize the object. We show that existing models cannot recognize the object from challenging object views. To address this problem, I proposed a training method “PIE” to encourage the model to learn a

structured feature space that is view-invariant. However, PIE learns the structured feature space in supervised manner, which requires annotation from human. To overcome this limitation, I further proposed VISPE to learn the view-invariant feature in a self-supervised manner for multiview classification and retrieval tasks.

In addition to learning invariance with images, the use of language as an abstraction of invariant object representation is investigated. I first discover that the existing visual language models fails to produce a consistent prediction when the object class name changes to different granularity (e.g. from Bengal tiger to tiger). A novel prompt tuning methods is proposed to regularize model prediction and to better align language and vision. However, it remains a question whether these visual language models can realize the existence of an object. This is investigated by curating a visual question answering dataset that contains “unanswerable questions”, where the referred object is not in the image. An unsupervised approach is proposed to encourage the model to be aware of unanswerable questions. These efforts have improved the understanding of how to learn object invariance across modalities.

# **Chapter 1**

## **Introduction**

## 1.1 Object Invariance Across Modalities

Human is good at identifying and localizing an object in a complex scene by observing part of the object's views or through a sentence that refers to the object. This is an interesting property of human perception that machine is not at a human level. For example, when we are looking at a red armchair, we realize that it is the same armchair from different viewing angles and we can consistently recognize it regardless of what view of the armchair is shown to us. Furthermore, we can identify fine-grained difference between objects, such as the chair with and without an arm.

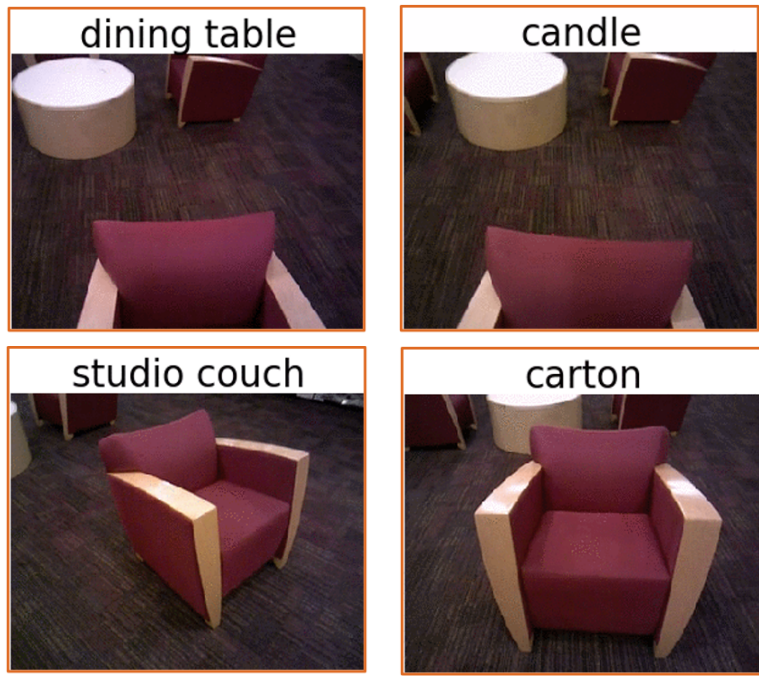
More importantly, human has the ability to identify the object referred by a sentence. For example, as shown in Figure 1.1, we are able to comprehend the sentences with both coarse-grained or fine-grained description such as "Red object next to the shelf" or "the red armchair on the right" and then locate the chair based on the description. However, when we are asked to locate the bottle under the table, which does not exist, we might response "There is no bottle under the table. I cannot find it". This indicates that human not only have the ability to locate the referred object, but also able to spot the mismatch between the query sentence and the image. The above observations indicate that there exists an invariant representation of the object of interest in our mind which correlates the underlying 3D structure from a 2D image and the referred sentence.

## 1.2 Limitations of Existing Methods

While there exists an object invariant representation in human's mind, this is not the case for current computer vision models. For example, as shown in Figure 1.2, when different views of an red arm chair are presented to a deep model [167, 65, 101], the deep model has inconsistent predictions. Such inconsistency will hinder the development of many downstream applications. For example, consider an augmented reality (AR) app for safari tour, where the device has to preform consistent animal recognition when the animal is walking around. In addition, the device needs to simultaneously support fine-grained classification (e.g. classifying Malayan Tiger vs Bengal Tiger) for a zoologist and coarse-grained classification for a tourist (e.g. classifying Tiger



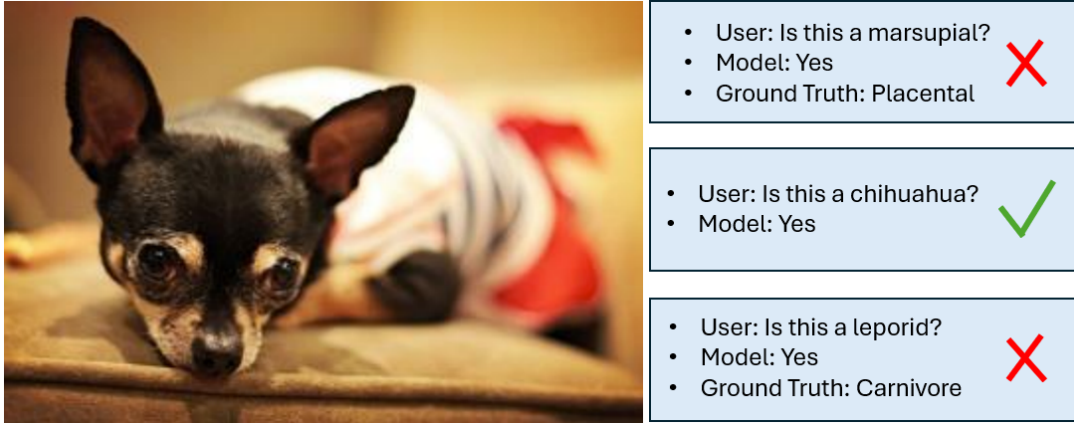
**Figure 1.1.** Different text queries are given to the deep model. For the query object that exists in the image, the model is expected to localize the object (i.e. on the right). On the other hand, the model should understand what are the objects that does not exist in the image.



**Figure 1.2.** Inconsistent prediction of deep model when the object view point changes.

vs Bobcat). Finally, the user might ask “Where is the pregnant Bengal Tiger? Can you locate it for me?” to the device, and the device will need to pinpoint the specific Bengal tiger for the user.

To achieve these above goal, the device has to comprehend the user’s question and find the object in the complex scene. Recently, some of the visual language models [159, 228, 227, 94, 83] demonstrate the zero-shot classification ability, which allows the user to input any label set that



**Figure 1.3.** The existing visual language model fails to provide consistent prediction when questions of different granularity are given.

the user care about. For example, the user can input the label set of (“cat”, “tiger”, “frog”) or (“cup”, “tiger”, “person”) to the model and the model is expected to predict “tiger” under both label sets. While these model can potentially be leveraged for accomplishing the goal, it remain unknown whether these visual language models are robust when (1) label set covers object classes from different granularity (also referred to as *open granularity classification*) and (2) there are mismatches between the image and text (i.e. asking a question about a dog, while there is not dog in the image).

The task of open granularity classification aims to learn a visual language model that have consistent prediction for a chihuahua image when the label set changes from (“cat”, “dog”, “frog”) to (“cat”, “chihuahua”, “frog”). To investigate whether the existing method supports both fine-grained and coarse-grained classification, we evaluate the state-of-the-art visual language models using the hierarchy label set from ImageNet [33] dataset. As shown in Figure 1.3, while the model predicts the image as a chihuahua, it also predicts the image as marsupial and leporid, which are incorrect. This indicates that even these large model are not able to tackle the open-granularity classification task. Furthermore, to evaluate whether the existing model truly understand the user’s question and provide a response with high reliability, we curate a dataset that contains *unanswerable questions (UQs)*. UQ refers to the questions that the deep models are



On which side is the **doll**?



What is parked near the **piano**  
the nearest **traffic light** is  
across from?



What is the item of furniture to  
the left of the light **white**  
**couch**?

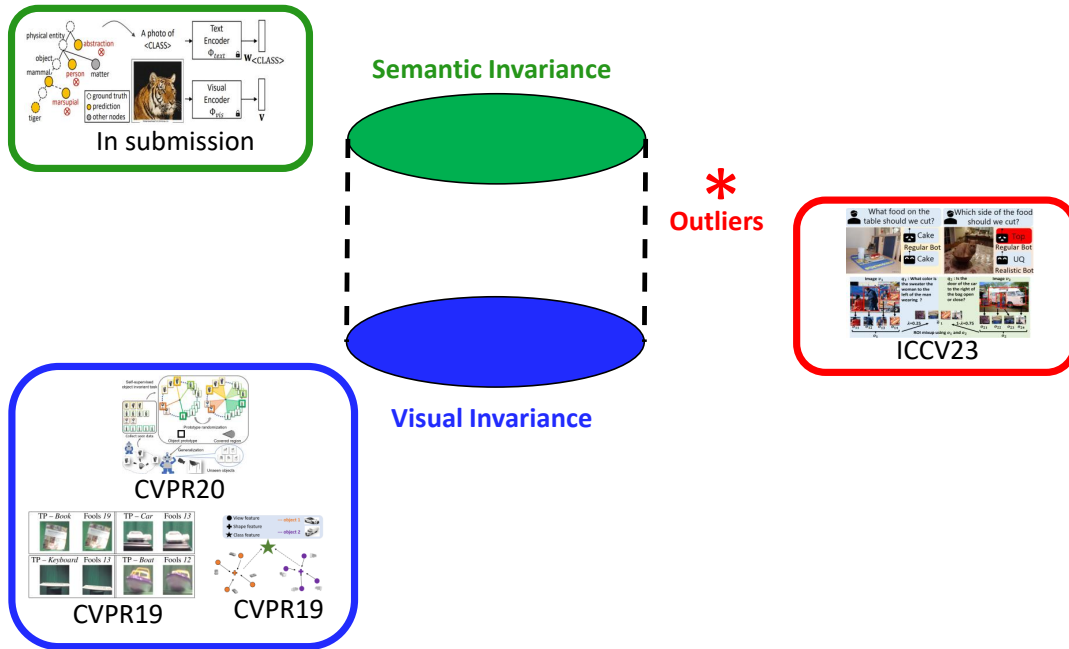
**Figure 1.4.** Unanswerable questions (UQs) cannot be answered by inspecting the image. The red text indicates the mismatch between text and image that cause the UQ.

not able to answer by inspecting the visual information (i.e. image). Figure 1.4 shows three examples of UQ from the proposed dataset, where the question cannot be answered by looking at the image. For example, the question “On which side is the doll?” is an UQ, because there is no doll in the image. While human can easily reject to answer those UQs, our study shows that this is not the case for existing deep models, which is not trustworthy and can hallucinate the answers.

### 1.3 Contributions of the Thesis

In this thesis, we focus on studying the object invariance representation for machines at visual level and semantic level. For visual invariance, we study the robustness of existing deep models when different views of an object are presented to them. A supervised and an unsupervised approach are designed to equip these deep model to be view invariant. For the semantic invariance level, we consider language as an abstraction of invariant object representation and study the robustness of existing visual language models when the text phrase that describe the object changes. Finally, we also encourage the model to be aware of the outliers of semantic invariance, which are the mismatch between language and image. Figure 1.5 illustrates the three aspects of studying object invariance in this thesis.





**Figure 1.5.** Three aspects of studying object invariance, including visual invariance, semantic invariance and outlier detection.

### 1.3.1 Multiview Dataset and Visual Invariant Model

When different view of an object are presented to CNN models, their prediction consistency is investigated. To study the robustness of CNN models, we consider the problem of adversarial CNN attacks, with an emphasis on attacks that are trivial to perform but difficult to defend. A framework for the study of such attacks is proposed, using real world object manipulations. Unlike most works in the past, this framework supports the design of attacks based on both small and large image perturbations, implemented by camera shake and pose variation. A setup is proposed for the collection of such perturbations and determination of their perceptibility. It is argued that the latter depends on context, and a distinction is made between imperceptible and semantically imperceptible perturbations. While the former survive image comparisons, the latter are perceptible but have no impact on human object recognition. A procedure is proposed to determine the perceptibility of perturbations using Turk experiments, and a dataset of both perturbation classes which enables replicable studies of object manipulation attacks, is assembled.

Experiments using defenses based on many datasets, CNN models, and algorithms from the literature elucidate the difficulty of defending these attacks. In fact, none of the existing defenses is found effective against them. Better results are achieved with real world data augmentation, but even this is not foolproof. These results confirm the hypothesis that current CNNs are vulnerable to attacks implementable even by a child, and that such attacks may prove difficult to defend.

### **1.3.2 Supervised Visual Invariant Model**

To encourage the robustness of deep models, the role of view (or pose) invariance in image recognition and retrieval is studied. A taxonomic classification of embeddings, according to their level of invariance, is introduced and used to clarify connections between existing embeddings, identify missing approaches, and propose invariant generalizations. This leads to a new family of pose invariant embeddings (PIEs), derived from existing approaches by a combination of two models, which follow from the interpretation of CNNs as estimators of class posterior probabilities: a view-to-object model and an object-to-class model. The new pose-invariant models are shown to have interesting properties, both theoretically and through experiments, where they outperform existing multiview approaches. Most notably, they achieve good performance for both 1) classification and retrieval, and 2) single and multiview inference. These are important properties for the design of real vision systems, where universal embeddings are preferable to task specific ones, and multiple images are usually not available at inference time. Finally, a new multiview dataset of real objects, imaged in the wild against complex backgrounds, is introduced. We believe that this is a much needed complement to the synthetic datasets in wide use and will contribute to the advancement of multiview recognition and retrieval.

### **1.3.3 Self-Supervised Visual Invariant Model**

While multiview recognition has been well studied in the literature and achieves decent performance in object recognition and retrieval task, most previous works rely on supervised learning. In addition, they rely on some impractical underlying assumptions, such as the avail-

ability of all views in training and inference time. In this work, the problem of multiview self-supervised learning (MV-SSL) is investigated, where only image to object association is given. Given this setup, a novel surrogate task for self-supervised learning is proposed by pursuing “object invariant” representation. This is solved by randomly selecting an image feature of an object as object prototype, accompanied with multiview consistency regularization, which results in view invariant stochastic prototype embedding (VISPE). Experiments shows that the recognition and retrieval results using VISPE outperform that of other self-supervised learning methods on seen and unseen data. VISPE can also be applied to semi-supervised scenario and demonstrates robust performance with limited data available.

### 1.3.4 Object Invariance at Semantic Level

The study of object invariance is extended to the visual language domain, where the language is used as an abstraction to represent the object. Visual-language foundation models, like CLIP, learn generalized representations that enable zero-shot open-set classification. Few-shot adaptation methods, based on prompt tuning, have been shown to further improve performance on downstream datasets. However, these methods do not fare well in the *taxonomic open set* (TOS) setting, where the classifier is asked to make prediction from label set across different levels of semantic granularity. Frequently, they infer incorrect labels at coarser taxonomic class levels, even when the inference at the leaf level (original class labels) is correct. To address this problem, we propose a prompt tuning technique that calibrates the hierarchical consistency of model predictions. A set of metrics of hierarchical consistency, the Hierarchical Consistent Accuracy (HCA) and the Mean Treecut Accuracy (MTA), are first proposed to evaluate TOS model performance. A new *Prompt Tuning for Hierarchical Consistency* (ProTeCt) technique is then proposed to calibrate classification across label set granularities. Results show that ProTeCt can be combined with existing prompt tuning methods to significantly improve TOS classification without degrading the leaf level classification performance.

### 1.3.5 Outlier of Object Invariance

Unlike previous projects that encourages invariant features across modalities, we also encourage the model to be aware of those outliers, where the objects referred by language description does not match the object in the image. To investigate this, we considered the problem of realistic VQA (RVQA), where a model has to reject unanswerable questions (UQs) and answer answerable ones (AQs). We first point out 2 drawbacks in current RVQA research, where (1) datasets contain too many unchallenging UQs and (2) a large number of annotated UQs are required for training. To resolve the first drawback, we propose a new testing dataset, RGQA, which combines AQs from an existing VQA dataset with around 29K human-annotated UQs. These UQs consist of both fine-grained and coarse-grained image-question pairs generated with 2 approaches: CLIP-based and Perturbation-based. To address the second drawback, we introduce an unsupervised training approach. This combines pseudo UQs obtained by randomly pairing images and questions, with an RoI Mixup procedure to generate more fine-grained pseudo UQs, and model ensembling to regularize model confidence. Experiments show that using pseudo UQs significantly outperforms RVQA baselines. RoI Mixup and model ensembling further increase the gain. Finally, human evaluation reveals a performance gap between humans and models, showing that more RVQA research is needed.

## 1.4 Organization of the Thesis

The rest of the thesis is structured as follows. Chapter 2 introduces a new multiview dataset captured by hovering drone and the dataset is used to probe the deep classifier to measure their classification consistency. We found that the classifier have inconsistent prediction when different view of a same object is shown to the classifier. In Chapter 3, in order to encourage the deep classifier to be view invariant, we introduce a new loss function PIE, which can be trained in supervised manner. In Chapter 4, we propose VISPE as a self-supervised learning alternative to learn the view invariance. In Chapter 5, we show that the recent vision language model, which

can classify an object based on the user label set, is not robust to the change of label set across granularity and hierarchy. A novel training framework ProTeCt is then proposed to assist the existing vision language models to have hierarchical consistency. Chapter 6, we investigate the robustness of existing visual language models on question answering task by presenting them unanswerable questions. We also propose a training procedure to help the model to spot the mismatch between the query and the image. Chapter 7, we summarize and conclude the thesis.

## **Chapter 2**

# **Catastrophic Child's Play: Easy to Perform, Hard to Defend Adversarial Attacks**

## 2.1 Introduction

Convolutional neural networks (CNNs) trained on large corpora such as ImageNet [34] have enabled significant advances in computer vision in recent years. While initially popular for recognition, these models have shown to be remarkably easy to train and transfer across vision tasks, and are now almost universally used across computer vision. Recently, however, this robustness has been questioned by some puzzling findings derived from adversarial CNN attacks [102, 150, 140, 17, 139, 177]. Although CNNs have excellent, even superhuman [66], recognition performance on randomly internet-collected test images, it is quite easy to generate images where they fail dramatically [180, 56]. In fact, most images that a CNN classifies correctly can be transformed into images that it cannot classify, by the addition of a very small adversarial perturbation. Most interestingly, this perturbation can usually be made so small as to be imperceptible, i.e. impossible to detect, by a human. This suggests that the space of images correctly classified by most CNNs is, at most, a countable dense subset of the space of images recognizable by humans, e.g. similar to the relationship between rational and real numbers.

This problem is of great concern for many applications. For example, smart cars depend on CNNs to make decisions that could have life or death consequences, security and surveillance systems rely on CNNs for identity verification, etc. The existence of many images capable of fooling CNNs poses a significant challenge to such applications. This has spurred interest in adversarial attacks [43, 102] and a literature has emerged in the area, with many variants of the problem being proposed. In result, there are at least four fundamental dimensions along which adversarial algorithms can differ: they can be 1) “white” [180, 56, 102, 150, 140, 17, 139] or “black”-box [22, 224, 177, 42, 77], depending on whether knowledge of the CNN model to attack is required, 2) “targeted” or “non-targeted,” depending on whether the goal is to induce the network to make specific errors [180, 150, 17] or to simply make an error [56, 102, 140, 139, 224], 3) “digital” or “real-world” depending on whether the examples used in the attack are produced by an algorithm [180, 56, 177] vs. object manipulation in the real world [102, 43, 8], and 4) “single

model” or “universal” depending on whether they aim to fool a single network [102, 140, 17, 224] or many models [139]. Interestingly enough, the relative difficulty of these problems does not always correlate with what would be intuitively expected. For example, it appears that most of the attacks designed to fool a particular CNN, e.g. AlexNet [101], also fool most other CNNs [149, 185, 103, 124, 215], e.g. VGG [168], Inception [179], or ResNet [67]. Similarly, some “black-box” attacks involving simple image transformations [42, 77] appear to be much more effective than “white-box” methods that require access to the CNN and optimization based on backpropagation style of algorithms.

In general, however, it can be quite difficult to compare the merits of different algorithms. This is due to two main problems. First, most methods use perturbations that cannot be easily compared. While many authors rely on the standard of an “image that is perceptually indistinguishable from the original” to define a valid attack, it is not clear what the boundaries of “indistinguishable” are and no attempts have been made to define this concept. Instead, the standard is usually met by adoption of a very conservative attack strategy, e.g. the use of an “infinitesimally small step along some gradient direction”. It is frequently unclear if the use of larger perturbations would enable the same algorithm to produce more successful attacks. Second, most adversarial works do not even attempt to compare performance with previous approaches. This is unlike most other areas of computer vision, where the ability to compare algorithms is considered critical to evaluate progress.

Recently, some works have started to address the second problem through a strategy that we denote as the “arms race”. This exploits the fact that any attack procedure can be transformed into a defense, by 1) augmenting the training set, e.g. ImageNet, with examples produced by the procedure and 2) fine-tuning the network. While not guaranteeing full robustness against the attack [103, 185, 78], this defense strategy renders most attacks much less effective. Under the “arms race” paradigm, a new attack strategy is considered state of the art if it fools a network that implements defenses to previously known attacks [208]. The “arms race” captures the fact that, for practical applications, the only significant attacks are those for which no defenses are



available. However, while knowing the attack procedure enables a defense, not all attacks are equally easy to defend. An important variable is the defense’s cost. For example, attacks that require more computation to defend against are more costly than attacks that can be thwarted with little computation. Similarly, attacks have different costs. For example, white box attacks can be rendered impractical by the simple use of a proprietary CNN. Overall, the most concerning attacks are those easiest to execute and hardest to defend against.

In this work, we consider the design of such attacks. We argue that the most successful attacks are those that leverage the limitations of computer vision, namely those based on perturbations that are easily produced by people but cannot be replicated by computers. This exploits the large imbalance between the cost of attack and defense in terms of the number of required examples. While an attack requires a few well chosen examples, its defense requires augmenting the training set with an extensive number of examples. Hence, while attacks can be generated manually, those that cannot be defended with computer generated examples are impractical to defend against. We then consider a set of image perturbations based on *variation of object pose*. This is an operation that can be implemented by a child (simply by rotating an object) but is very hard to defend against, due to the well known difficulty of synthesizing objects under different poses [151, 204]. We consider attacks using both small and large perturbations, due to *camera shake* (CS) and *pose variations* (PV). However, the study of such attacks requires a definition of which perturbations are valid. After all, extreme poses can confuse even humans. Unfortunately, common definitions, such as “infinitesimal gradient steps” or imperceptibility on side-by-side image comparisons, are not suitable for *large* perturbations. We argue that these can only be declared imperceptible given an attack context and seek definitions of imperceptibility suited for the object recognition context. This suggests a distinction between imperceptible perturbations (IPs), which survive image comparisons, and semantic IPs (SIPs), which are perceptible on image comparisons but have no impact on human ability to recognize objects.

Overall, this work makes three contributions to the study of adversarial attacks on CNNs. The first is a dataset of images of multiple object classes under CS and PV. The object classes

are a subset of ImageNet, to enable the attack of ImageNet trained CNNs, and each object is imaged with extensive coverage of both small (CS) and large (PV) view variability. The second contribution is a procedure to determine which perturbations are imperceptible to humans, using Amazon Turk experiments. The procedure is designed to support many attack contexts and could be used to characterize many other types of attacks. We consider two contexts, image and object retrieval, that enable the differentiation between IPs and SIPs for object recognition; these can be thought of as small vs. large perturbations. A dataset containing CS and PV perturbations of the two types is finally assembled, to support the study of recognition attacks. A final contribution is an extensive experimental study of CS and PV attacks’ performance, against multiple CNN models, trained on multiple datasets, and augmented with multiple defenses from the literature. This shows that pose attacks are highly successful against existing CNNs, previous defenses are ineffective against them, and even data collection can have limited effectiveness. Thus, while easy to perform, pose attacks can be difficult to defend.

## **2.2 Prior work**

There is now a significant literature on adversarial attacks. The most popular setting is a non-targeted white-box digital attack of a single model [56, 102, 140]. The attack is usually an image perturbation based on an infinitesimal step along the gradient of the loss used to train the model, evaluated at the image [56, 102]. The simplest such attacks reduce to one back-propagation iteration, computing derivatives with respect to the input image, and require a forward and backward pass through the network [56]. Many variants have been proposed, including different algorithms [150, 17, 124] or slight variations on the problem. For example, [139] proposed similar techniques for universal attacks, i.e. perturbations that fool many models, and [180, 150, 17] considered targeted attacks. These aim to induce specific errors, e.g. the classification of “apples” as “oranges”, using a somewhat more sophisticated optimization. All these methods are digital and can, in principle, be defended against by using the attack algorithm

to generate augmentation data to retrain the CNN.

More recently, there has been interest in attacks based on object manipulation in the real world [102, 43, 8]. Some of these address specific applications, such as recognition by smart cars. For example, [43] investigated attacks based on the addition of stickers to traffic signs. This is much less general than the attacks now proposed, which can be applied to any object. Others have investigated the manipulation of images in the world, or the fabrication of objects with certain properties. For example, [8] devised an interesting procedure to fabricate objects that can consistently fool CNNs irrespective of viewing angle. While having some similarities to the attacks now proposed, this setup is substantially more complex than the one presented, which does not require object fabrication. Fabrication raises the cost of attack, by requiring access to and knowledge of object fabrication, and drastically reduces the cost of defense, since it relies on algorithms that can be leveraged to produce defenses digitally. For example, because the objects fabricated by [8] have digital textures, their images can be rendered by computer. This is unlike real objects and textures, which are well known to be difficult to capture and render accurately under pose variation [204].

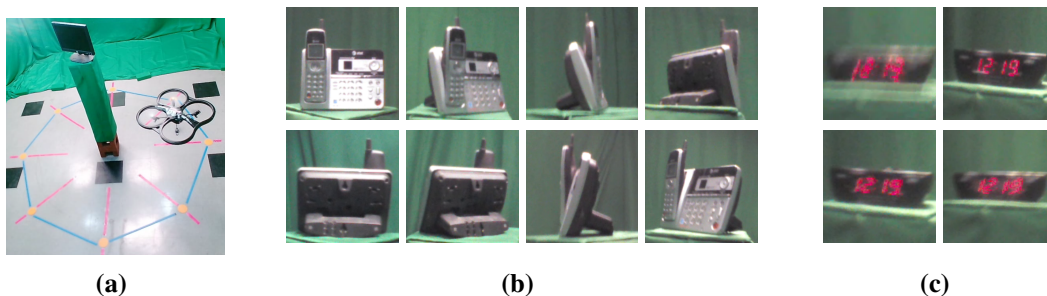
Perhaps most related to this work are previous efforts based on image transformations. For example, [42] has shown that black box attacks by simple image rotation can fool CNNs more effectively than white-box attacks based on gradient optimization. A recent extension of this idea uses spatial transformer networks to synthesize image transformation attacks more general than rotations [208]. This work again showed that image transformations are successful even on networks that implement defenses against gradient attacks. However, all these methods implement digital attacks, using algorithms that can in turn be exploited to defend against them. We propose a setting that generalizes these procedures, relying on real world image manipulation. This is much harder to defend against.

## 2.3 Using pose to attack recognition networks

There are several challenges to the study of adversarial attacks. A meaningful attack requires two images: a *true positive*  $x$ , i.e. a successfully classified image, and a perturbation  $x'$ . A first difficulty is that  $x'$  should be, in some sense, “identical” to  $x$ . Otherwise, it is illogical to ask the classifier to assign it to the same class, and the attack is ill-defined. We refer to this as the problem of *attack validity*. Consider the popular framework of attacks based on additive perturbations,  $x' = x + \eta \delta$ , where  $\delta$  is a function of the gradient of the classification loss with respect to  $x$  [103]. In the absence of a criterion to test whether  $x$  and  $x'$  are “identical”, validity is sought by constraining  $\eta$  to be very small, so as to make  $x'$  *visually indistinguishable* from  $x$ . However, this is not a full guarantee of validity, since a person with infinite time can frequently identify the perturbed image. There can also be moiré-like interference patterns that easily give the perturbation away. Some methods attempt to address the problem by thresholding the gradient, but this can produce salt-and-pepper artifacts. In general, it is difficult to guarantee that  $x$  and  $x'$  are indistinguishable.

For these methods, the validity problem follows from the lack of realism in the perturbations used for the attack. We refer to this as the *realism problem*. The difficulty is that  $\delta$  is *not* a natural image. Hence, the methods above simply produce images at the “edge” of the space of natural images. While overly large steps along  $\delta$  produce completely unrealistic images, a small enough  $\eta$  guarantees they are acceptable. Yet, because the perturbed images do not occur in the real world, the perturbations must be very small for the attack to remain valid. This leads to a third problem, which is the *small perturbation problem*, i.e. exclusion of attacks that are not immediate neighbors of the true positive. For most applications, such attacks are a much stronger concern than infinitesimal steps towards the edge of image space. For example, the shake and pose attacks proposed in this work can occur *naturally* during the operation of a vision system. This also implies that they are much easier to perform and thus much more likely to be executed – imagine a world where any child can hack a robot simply by showing it familiar objects in strange poses.

In summary, because there is lack of realism, validity can only be guaranteed by small



**Figure 2.1.** (a) Drone capturing images during flight. (b) Example images collected per viewing angle. (c) Examples of varying levels of camera shake as the drone hovers.

perturbations. This has motivated a recent emergence of perturbations  $x' = f(x)$  where  $f$  is no longer additive. Various functions have been proposed, from affine transformations [42] to affixing stickers on images [43, 15], to building 3D objects [8]. Because they are more realistic, the perturbations can be larger. On the other hand, large realistic perturbations exacerbate the difficulty of the validity problem since it is even harder to define an "indistinguishable" transformation. For example, a simple in-plane rotation can turn a '6' into a '9'. Similarly, if one is allowed to affix fur to a traffic sign, or repaint it, it will eventually stop being a traffic sign. While most works make an effort to select perturbations indistinguishable from the true positive in some form, this is never quantified. Beyond potentially compromising the significance of these studies, this makes it difficult to compare attacks.

In this work, we avoid these problems by introducing a new attack strategy based entirely on real-world object manipulations. This automatically eliminates the realism problem, since all attacks are based on natural images. We then propose a protocol to *guarantee* the validity of all attacks, by verifying that all perturbations are imperceptible to humans. Finally, we consider a domain (view transformations) that enables the characterization of the *size* of a perturbation. This enables the study of *both* small and large perturbations. We next discuss these contributions in detail.

### 2.3.1 Camera shake and pose manipulations

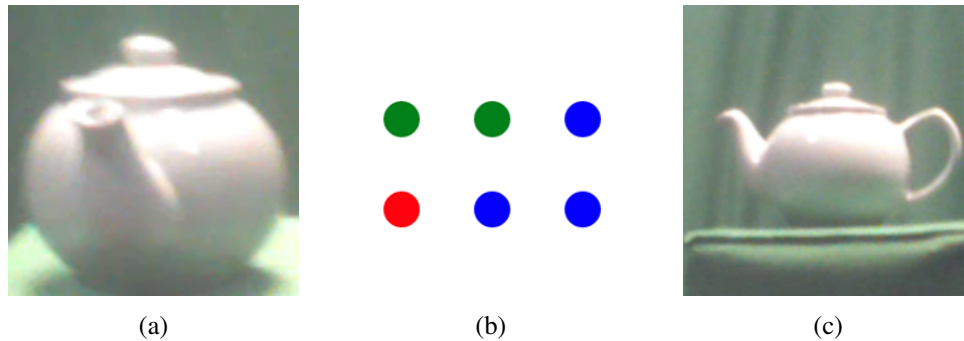
The ultimate goal of this work is to explore the space of attacks that are easy to perform, sometimes even following naturally from the operation of computer vision systems in the real

world, but are difficult to defend. The idea is to exploit image transformations that can be easily performed in the real world, but are hard to replicate by computer. This leverages the fact that, while an attack can be performed with a single example, most attacks can only be defended by training the classifier with many examples. When example collecting is costly, the defense becomes impractical. In this context, digital attacks which use algorithms to produce examples are easier to defend than real world attacks involving image manipulations not replicable by computer. Despite significant advances in photo-realistic rendering, it is still not possible to synthesize truly realistic examples from most object classes, at least without a significant investment in a sophisticated computer graphics infrastructure, rendering experts, etc. Hence, attacks with examples of objects under novel views or novel imaging conditions are difficult to defend. An additional benefit of these attacks is that they make it relatively easy to manipulate perturbation size, which correlates with the degree of view change. We illustrate this by introducing a family of attacks ranging from small transformations due to “camera shake” (CS – small variations of camera position) to larger transformations due to “pose variation” (PV – changes in viewing angle). These attacks are also particularly important because they are trivial to perform. For example, a child can shake a camera or rotate an object. In fact, they are inevitable in certain domains of computer vision, such as robotics, where objects can appear in many 3D orientations and the vision system is subject to nuisances such as shaking due to robot movement.

The first contribution of this work is a dataset to enable replicable studies of CS and PV attacks. For this, we relied on a drone-based imaging setup. A drone was flown around an object, as illustrated in Figure 2.1(a), using markings on the ground to define picture taking stops at regularly spaced intervals. By collecting images at these stops under alignment with the markings on the ground, the drone assembled a set of views of the object corresponding to different orientations of the object in 3D. We refer to these as “object poses”. Examples of multiple poses of an object are shown in Figure 2.1(b). Within each stop, the drone was allowed to hover and collect several images of the object, as shown in Figure 2.1(c). Due to the small hovering motion, many of these images are indistinguishable to the inattentive eye. They show the

**Table 2.1.** Turker imperceptibility rates for true positive (TP), camera shake (CS), pose variation (PV), and different object (DO) pairs.

	IPR (IR)	SIPR (OR)
TP	99.7	99.8
CS	72.4	91.6
PV	7.5	81.5
DO	0.2	1.0



**Figure 2.2.** Turk experiment. (a) True positive (TP)  $x$ , shown for 750 ms. (b) Distractor task: count dots of some color. (c) After correctly counting,  $x'$  shown for 750 ms. Then, turkers asked if the image (object) has changed.

same pose of the same object, varying by very small translations of the camera and some amount of motion blur. We refer to this as “camera shake”. The procedure was repeated for 20 objects per class from 23 different object classes. To facilitate attacks on existing object recognizers, these are classes represented in the ImageNet dataset, where the recognizers are trained. Overall, the dataset contains 30 CS images per pose and 8 poses for 460 objects, totaling 110,400 images. It is split into a *defense dataset* containing 16 objects per class and an *attack dataset* containing 4 objects per class. The defense dataset can be used to learn defenses against the proposed attacks. Each object is furthermore assigned a “frontal” pose by visual inspection, e.g. the frontal pose of the telephone in Figure 2.1(b) is that in the upper left corner. It should be noted that this setup is only necessary to enable *replicable studies* of the proposed attacks and to collect data for *defense purposes*. The attacks themselves can be performed by simply rotating objects.

### 2.3.2 Characterizing indistinguishable perturbations

A difficulty of real world attacks, especially those involving larger perturbations, is to guarantee their validity. After all, under extreme viewing angles, objects can be hard to recognize even for humans. The second contribution of this work is a replicable procedure, based on Amazon Turk experiments, to characterize *indistinguishable perturbations* (IP). We start by proposing that perturbations can only be declared indistinguishable within a certain application context. For object recognition, we identify two contexts of interest. The first is *image retrieval* (IR). This addresses the question of whether a person can distinguish two images. However, we do not pursue the forensic definition of distinguishability commonly used in the literature. Instead, we limit the resources available to the person, by asking them to compare the perturbation to an image they have committed to memory. This is more closely related to object recognition than forensic comparisons.

The setup is illustrated in Figure 2.2. A turker is shown the true positive  $x$  for 750 ms and asked to memorize it. The object disappears and the turker is asked to count the number of dots of some color in a 2x3 grid. This is a distractor task to prevent a purely iconic matching of image details. A second image  $x'$  is finally shown for 750 ms, and the turker is asked to indicate if  $x'$  was the *image* seen earlier. The second image can be of four types: the *true positive*  $x$  (15% of the time), a perturbation of  $x$  due to CS (35%), a perturbation of  $x$  due to a PV (35%), or an image of a *different object* (15%). All images used in this experiment are from the attack dataset of the previous section. From 30 frontal pose images, examples randomly sampled (with replacement) are used as the *true positive* per object. CS perturbations are also “frontal” poses. Given a true positive, an *image pair* is created by sampling one of the 29 remaining frontal poses of the object. This procedure was used to produce 70 CS pairs per object. PV perturbations use images from all object views. The set of perturbed images was created by randomly sampling 10 images from each pose, excluding frontal. Finally, for *different object*, examples were sampled randomly from frontal poses of other objects which belong to different classes. A probability of



error was recorded per type of  $x'$ . These are denoted  $p^{TP}, p^{CS}, p^{PV}$ , and  $p^{DO}$ , respectively. The values of  $p^{CS}$  and  $p^{PV}$  are used as *indistinguishable perturbation rates* (IPR) for CS and PV perturbations. A high IPR implies that turkers are not able to tell apart the perturbed  $x'$  from the true positive  $x$ .  $p^{TP}$  and  $p^{DO}$  are upper and lower bounds for the IPR, respectively. For increased accuracy, each image-perturbation pair was evaluated by 3 turkers. A final quality control threshold was imposed per turker: those who scored above 10% for  $p^{DO}$  and those who did not score at least 90% for  $p^{TP}$  were excluded, as evaluation is trivial in these cases. For the remaining image pairs, those with less than two identical evaluations were eliminated. The IPR was finally determined from the remaining evaluations, by majority vote.

So far, the experiments test if turkers can distinguish perturbed images. This is informative for IR, but ultimately not the goal of object recognition. For example, the rotation of a digit by  $30^\circ$  is a very perceptible image transformation. While most humans can easily tell the image has been rotated, this makes little difference for recognition. An equally large percentage of the population will be able to effortlessly recognize the rotated digit. In other words, recognition is invariant to the perturbation. We argue that, for recognition, it is also important to define the notion of *semantically indistinguishable perturbations* (SIPs). These are perturbations that may be noticeable but do not alter the image semantics. SIPs differ from IPs in that they are tied to the semantics of interest for an application. For example, while a smart car only cares about the presence/absence of pedestrians on the road, a face recognizer aims to determine the person's identity. Hence, replacing the true positive by an image of another person is an SIP for pedestrian detection but not for face recognition. An interesting property of the experimental setup above is that it can be extended to SIPs by simply modifying the *context* of the experiment. This is done by changing the *question* asked to the turkers. Rather than asking them if  $x'$  is the same *image* as  $x$ , they can be asked if it is an image of the same *person, object, animal, scene*, or whatever the semantics of interest are for the application. In this work, we consider the generic *object recognition* (OR) context, asking the turkers if the two images are of the same object. The probabilities  $p^{CS}$  and  $p^{PV}$  then become *semantically indistinguishable perturbation rates* (SIPR) for

CS and PV perturbations. They capture the degree to which the perturbations are imperceptible for OR. Note that a large pose transformation, clearly perceptible for image retrieval can easily be imperceptible for object recognition. This difference is captured by the two questions (“is this the same image?” vs. “is this the same object?”) that set different contexts for the experiment.

In summary, the probabilities  $p^{CS}, p^{PV}$  can be IPRs or SIPRs, depending on the context (IR or OR, respectively). Table 2.1 summarizes the rates observed on the Turk experiments. Several conclusions can be taken from the table. First, turkers’ scores were excellent when spotting replicas of the true positive (IPR  $> 99\%$ ) or rejecting images from different objects (SIPR  $\leq 1\%$ ). This suggests that the experimental protocol is robust. Second, all rates were lower for PV than for CS. This was expected, because PV induces larger image variations. These results confirm the hypothesis that CS is a *small* perturbation, while PV is a *larger* perturbation. Note that only 7% of the PV perturbations were IPs, while this held for 72% of the CS perturbations. Finally, it is clear that indistinguishability depends on context. While only 72% of the CS perturbations were IPs, 92% were SIPs. Similarly, while only 8% of the PV perturbations were IPs, 82% were SIPs.

### 2.3.3 Attacks and defenses

The third contribution of this work is a study of the difficulty of defending attacks based on real-world object manipulations. This is based on the image pairs declared as indistinguishable by the turk experiment <sup>1</sup>. While experiments were performed for both IPs and SIPs, we report on SIPs only, since these are the most relevant perturbations for object recognition. Three datasets were used to implement all defenses: 1) a subset of ImageNet containing all object classes used for attacks, denoted “ImageNet,” 2) a subset of the defense dataset of Section 2.3.1 containing only frontal pose images, denoted “Frontal,” and 3) the entire defense dataset, denoted “All”. Every attack was performed on AlexNet [101], ResNet34 [67], and VGG16 [168].

To evaluate the impact of the different object manipulations, the attacks were implemented with both CS and PV SIPs. For each TP  $x$ , the associated perturbation  $x'$  was fed to the

---

<sup>1</sup>All data collected in this work will be made available publicly upon publications of the paper.

classifier and recognition rates (RR)  $r^{CS}$  and  $r^{PV}$  are recorded. Defenses were evaluated under the "arms race" strategy, by synthesizing examples with different attack methods and retraining the network on a dataset augmented with these examples. We considered methods from the two broad categories discussed above: 1) transformation based and 2) gradient based.

1.
  - **Affine:** Random affine transformations with rotation less than 15 degrees.
  - **Blur:** Gaussian blur kernel with random standard deviation in  $[0,0.6]$ .
  - **Blur-Affine:** Combination of affine and blur.
  - **Worst-of:** The worst-of-K method of [42]. Ten affine transformations are randomly sampled and the one of highest loss is selected.
  - **Color Jitter:** Image saturation and hue transformation according to [77].
2.
  - **FGSM:** The fast gradient sign method of [103].
  - **ENS:** The ensemble adversarial training method of [185].
  - **IFGSM:** The iterative fast gradient sign method of [103].

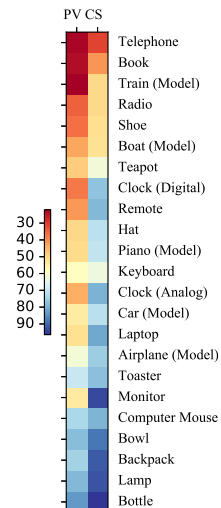
## 2.4 Experiments

**Attack and defense efficiency:** A preliminary observation was that the attacks had similar effect on the three networks. While some models have higher accuracy than others, the relative drop in accuracy due to the attacks were nearly identical. Hence, for brevity, we only discuss average accuracy of the three models here. Table 2.2 presents the RRs of CS and PV manipulation attacks, for networks with various defenses. Each defense was implemented on the three defense datasets and RRs are presented per defense method and dataset. Since all perturbations have been declared SIPs by turkers, the human RR is 1 on these experiments (under the assumption that turkers could correctly classify the TP).

Various conclusions can be drawn. First, as expected, *PV is the more dangerous attack*. For standard ImageNet classifiers the RR drops to almost half (from 70s to 40s), independently of

**Table 2.2.** RRs for CS and PV SIP attacks, under different defense methods and datasets. RRs are averaged over AlexNet, ResNet34 and VGG16.

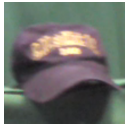
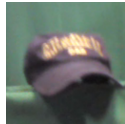














		Attack							
		CS	PV	CS	PV	CS	PV	CS	PV
Defense		ImageNet		Frontal		All		Avg	
	None	73.7	47.2	82.0	63.7	87.1	<b>79.1</b>	80.9	<b>63.3</b>
Transformation	Affine	71.8	45.1	83.4	58.8	85.2	76.5	80.1	60.1
	Blur	74.2	45.2	84.8	<b>64.1</b>	86.9	78.3	82.0	62.5
	Blur-Affine	75.4	47.5	83.5	60.0	<b>88.0</b>	76.6	82.3	61.3
	Worst-of	73.0	47.1	83.8	63.0	86.4	76.1	81.0	62.0
	Color Jitter	74.5	45.5	<b>86.4</b>	61.6	87.1	<b>79.1</b>	<b>82.7</b>	62.0
	Avg	73.8	46.1	84.4	61.5	86.7	77.3	81.6	61.6
Gradient	FGSM	72.9	<b>49.2</b>	84.7	61.1	83.2	74.3	80.3	61.5
	ENS	<b>75.7</b>	46.3	83.6	58.1	81.9	72.8	80.4	59.0
	IFGSM	71.8	47.0	82.8	55.5	83.3	70.0	79.3	57.5
	Avg	73.5	47.5	83.7	58.2	82.8	72.3	80.0	59.3



**Figure 2.3.** Per class RR for CS/PV SIP attacks.

the defense implemented. Second, *no defense method stands out*. While gradient methods achieve best performance for ImageNet training, transformations have superior performance for Frontal and All training. Within each category, relative performance varies with dataset and perturbation type. On average (as seen in the last column of the table), Color Jitter is the top defense against CS. Third, none of the defense algorithms improves significantly on no defense. In fact, *the absence of defense is the best defense against PV*, and close to the best (80.9 vs. 82.7 RR) against CS, on average. Fourth, *data collection is a much more effective defense than algorithms*. Independently of the algorithm, RRs increase significantly from ImageNet to Frontal (10+ points) and increase further from Frontal to All (2 points). However, even the collection of data with CS and PV perturbations fails to fully defend against real-world object manipulations. The best performance against PV (none) has an RR of only 63.3%. For CS the top RR is 82.7%. All these observations support the hypothesis that real-world manipulations are a very effective tool to attack CNNs. Besides trivial to perform, they can be very hard to defend. Since the collection of real data fails to produce a foolproof defense, it is questionable that digital defenses could fully deflate these attacks. Clearly, simple digital defenses, such as Affine or Blur transformations, are ineffective.

**Table 2.3.** Examples of IPs and SIPs, for CS and PV perturbations, that fool many classifiers. In all cases, TP is left, perturbation right. Also shown is ground truth class and # of classifiers fooled .

IPs				SIPs			
CS		PV		CS		PV	
TP: <i>Hat</i>	Fools 16	TP: <i>Bowl</i>	Fools 36	TP: <i>Car</i>	Fools 13	TP: <i>Plane</i>	Fools 20
							
TP: <i>Remote</i>	Fools 21	TP: <i>Hat</i>	Fools 20	TP: <i>Keyb.</i>	Fools 13	TP: <i>Car</i>	Fools 32
							

In-depth comparisons of the table also challenge some common assertions in the adversarial literature. One striking effect is the reversal of performance between gradient and transformation based methods with the defense dataset. Gradient methods work better on ImageNet, but are not effective when CS and PV perturbations are added to the defense set. This supports the hypothesis that they mostly push examples to the edge of the natural image space. Better coverage of these regions, by camera shake and more camera views, eliminates these methods' gains. For example, the average RR of the gradient methods on the All defense dataset is 4 to 7 points weaker than using no defense algorithm at all. Transformation based methods perform significantly better on this dataset. In the adversarial literature, IFGSM and ENS have also been claimed to outperform FGSM. This is because IFGSM generates stronger adversarial examples and ENS decouples the adversarial example generator for the defender (by adding adversarial examples from a third party to the training set). However, this is nearly the opposite of the results on Table 2.2. On average, FGSM outperforms IFGSM and ENS. Again, this is likely due to the real world nature of the attacks. The fact that IFGSM and ENS are better defenses against digital attacks, seems to translate into no benefits for real world attacks.

**Objects:** It is also pertinent to ask which types of objects lead to more successful attacks. Figure 2.3 shows the RR of CS and PV perturbations per object class. While CS leads to higher

RRs for all objects, the RR trend is similar for CS and PV. This suggests that attack efficiency is indeed determined by object properties. Finally, a "lack of symmetry" seems to be the object property most predictive of successful attacks – symmetric objects, such as bottles, lamps, and bowls are less effective attackers than less symmetric objects like telephones, radios, and trains.

**Universal attacks:** A final question is which attacks fool a large number of classifiers. Table 2.3 shows some examples of the most successful perturbations, from this point of view . Some interesting observations can be made. First, perturbations that are clearly noticeable under a forensic comparison (side-by-side images, infinite time) can become indistinguishable under the memory recall paradigm of Figure 2.2. Take the "bowl" and "hat" examples for instance, which were deemed IPs by the turkers; the fact that these perturbations were deemed the same *image* as the TP shows that the standard practice of determining attack validity by forensic comparisons is poorly suited for object recognition. Second, it appears that perturbations of all sizes can fool a large number of models. Note that the perturbations shown range from "insignificant" (almost imperceptible even on a forensic comparison, e.g. "remote") to "large" (significant PVs, e.g. "car" on the bottom right). Overall, it appears that even very elementary natural perturbations can fool state-of-the-art classifiers.

## 2.5 Conclusion

This work makes several contributions to the study of adversarial attacks that are easy to execute but difficult to defend, using a new setup based on real-world object manipulations. Unlike the standard practice in the literature, we considered both small and large perturbations, generated by camera shake and pose variation, and introduced a procedure for systematic collection of such perturbations. This was complemented by a replicable procedure to measure the imperceptibility of perturbations, using Turk experiments. These contributions enabled the creation of a dataset of small and large perturbations, imperceptible under two contexts of interest for object recognition. Experimental results comparing defenses based on many datasets, CNN

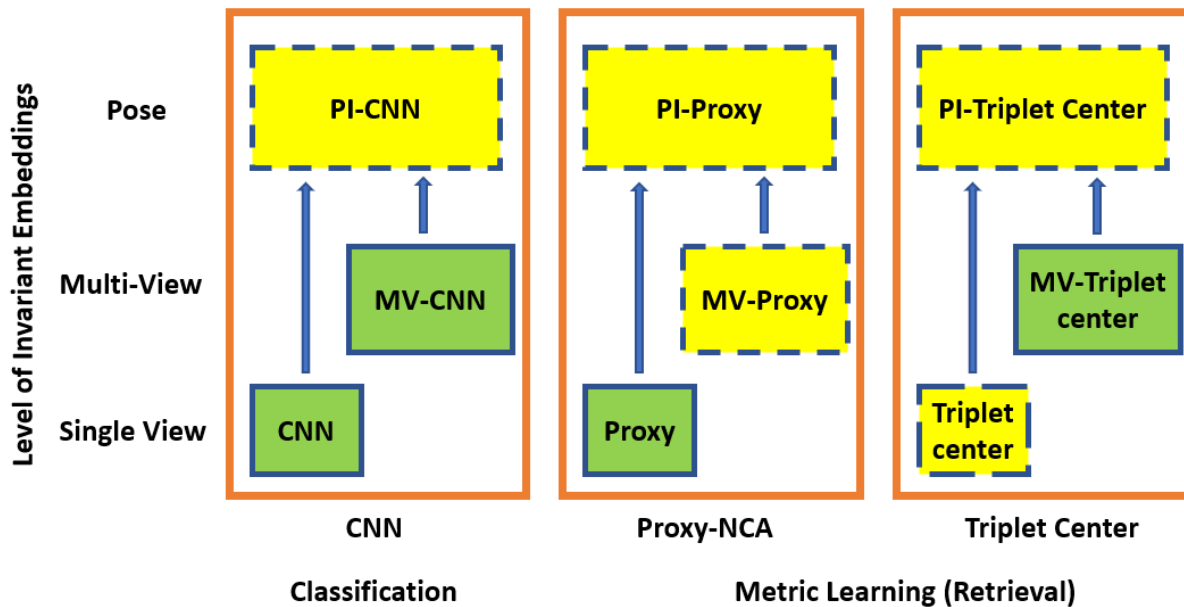
models, and algorithms from the literature elucidated the difficulty of defending these attacks. None of the existing defenses were effective against them, and while better results were achieved with real world data augmentation, this is costly and not foolproof. These results suggest that more research is needed on defenses against "easy to perform" attacks and that the data now assembled can play an important role in this regard.

Chapter 2 is, in full, based on the material as they appear in the publication of "Catastrophic Child's Play: Easy to Perform, Hard to Defend Adversarial Attacks", Chih-Hui Ho\*, Brandon Leung\*, Erik Sandstrom, Yen Chang, and Nuno Vasconcelos, In Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2019. The dissertation author was the primary investigator and author of this paper.

## **Chapter 3**

# **PIEs: Pose Invariant Embeddings**





**Figure 3.1.** Taxonomy of embeddings learned by different methods according to different level of invariance. Green solid boxes represent methods in the literature and yellow dashed boxes represent methods proposed in this work. The proposed pose invariant embedding incorporates single view and multiview invariance and can be applied to different methods, including CNN, proxy-NCA and triplet center. While CNN is designed for classification, the other two aim for metric learning (retrieval task).

### 3.1 Introduction

Convolutional neural networks (CNNs) are frequently used for classification and metric learning, among other tasks. Classification is the central problem of important computer vision applications, such as object and action recognition or detection. Metric learning plays a similar role for image retrieval, face recognition and identification, or zero shot learning. Despite the many different applications, the two tasks are closely related, since they both learn an embedding  $g: \mathcal{X} \rightarrow \mathcal{G}$  that maps images  $\mathbf{x} \in \mathcal{X}$  into features  $g(\mathbf{x}) \in \mathcal{G}$  and are implemented with several CNN layers. Classification aims to produce a discriminant feature space  $\mathcal{F}$ , which separates the different classes, while metric learning aims to produce a feature space  $\mathcal{F}$  with a certain metric structure, where similarity can be captured by some distance function, typically the Euclidean distance.

As shown in the bottom row of Figure 3.1, classification and metric learning have evolved in lockstep. While details of the architecture of  $g(\cdot)$  may favor one or the other, approaches

to the two problems have differed mostly in the subsequent network layers and loss function. Classifiers complement the embedding  $g$  with a softmax layer trained with the logistic loss. Classic metric learning uses no additional layers and a different loss. While several variants have been proposed [27, 61, 170, 141], the most popular is the *triplet* loss of [201, 194, 12, 164]. In practice, however, the differences can be significant. Since triplets raise the dataset size to its cube, metric learning networks are more difficult to train than classifiers. To address this, much of the embedding literature has been devoted to triplet sampling strategies [194, 12, 164, 147, 170], aimed to increase training speed. Recently however, [141] has shown that much faster training is possible by using *proxy* embeddings, which make metric learning a lot more like classification. Inspired by a metric learning approach known as *neighborhood component analysis* (NCA) [53], it adds a layer that resembles a softmax classifier to the embedding and uses the logistic loss for training. A similar generalization of triplet embeddings has been proposed in [68], and denoted as *triplet center* embeddings.

Ideally, an embedding should map all the images of an object collected from multiple views, depths or under different illuminations, into a single point, known as the *object invariant* to these transformations. However, this is hard to achieve on datasets such as ImageNet, which tend to emphasize class diversity and maximize the number of objects imaged per class. They do not provide a dense covering of the transformations (imaged from different camera positions, variable lighting, etc) where an object may be subjected to. Recently, this problem has received significantly more attention, with the introduction of datasets such as ModelNet [206] or ShapeNet [19]. Being datasets of synthetic images rendered from 3D CAD models, these allow the generation of many views of each object labelled for view angle, also known as *object pose*.

The introduction of these multiview synthetic datasets motivated a new wave of algorithms for multiview [175, 92, 45] classification and retrieval, as shown in the middle row of Figure 3.1. These methods have been shown competitive, if not superior, to many methods based on 3D representations, such as voxels [206, 135, 14, 204] or point clouds [50, 213, 211]. This is important because view-based representations can be easily deployed in the real world,

where 3D representations are much more expensive, if not completely infeasible. The most popular architecture for view-based classification is the multiview-CNN (MV-CNN) [175], which complements a standard CNN embedding with a view pooling mechanism that produces a shape descriptor. The shape descriptor is then fed to the softmax layer for classification. Similarly, [68] have introduced the triplet-center loss for multiview metric learning. This is a generalization of the triplet loss and center loss to a multiview level for NCA style metric learning.

While these approaches have been shown to be effective for multiview classification and retrieval, which can be performed easily in the CAD world (e.g ModelNet and ShapeNet), their usefulness for real vision systems is more questionable, for two reasons. First, it is not known how well they work on real images due to the absence of datasets of real images in the wild, with coverage of pose trajectories. While some dense pose datasets exist [13, 92, 51, 145], they are small and tend to depict objects on turntables, without complex backgrounds. Second, and more important, these approaches do not really learn *pose invariant* embeddings. While the shape descriptor is a summary of all the views of the object, the embedding of a single image is not constrained to be similar to this descriptor. In result, these methods tend not to perform well for *single view* recognition or retrieval, where they frequently have weaker performance than standard CNNs. This is important because the multiview setting is not realistic for most real world applications. While multiview training is of interest to enable learning algorithms to capture object variability under various transformations, applications frequently constrain inference to single views. To support the latter, multiview training must produce truly pose invariant embeddings.

In this work, we address these limitations through a combination of contributions. First we perform a review of various approaches in the literature, placing various methods on equal footing and enabling a better understanding of their relative strengths and weaknesses. This results in Figure 3.1, which groups embeddings by their level of invariance. Existing methods are identified by green boxes. It is clear that no truly pose invariant embeddings are available. While view-based embeddings have little invariance, multiview embeddings produce a shape descriptor that represents multiple views, but do not map individual views to this descriptor.

Second, we propose a number of new approaches, showed as yellow boxes in Figure 3.1. Some of these just fill holes in the layers populated by existing methods. For example, MV-Proxy is simple variants of [141] for multiview level and triplet center is variants of [68] for single view level. Other yellow boxes in pose invariant level (top row) are based on new loss functions that encourage embeddings that cluster individual images in the neighborhood of shape descriptors. This makes the shape descriptors truly invariant and enables better performance on single view retrieval and recognition tasks. Finally, we introduce a new multiview dataset for object recognition in the wild. This dataset is composed of objects belonging to ImageNet, and are in all aspects similar to ImageNet images. However, each object is imaged under a set of pre-defined poses, which are provided as additional labels. Similarly to ShapeNet and ModelNet, this enables the learning of pose invariant representations. However, because the images are real, the new dataset enables the testing of invariance in a more realistic setting. Experiments on both the proposed dataset and synthetic datasets show that the proposed pose invariant embedding is more robust to a variable number of views provided for inference.

## 3.2 Related work

Many works have addressed embeddings for classification and retrieval. We review the literature in this section, emphasizing the ideas that are directly relevant to this work.

**Classification:** Given observations and class labels drawn from random variables  $X \in \mathbb{R}^m$  and  $Y \in \{1, \dots, C\}$  the classifier of minimum probability of error is  $y^* = \operatorname{argmax}_y P_{Y|\mathbf{X}}(y|\mathbf{x})$ . A CNN is a model for the posterior probabilities

$$P_{Y|\mathbf{X}}(y|\mathbf{x}) = h_y(\mathbf{x}; \mathbf{W}, \mathbf{b}) = \frac{e^{\mathbf{w}_y^T g(\mathbf{x}) + b_y}}{\sum_{k=1}^C e^{\mathbf{w}_k^T g(\mathbf{x}) + b_k}} \quad (3.1)$$

composed of two stages. The first is an embedding  $g(\mathbf{x}) \in \mathcal{F} \subset \mathbb{R}^d$ , implemented by the layers of the network up to the last one, where  $g$  is a  $d$  dimension feature extractor. Usually,  $g$  consists of a combination of convolutions, pooling, and a ReLU non-linearity. The second is a softmax

layer that resides at the top of the network and computes (3.1) using a layer of weights  $\mathbf{W} \in \mathbb{R}^{d \times C}$  and biases  $\mathbf{b} \in \mathbb{R}^C$ . To minimize notational clutter, we will omit the bias vector in many of the expressions below. This follows the common practice of absorbing it in  $\mathbf{W}$  and using homogeneous coordinates. CNNs are trained by cross-entropy minimization. Given a dataset  $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^n$  this consists of finding  $\mathbf{W}$  and the parameters of  $g$  that minimize the risk

$$\mathcal{R}(\mathcal{D}) = \sum_i L(\mathbf{x}_i, y_i), \quad (3.2)$$

defined by the logistic loss  $L(\mathbf{x}, y) = -\log h_y(\mathbf{x}; \mathbf{W})$ .

**Metric learning:** Metric learning aims to endow the feature space  $\mathcal{F}$  with a metric, usually the Euclidean distance

$$d(g(\mathbf{x}), g(\mathbf{y})) = \|g(\mathbf{x}) - g(\mathbf{y})\|^2, \quad (3.3)$$

so as to allow the geometric implementation of operations like classification, e.g. using nearest neighbors. While many losses have been proposed [27, 61, 170], this is usually done with a loss function that operates on example triplets, pulling together (pushing apart) similar (dissimilar) examples [201, 194, 12, 164]. Given an anchor  $\mathbf{x}$ , a similar  $\mathbf{x}^+$  and a dissimilar example  $\mathbf{x}^-$ , the triplet loss is defined as

$$L(\mathbf{x}, \mathbf{x}^+, \mathbf{x}^-) = \phi(d(g(\mathbf{x}), g(\mathbf{x}^-)) - d(g(\mathbf{x}), g(\mathbf{x}^+))), \quad (3.4)$$

where  $\phi(\cdot)$  is a margin loss, e.g. the hinge loss  $\phi(v) = \max(0, m - v)$  or the logistic loss  $\phi(v) = \log(1 + e^{-v})$ . In general, similar and dissimilar examples are determined by the class labels of  $\mathcal{D}$ . We refer to these methods as *triplet* embeddings.

Modern CNNs are learned by stochastic gradient descent (SGD), processing the data in batches of relatively small size, e.g.  $b=32$ . On a dataset of size  $n$  there are  $O(n)$  examples and  $O(n^3)$  triplets. Similarly, there are  $O(b)$  examples and  $O(b^3)$  triplets in a batch. Hence, while the

number of batches needed to cover the dataset is  $O(n/b)$  for examples, it becomes  $O((n/b)^3)$  for triplets [141]. Since  $n/b$  is in the tens of thousands, triplet learning is cubically more complex than example-based learning. While many sampling strategies have been proposed to address this problem [164, 194, 147, 178], metric learning methods are substantially harder to use and slower to converge than classification methods.

Recently, [141] has shown that this problem can be overcome using a loss function inspired by neighborhood component analysis (NCA) [53]. This consists of defining a *proxy*  $\mathbf{p}_y$  per class, adding a softmax-like layer

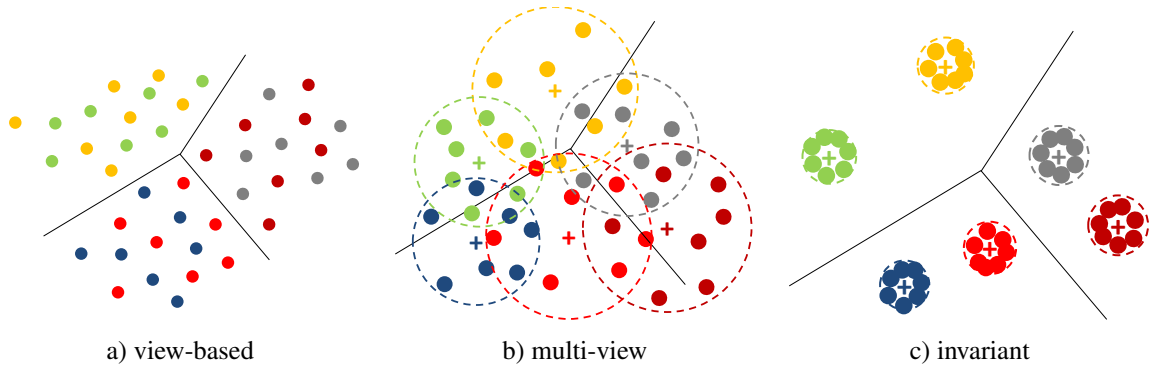
$$s_y(\mathbf{x}; \mathbf{P}) = \frac{e^{-d(g(\mathbf{x}), \mathbf{p}_y)}}{\sum_{k \neq y} e^{-d(g(\mathbf{x}), \mathbf{p}_k)}}, \quad (3.5)$$

where  $\mathbf{P}$  is the matrix of proxies  $\mathbf{p}_k$ , and learning both  $\mathbf{P}$  and  $g(\mathbf{x})$  by minimizing the risk of (4.2) with the logistic loss  $L(\mathbf{x}, y) = -\log s_y(\mathbf{x}; \mathbf{P})$ . We refer to this method as *proxy* embedding.

**Multiview classification:** In multiview classification, each observation consists of a set of  $V$  views  $\mathbf{X} = \{\mathbf{x}_k\}_{k=1}^V$  and parameters are learned from a multiview dataset  $\mathcal{D}_m = \{(\mathbf{X}_i, y_i)\}_{i=1}^n = \{(\mathbf{x}_{i1}, \dots, \mathbf{x}_{iV}, y_i)\}_{i=1}^n$ . The goal is to jointly classify all these views. A popular approach is the multiview-CNN (MV-CNN) [175], which implements two embeddings. Each individual image  $x_k$ , where  $x_k$  is imaged at  $k^{\text{th}}$  predefined viewpoint, is processed by a shared feature extractor  $g$  and all the resulting view descriptors  $g(x_k)$  is then averaged to produce a *shape* descriptor

$$g_m(\mathbf{X}) = \frac{1}{V} \sum_{k=1}^V g(\mathbf{x}_k), \quad (3.6)$$

where subscript  $m$  denotes multiview. The embedding parameters are learned from a multiview dataset  $\mathcal{D}_m$  by using  $g_m$  with softmax layer (3.1), the risk of (4.2), and the logistic loss. Several variants of this approach have been proposed, either making specific architectural enhancements to the embedding  $g$  [190, 155], or using weighted versions of (3.6) [45]. Similar enhancements are possible for all methods discussed in this work.



**Figure 3.2.** Embeddings produced by methods at the three levels of invariance of Figure 3.1. In all plots, there are three classes, two objects per class, and each dot represents the embedding of an image. Dots of the same color correspond to different views of the same object. In b) and c), a '+' is used to denote the shape descriptor and a dashed circle to denote the distribution of views of the associated object. Only the invariant embedding of c) guarantees a good clustering of both shape descriptors per class and individual views per object.

**Multiview metric learning:** Substantially less work has been devoted to multiview metric learning. [68] combined the MV-CNN embedding with the proxy-based idea of [141], but applied to the triplet loss. They denote proxies as *centers* and define the multiview triplet-center loss

$$L(\mathbf{X}, y, \mathbf{P}) = \phi \left( \min_{j \neq y} d(g_m(\mathbf{X}), \mathbf{p}_j) - d(g_m(\mathbf{X}), \mathbf{p}_y) \right) \quad (3.7)$$

where  $\mathbf{P}$  is the matrix of centers  $\mathbf{p}_j$  and  $g_m$  is defined as in (3.6). We refer to this method as *multiview triplet center (MV-TC)* embedding.

### 3.3 Bringing object invariants to the real world

In this section, we discuss a number of contributions that follow from the above review.

#### 3.3.1 New view-based and mutiview embeddings

Figure 3.1 provides a functional organization of embeddings for classification and metric learning. The bottom two rows summarize the state of the literature, with green boxes identifying the approaches that have been proposed. They group these methods according to whether they embed single or multiple views. One immediate contribution is that there are a number of

“missing” approaches (e.g. multiview proxy and single view triplet center). We propose to fill the gaps, introducing several new embeddings, which are extensions of those available: the *triplet center* embedding is the view-based equivalent of the multiview triplet center embedding[68], replacing multiview triplet-center loss (3.7) with single view

$$L(\mathbf{x}, y, \mathbf{P}) = \phi \left( \min_{j \neq y} d(g(\mathbf{x}), \mathbf{p}_j) - d(g(\mathbf{x}), \mathbf{p}_y) \right), \quad (3.8)$$

and the *MV-proxy* generalizes the single view proxy embedding (3.5) to multiview

$$s_y^m(\mathbf{X}; \mathbf{P}) = \frac{e^{-d(g_m(\mathbf{x}), \mathbf{p}_y)}}{\sum_{k \neq y} e^{-d(g_m(\mathbf{x}), \mathbf{p}_k)}}, \quad (3.9)$$

where superscript  $m$  denotes multiview.

### 3.3.2 The need for invariant embeddings

A second, and practically more important, contribution of Figure 3.1 is to show that no attention has been given to the design of truly *invariant* embeddings. This is important for many real-world systems, where one would like to leverage multiview data for training but perform classification or retrieval on single views. In general, it is not realistic to expect that multiple views of an object will be available at classification or retrieval time. We refer to this problem as *pose invariant classification and retrieval*. Figure 3.2 illustrates the limitations of existing approaches to address this problem.

View-based embeddings do not leverage multiple object views during training, treating all views of all objects in the same class equally. In result, as illustrated in Figure 3.2 a), there is no guarantee that these embeddings will cluster views from same object. While clustering views into classes, they are free to intertwine the views of different objects in the same class. On the other hand, multiview embeddings (3.6) only constrain the shape descriptor, i.e. the *average* of single view embedding. As illustrated by Figure 3.2 b), where shape descriptors are denoted by



a '+' , this suffices to produce a good shape descriptor clustering. However, it does not guarantee a good clustering of all individual views from an object. Note that the shape descriptors are all correctly classified, but this is not the case for the individual views, which can spread across class boundaries. This is illustrated by the dashed circles, which identify the distribution of images of each object. Due to this problem, multiview approaches tend to underperform the single view embeddings of a) for single view classification and retrieval [92, 45].

In order to address these problem, a new form of embeddings is needed. Figure 3.2 c) shows the behavior desired for a truly invariant embedding, which should be both single view invariant and multiview invariant. We denote this new form of embedding as *pose invariant embedding (PIE)*. PIE guarantees two properties: that 1) single view embeddings (image descriptors) of an object are clustered around multiview embedding (shape descriptor) and 2) multiview embedding is clustered around the descriptor of its labeled class.

To guarantee the two properties, we return to the probabilistic formulation and introduce an intermediate object variable  $O$ , leading to

$$\begin{aligned}
 P_{Y|\mathbf{X}}(y|\mathbf{x}) &= \sum_n P_{Y|O,\mathbf{X}}(y|n,\mathbf{x})P_{O|\mathbf{X}}(n|\mathbf{x}) \\
 &= \sum_n P_{Y|O}(y|n)P_{O|\mathbf{X}}(n|\mathbf{x})
 \end{aligned}
 \tag{3.10}$$

where we have used the fact that once the object is known the class is independent of the view. This provides a decomposition of the posterior probabilities into an object-to-class  $P_{Y|O}(y|n)$  and a view-to-object  $P_{O|\mathbf{X}}(n|\mathbf{x})$  model. This decomposition can be exploited to enforce the two properties above. We next discuss how to do this for the various approaches of Figure 3.1.

### 3.3.3 Pose invariant proxy embedding

We start by extending the proxy embedding [141] of (3.5) with the conditional probabilities of (3.10). We then note that the multiview form of proxy embedding, given by (3.9), is an object-to-class model, if the shape descriptors is produced by averaging image descriptors

associated with the same object (3.6). Hence, the object-to-class model can be identical to the multiview proxy embedding (3.9)

$$P_{Y|O}(y|n) = s_y^m(\mathbf{X}_n; \mathbf{P}). \quad (3.11)$$

The view-to-object model should be similar to single view proxy (3.5) but use a set of object proxies. To encourages the clustering of Figure 3.2 c), we propose adopting the shape descriptor produced by (3.6) as the proxy for the associated object. This leads to the model

$$P_{O|\mathbf{X}}(n|\mathbf{x}) = \frac{e^{-d(g(\mathbf{x}), g_m(\mathbf{X}_n))}}{\sum_{j \neq n} e^{-d(g(\mathbf{x}), g_m(\mathbf{X}_j))}}. \quad (3.12)$$

*Pose invariant proxy* (PI-Proxy) embedding can then be derived by combining the two models with conditional probability (3.10). The approximated probabilities in [141] is then used and we have

$$s_y^{inv}(\mathbf{x}, \mathbf{P}) = \frac{\sum_n e^{-d^{inv}(\mathbf{x}, \mathbf{X}_n, \mathbf{p}_y)}}{\sum_{i \neq y, n} e^{-d^{inv}(\mathbf{x}, \mathbf{X}_n, \mathbf{p}_i)}}, \quad (3.13)$$

where

$$d^{inv}(\mathbf{x}, \mathbf{X}_n, \mathbf{p}_y) = \alpha d(g(\mathbf{x}), g_m(\mathbf{X}_n)) + \beta d(g_m(\mathbf{X}_n), \mathbf{p}_y) \quad (3.14)$$

is denoted as the *pose invariant* distance.  $\alpha, \beta$  are two hyperparameters that enable control over the contribution of the two components of the distance. Note that the feature extractor  $g$  is exactly the same as in the MV-CNN, i.e. there is no additional parameters and no change in the network.

### 3.3.4 Properties of pose invariant distance

The pose invariant distance of (3.14) has several properties of interest. First, setting  $\alpha = 0$  and  $\beta = 1$  results in the distance of the MV-proxy embedding (3.9), which leads to Figure 3.2

b). Second, for  $\alpha = \beta = 1$ , it becomes Figure 3.2 c) and follows from the triangle inequality that

$$\begin{aligned} d^{inv}(\mathbf{x}, \mathbf{X}_n, \mathbf{p}_y) &= d(g(\mathbf{x}), g_m(\mathbf{X}_n)) + d(g_m(\mathbf{X}_n), \mathbf{p}_y) \\ &\geq d(g(\mathbf{x}), \mathbf{p}_y), \end{aligned} \quad (3.15)$$

i.e. the invariant distance is an upper bound on the distance of the single view proxy. While the  $\alpha$  term encourages clustering of individual views around the object (shape descriptor), the  $\beta$  term encourages clustering of objects into object class. Hence, the PI-Proxy embedding offers a range of solutions between the behaviors of Figure 3.2 b) and c).

### 3.3.5 Generating pose invariant embeddings

The procedure above can be generalized to all approaches of Figure 3.1 that use proxies. This is also true for classifiers, where the weights  $\mathbf{w}_y$  of (3.1) play the role of proxies. The procedure for producing a pose invariant model is as follows.

1. use the multiview model as object-to-class model  $P_{Y|O}(y|n)$ .
2. use the view-based model as view-to-object model  $P_{O|\mathbf{X}}(n|\mathbf{x})$ .
3. replace the proxies of  $P_{O|\mathbf{X}}(n|\mathbf{x})$  by the shape descriptors of (3.6). Use the shape descriptor of object  $O$  as proxy for this object.
4. use the conditional probability (3.10) to combine the two models into a pose invariant model.

Applying this procedure to the CNN of (3.1) leads to the *pose invariant-CNN* (PI-CNN)

$$h_y^{inv}(\mathbf{x}, y; \mathbf{W}) = \frac{\sum_n e^{d^{inv}(\mathbf{x}, \mathbf{X}_n, \mathbf{p}_y)}}{\sum_{n,j} e^{d^{inv}(\mathbf{x}, \mathbf{X}_n, \mathbf{p}_j)}}, \quad (3.16)$$

where  $d^{inv}(\mathbf{x}, \mathbf{X}_n, \mathbf{p}_y)$  is defined as in (3.14). This is identical to the MV-CNN when  $(\alpha, \beta) = (0, 1)$ . For larger  $\alpha$ , the classifier also discriminates between objects in the same class, assigning each

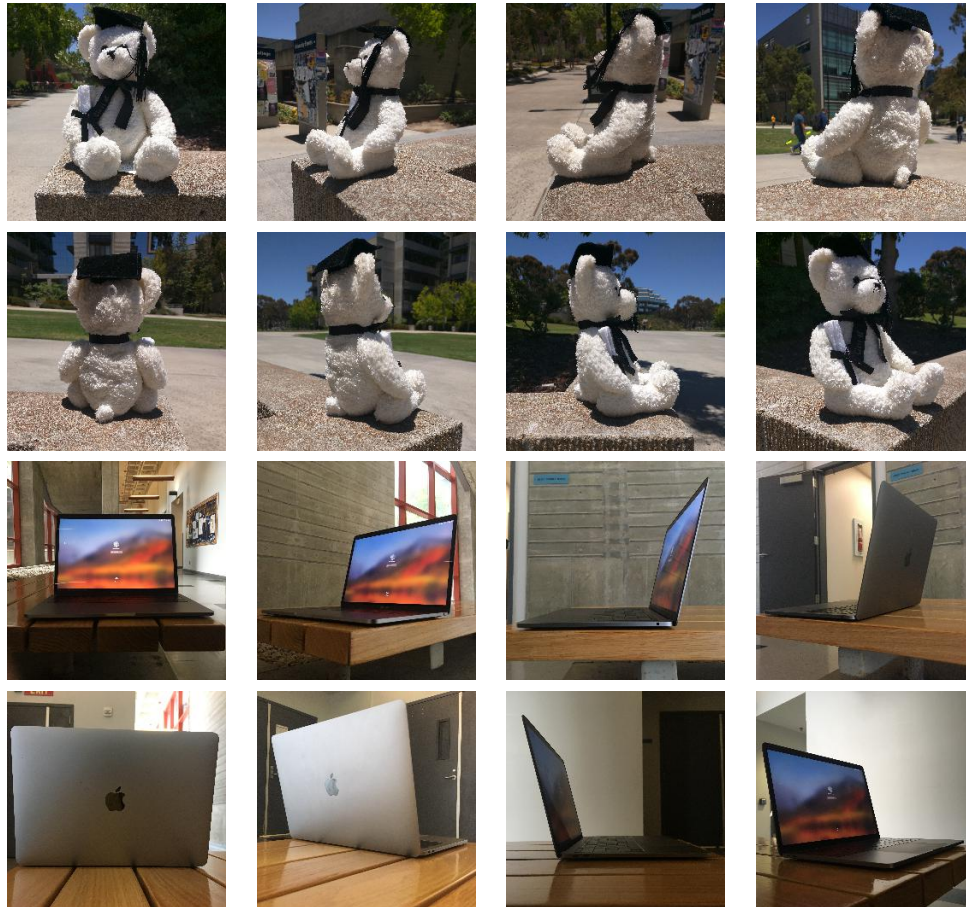
view to the corresponding object descriptor. only assigns views to objects.

Applying the procedure to the triplet center approach, leads to the *pose invariant triplet center* (PI-TC) embedding. This combines the multiview triplet center distance of (3.7) and the triplet center loss of (3.8), using shape descriptor as centers, leading to the loss function

$$\begin{aligned}
L(\mathbf{x}, y, \mathbf{P}) &= \\
&= \phi(\alpha(\min_{k \neq n} d(\mathbf{x}, \mathbf{X}_k) - d(\mathbf{x}, \mathbf{X}_n)) \\
&+ \beta(\min_{i \neq y} d(\mathbf{X}_n, \mathbf{p}_i) - d(\mathbf{X}_n, \mathbf{p}_y))) \tag{3.17}
\end{aligned}$$

### 3.3.6 Learning and inference

The models of (3.13), (3.16), and (3.17) are all functions of the view and multiview embeddings,  $g$  and  $g_m$ . However, since view feature extractor  $g$  is shared by all views and  $g_m$  is the average over view features given by (3.6), the total number of parameters is equal to that of a single CNN. In this aspect, all invariant embeddings of Figure 3.1 have the same complexity. Training boils down to learning the parameters of CNN, using (3.13), (3.16), and the logistic loss or (3.17) in risk  $\mathcal{R}$  (4.2). This is a standard backpropagation learning problem. For inference, several modes are possible. In the multiview mode, only the model  $P_{Y|O}(y|\mathbf{X})$  is used. This is equivalent to using the multiview methods in the second row of Figure 3.1, i.e. MV-CNN, MV-proxy (3.9), and MV-triplet center (3.7). However, these models can still benefit from invariant training. For pose invariant recognition and classification, the models are those of (3.13), (3.16), and (3.17). In the case where a single view  $\mathbf{x}$  is available at inference time, i.e.  $\mathbf{o}_n = g_m(\mathbf{X}_n)$  is not available, all expressions can be simplified. For example, the PI-CNN reduces to  $h_y^{inv}(\mathbf{x}, y) = \frac{e^{-d(\mathbf{x}, \mathbf{p}_y)}}{\sum_j e^{-d(\mathbf{x}, \mathbf{p}_j)}}$ . If partial views are available at inference time, the multiview mode is again used, but (3.6) is reformulated as  $g_m(\mathbf{X}) = \frac{1}{V'} \sum_{k=1}^{V'} g(\mathbf{x}_k)$ , where  $V'$  is the number of views available.



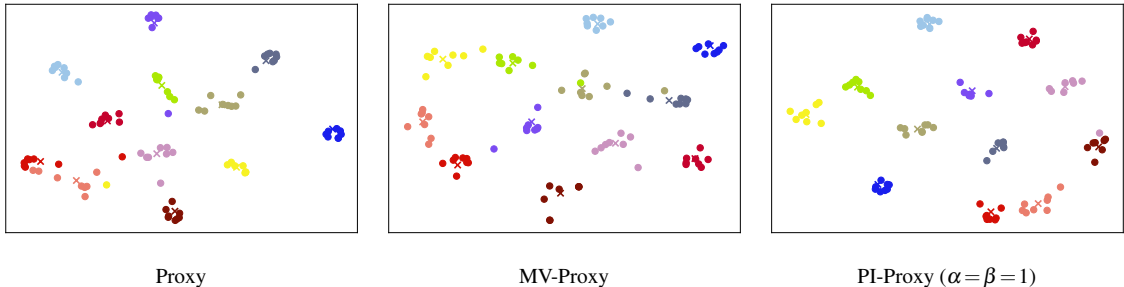
**Figure 3.3.** Examples of the 8 viewpoints of ObjectPI, for 2 objects.

### 3.4 Pose invariance dataset

Existing multiview object datasets can be grouped in two classes. The first includes synthetic datasets such as ModelNet [206] or ShapeNet [19]. These are large and popular, but only depict computer graphics rendered objects. The second includes “turntable datasets”, i.e. datasets imaged in the lab, by collecting images placed on a turntable, as it is rotated [13, 51, 145]. These are more realistic, but still lack natural backgrounds. In this work, we introduce a new dataset that addresses these limitations. It consists of images collected in the wild, by placing each object in a scene and taking pictures with a camera, which is moved around the object. An example of the views collected for an object is shown in Figure 3.3. The dataset contains 8 views per object, for 500 objects from 25 classes. These classes are chosen from ImageNet,

**Table 3.1.** Proxy based methods on ObjectPI.  $\alpha = \beta = 1$  for PI-Proxy.

Task		Proxy	MV-Proxy	PI-Proxy
Class.	Single	68.5	63.2	<b>68.7</b>
	Multi	78.8	78.3	<b>80.0</b>
(Acc.)	Avg	73.7	70.7	<b>74.4</b>
Object		47.7	49.3	<b>49.4</b>
Retr.	Single	59.7	57.9	<b>62.6</b>
	Multi	76.8	74.7	<b>78.2</b>
(mAP)	Avg	61.4	60.6	<b>63.4</b>



**Figure 3.4.** TSNE visualization of proxy based embeddings on ObjectPI. Each dot is an object view, objects are identified by color, and their shape descriptors by 'x's.

to enable the use of CNNs pre-trained on the latter. The dataset is split into a training and test set, containing 16 and 4 objects respectively for each class. We refer to the dataset as the *object pose invariance* (ObjectPI)<sup>1</sup> dataset.

## 3.5 Experiments

In this section, we report on an experimental evaluation of the methods of Figure 3.1 on 5 different tasks, covering classification and retrieval at different levels of invariance.

### 3.5.1 Experimental setup

**Dataset:** All experiments are based on three datasets. **ModelNet40** [206] is a 3D CAD dataset, of 40 object classes and 3183 objects. We use the training and testing splits of [175, 68], with 80 training and 20 test objects. For each object, 12 views are rendered uniformly (viewpoint interval 30 degree), identical to [175] and case (i) of [92]. Note that all reported results are for

<sup>1</sup>All data collected in this work will be made available publicly.

instance accuracy. **MIRO** [92] is a dataset of real world objects. Each object is imaged from 10 elevations and 16 azimuths, to produce 160 images. We use the 16 images of  $0^\circ$  elevation. **ObjectPI** is described in Section 3.4.

**Tasks:** All embeddings are tested on retrieval and classification and trained with all object views. Both single and multi-view inference are considered. **Classification:** For CNN based methods, class is determined by the probabilities generated by the network, while for proxy and triplet center (TC) based methods, a nearest neighbor classifier is used. Classification accuracy is reported. **Single view classification** predicts the class of one image. **Multiview classification** predicts the class of a set of object views. For a CNN, this is done by averaging class probabilities over all views. For proxy and triplet center methods, a nearest neighbor classifier compares the shape descriptors extracted from the set of views to the class descriptors obtained from the training set. **Retrieval:** Retrieval results are reported in terms of mean average precision (mAP). Three retrieval tasks are considered. **Single view retrieval** aims to retrieve images in the class of a query view. **Object retrieval** aims to retrieve other views of the object in the query view. These methods compare view descriptors. **Multiview retrieval** compares shape descriptors, aiming to retrieve the objects in the same class of the object used to generate a set of query views.

**Implementation:** All experiments use a VGG16 [167] model implemented on Pytorch. For MV approaches, view pooling is performed before the softmax function. Learning rate is  $1e-5$  and Adam[98] optimizer is used in all experiments.

### 3.5.2 Joint classification and retrieval

The development of representations for joint classification and retrieval has shown to be difficult. Most methods specialize on one of the tasks, to the point that the papers do not even present results for the other. For example, [92] only addresses classification, while [68] is mainly designed for retrieval. The few works that report both classification and retrieval results use additional steps to prop at least one of the tasks. For example, [45, 175] train an additional low rank Mahalanobis metric to boost retrieval performance. In addition, only few methods report

single image retrieval and classification result on classifier trained with multiview. It is simply accepted that view based embeddings have better performance for view classification and retrieval, while multiview embeddings are better for multiview classification and retrieval. It has so far not been shown that a single embedding can perform well on both tasks for both single and multiview.

**Visualization:** To study this issue in more detail, we consider the proxy based approaches of Figure 3.1, namely Proxy, MV-Proxy, and PI-Proxy. We start by visualizing, in Figure 3.4, the embeddings produced by the three approaches, using TSNE [187]. To simplify the plots, only 12 classes and 1 object per class are shown. Objects are identified by dots of the same color, which correspond to individual views. The shape descriptor of (3.6) is also shown as an 'x'. The classes and objects used in the visualization were chosen randomly. This plot confirms the predictions of Figure 3.2. While all methods succeed at separating the shape descriptors, the placement of individual views is very different. For Proxy and MV-Proxy, these may be embedded far away from the shape feature. MV-Proxy, which only optimizes the shape embedding (ignores the placement of views) produces the most scattered distribution. Proxy methods have more clustered embeddings, but the clustering is significantly inferior to that of PI-Proxy. In this case, most views cluster around the shape embedding produce object clusters of very small overlap. This is a direct consequence of the use of the pose invariant distance of (3.14).

**Classification & Retrieval:** Table 3.1 shows that PI-Proxy achieves the best performance of the three methods on all retrieval and classification tasks. While this is not surprising, given the clusterings of Figure 3.4, the differences can be quite significant, depending on the the task. Note that MV-Proxy is particularly poor for single view classification. This is explained by the poor view clustering and is a well known limitation of multiview methods [45]. Proxy, is competitive with PI-Proxy on image classification, but inferior (2-3% points weaker) on the other tasks.

**Robustness to number of views:** Although multiview training improves classification accuracy [175], the latter often decreases dramatically for single view inference [45, 92]. In this setting multiview CNNs frequently underperform a standard single view classifier. This is unlike



**Table 3.2.** Comparison with state of the art methods on ModelNet dataset for 5 different tasks on VGG16.

Method	ModelNet (12 views)						
	Classification (Accuracy %)			Retrieval (mAP %)			
	Single	Multi	Avg.	Object	Single	Multi	Avg.
RN[92]	80.2	89.0	84.6	22.6	20.2	63.9	35.6
MV-CNN[175]	71.0	87.9	79.4	29.6	41.7	71.5	47.6
PI-CNN	<b>85.4</b>	88.0	86.7	<b>50.8</b>	77.5	81.8	<b>70.0</b>
MV-TC[68]	77.3	88.9	83.1	36.6	63.5	84.0	61.4
PI-TC	81.2	88.9	85.1	41.4	71.5	84.2	65.7
MV-Proxy	79.7	<b>89.6</b>	84.7	35.0	66.1	<b>85.1</b>	62.1
PI-Proxy	85.1	88.7	<b>86.9</b>	40.6	<b>79.9</b>	<b>85.1</b>	68.6

**Table 3.3.** Comparison with state of the art methods on MIRO dataset for 5 different tasks on VGG16.

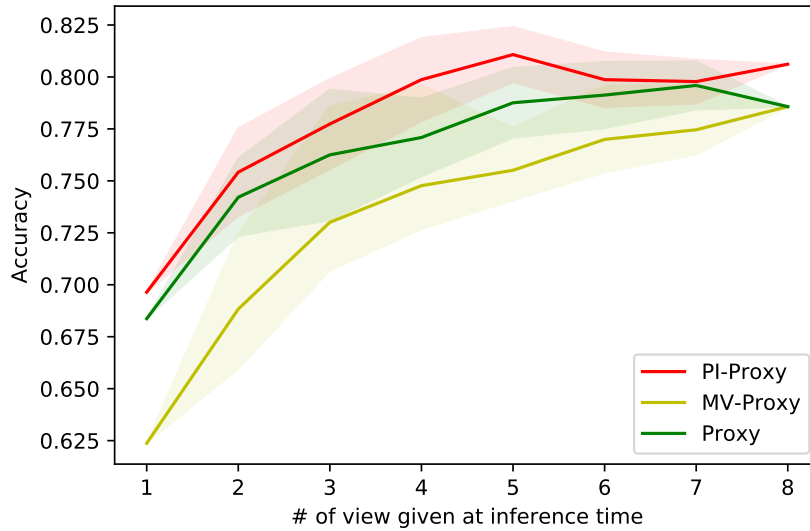
Method	MIRO (16 views)					
	Classification (Accuracy %)			Retrieval (mAP %)		
	Single	Multi	Avg.	Object	Single	Avg.
RN[92]	93.2	<b>100</b>	96.6	33.0	33.0	33.0
MV-CNN[175]	<b>100</b>	<b>100</b>	<b>100</b>	92.0	92.0	92.0
PI-CNN	<b>100</b>	<b>100</b>	<b>100</b>	<b>100</b>	<b>100</b>	<b>100</b>
MV-TC[68]	<b>100</b>	<b>100</b>	<b>100</b>	99.8	99.8	99.8
PI-TC	<b>100</b>	<b>100</b>	<b>100</b>	<b>100</b>	<b>100</b>	<b>100</b>
MV-Proxy	<b>100</b>	<b>100</b>	<b>100</b>	99.8	99.8	99.8
PI-Proxy	<b>100</b>	<b>100</b>	<b>100</b>	<b>100</b>	<b>100</b>	<b>100</b>

**Table 3.4.** Comparison with state of the art methods on ObjectPI for 5 different tasks on VGG16.

Method	ObjectPI (8 views)						
	Classification (Accuracy %)			Retrieval (mAP %)			
	Single	Multi	Avg.	Object	Single	Multi	Avg.
RN[92]	37.5	63.2	50.3	40.1	25.2	41.9	35.7
MV-CNN[175]	62.1	74.1	68.1	42.6	53.8	72.3	56.2
PI-CNN	66.5	76.5	71.5	60.7	58.9	72.1	63.9
MV-TC[68]	65.7	79.2	72.4	51.8	59.5	77.3	62.9
PI-TC	<b>69.3</b>	77.5	73.2	<b>61.8</b>	<b>63.8</b>	76.7	<b>67.4</b>
MV-Proxy	63.2	78.3	70.7	49.3	57.9	74.7	60.6
PI-Proxy	68.7	<b>80.0</b>	<b>74.4</b>	49.4	62.6	<b>78.2</b>	63.4

the proposed pose invariant embeddings, as shown in Figure 3.5.

The PI-Proxy embedding has performance comparable to that of MV-Proxy for multiple



**Figure 3.5.** Classification accuracy of proxy based embedding on ObjectPI as a function of number of views at inference time.

views, but much superior performance as the number of views decreases. This is again justified by the improved view clustering of Figure 3.4.

### 3.5.3 Comparison to the state of the art

We next performed a comparison of all embeddings of Figure 3.1 to other methods in the literature, on ModelNet, MIRO and ObjectPI datasets. Since most previous work has been done on ModelNet, we used the results on this dataset as guidance to select some state of the art models. It should be said that this is not easy, because the existing methods vary along many dimensions. This includes the use of different backbone network architectures (e.g. VGG-M instead of the more popular VGG16 that we adopt), architectural enhancements (e.g. view pooling layers that implement operations different from averaging view descriptor (3.6)) and complementary steps (e.g. optimizing the distance metric used for retrieval after the embedding is learned). All these variations are orthogonal to the invariance issue studied in this work, and could be applied to any of the embeddings of Figure 3.1.

Furthermore, most existing methods only report results for few, sometimes even only one, of the 5 tasks that we consider. This allows for the detailed optimization of the embeddings

for these tasks. Such optimization is not feasible under the experimental protocol now proposed, given the need to compare many embeddings on the 5 tasks and the goal of identifying embeddings that perform well across the 5 tasks. We believe that this is a set-up of greater practical significance, which future works in this area should adopt. Nevertheless, we used existing results to identify two state of the art models on ModelNet: the RotationNet (RN) [92] for classification and the triplet-center of [68] for retrieval. The later is what we denote by MV-triplet center (MV-TC) in Figure 3.1. For fair comparison, we re-trained these models under our set-up and tested them on the 5 tasks and 3 datasets that we now consider. For example, RN is re-trained with VGG16 instead of AlexNet . We also present results for the other existing methods of Figure 3.1, namely the MV-CNN [175] and the proxy embedding of [141].

Table 3.2- 3.4 summarizes the results of multiview and PIE based methods on the three datasets. The best result of each task is marked in bold and shadow denotes that the result of pose invariant based method is better or comparable than that of multiview based. Several conclusions can be drawn. First, pose invariant embeddings (PIEs) are clearly more robust than multiview embeddings (MVEs) on both classification and retrieval tasks. Among the 60 results listed in the table, PIEs outperformed MVEs on 46. In some cases, the difference was drastic. For example, for single view classification on ModelNet, the PI-CNN achieved 85.4% accuracy, outperforming the MVCNN by 14%. Second, one possibility to compare the performance of the different PIEs is to count the number of boldfaced entries. These indicate the number of "wins," i.e. how many times the method had equal or better performance than all others. Under this metric PI-proxy (12 wins) had slightly better performance, followed by PI-TC (10 wins), and PI-CNN (9 wins). However, the difference was not very significant. This shows that adding PIEs increases robustness regardless the approaches being used in the multiview level. Third, regarding classification vs. retrieval, the methods behave somewhat differently. While PI-Proxy achieved the best classification results on all datasets, PI-CNN had the best retrieval results in ModelNet and PI-TC on ObjectPI. However, the results of the three PIEs were close in most cases. Again, the most significant observation is how this differs from the behavior of the

embeddings in the literature. For example, the RotationNet(RN) is competitive for classification but has very weak retrieval performance. Fourth, regarding datasets, best results were obtained on MIRO, then ModelNet, with ObjectPI posing the greatest challenge to most embeddings. This is not totally surprising, since MIRO and ModelNet have no backgrounds, MIRO is a relatively small dataset (120 objects), and ModelNet has no object textures. Nevertheless, these results confirm the need for a more realistic dataset, such as ObjectPI.

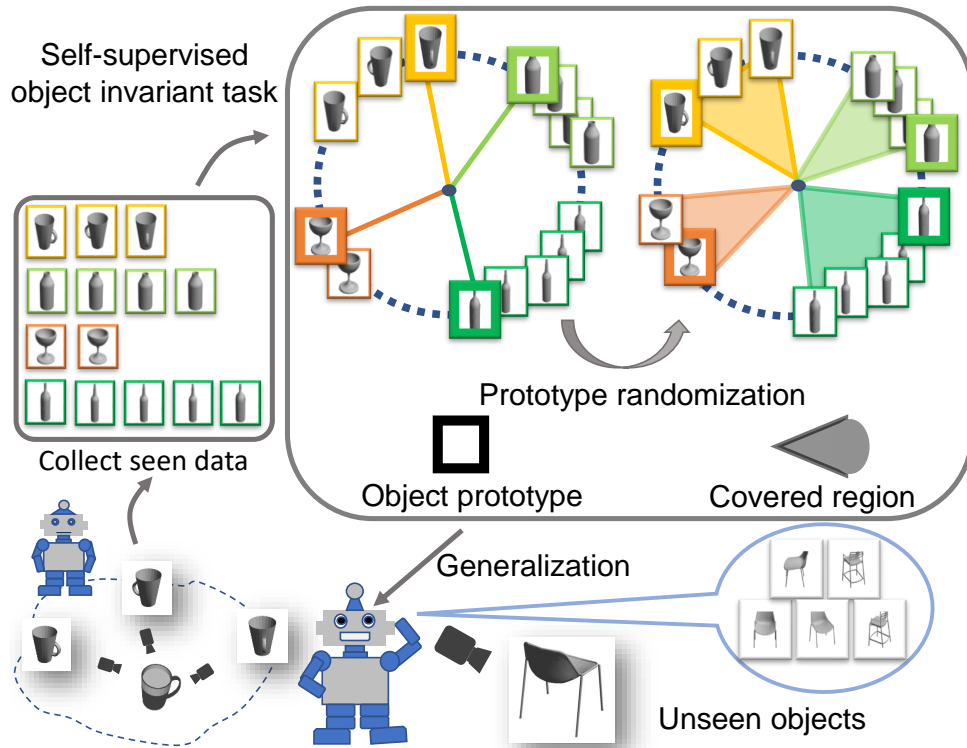
### **3.6 Conclusion**

This work makes several contributions to the study of pose invariance for image classification and retrieval tasks. We started by introducing a functional organization of embeddings to elaborate the relationships between existing methods. As the taxonomy is organized according to different level of invariance, some missing approaches are identified and existing approaches are further generalized. A new family of pose invariant embeddings (PIEs) is then derived from existing methods, by combining a view-to-object model and a object-to-class model. We show that the proposed PIEs have mathematically interesting properties and have good performance for both 1) classification and retrieval, and 2) single and multiview inference. The generalization of PIEs is important because such embeddings can be applied to different tasks and circumstances, which is a more realistic scenario for vision application. Finally, we introduced a multiview dataset, ObjectPI, with images of real objects captured with in the wild backgrounds. We believe that the proposed dataset will complement the synthetic datasets and contribute to the advancement of multiview study.

Chapter 3 is, in full, based on the material as they appear in the publication of “PIEs: Pose Invariant Embeddings”, Chih-Hui Ho, Pedro Morgado, Amir Persekian, and Nuno Vasconcelos, In Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2019 . The dissertation author was the primary investigator and author of this paper.

## **Chapter 4**

# **Exploit Clues from Views: Self-Supervised and Regularized Learning for Multiview Object Recognition**



**Figure 4.1.** Lightweight unsupervised multiview object recognition. A household robot collects multiple object views by moving around, aggregating a multiview object database without view labels. A self-supervised learning algorithm is applied to this database to create an embedding that maps images from same object into an object invariant. At inference time, this embedding generalizes to new views, objects, and object classes.

## 4.1 Introduction

3D recognition has received increasing attention in computer vision in recent years. A popular approach, which we pursue in this work, is to rely on the multiview object representation. Several multiview recognition approaches have been proposed in the literature, including the use of recurrent neural networks[62, 29, 129], feature aggregation from different views[49, 46, 176], graph modeling[44] and integration with other modalities[214, 70, 144, 156]. While achieving good recognition performance, two strong assumptions are made. The first is that a dense set of views, covering the entire range of view angles, is available per object [176]. While some methods support missing views during inference [76, 91, 46], a complete view set is always assumed for training. The second is that all these images are labeled, for both object classes and

view angle. The two assumptions make multiview techniques difficult to implement and limit their generalization. For example, while previous works [176, 91, 46] show strong performance on training classes, recognition on classes unseen during training is usually not considered.

These limitations prevent many applications of interest. Consider, for example, the setting of Fig. 4.1, where a household robot of limited memory is tasked with picking scattered objects and returning them to their locations. In this setting, it is impractical to pre-train the robot with a dense set of labelled views for each object class in the world. Instead, the robot must be able to efficiently learn objects from unseen classes after deployment. This is similar to problems like image retrieval [30, 88] or face verification [165, 191], which are usually solved by metric learning. An embedding is learned from a large dataset of annotated objects, unseen object classes are modelled by projecting example images onto the embedding, and classification is performed with a nearest neighbor classifier. However, a multi-view embedding is challenging to learn in this manner, due to the need for complete and labeled sets of views. In the setting of Fig. 4.1 this means that, after the home robot is deployed, view angle labels must be collected by manually controlling the pose of the training objects, which is impractical.

This problem can be avoided by the introduction of *multiview self-supervised learning* (MV-SSL) methods. SSL is now well established for problems where annotation is difficult [85, 86]. The idea is to use “free labels,” i.e. annotations that can be obtained without effort, to define a surrogate learning task. However, the many surrogate tasks proposed in the literature [2, 154, 111, 212, 222, 107] are poorly suited for multiview recognition. This is because multiview embeddings must enforce an *invariance* constraint, namely that all views of an object map into (or cluster around) a single point in the embedding, which is denoted the *object invariant*. For embeddings with this property, views of objects unseen during training will naturally cluster around object invariants, without requiring view labels, consistency of view angles across objects, or even the same number of views per object. In this case, it suffices for the home robot to collect a set of views per object, e.g. by moving around it, as illustrated in Fig. 4.1. To emphasize the low-complexity of object acquisition under this set-up, we refer to

it as *lightweight unsupervised multiview object recognition* (LWUMOR).

In this work, we seek embeddings with good LWUMOR performance. We consider proxy embeddings [142], which have been shown to perform well for multiview recognition when dense views and class labels are available [76]. To derive an SSL extension, we propose a new surrogate task, where object instances are used as training “classes,” i.e. object identities serve as free labels for learning. We hypothesize, however, that due to the concentration of supervision on class prototypes, these embeddings *only* capture the metric structure of images in the neighborhood of these prototypes, thus overfitting to the training classes. We address this problem with a randomizing procedure, where the parameters of the softmax layer are sampled stochastically from the embeddings of different object views, during training. This has two interesting consequences. First, it forces the learning algorithm to produce an embedding that supports many classifiers, spreading class supervision throughout a much larger region of feature space, and enhancing generalization beyond the training classes. Second, because this supervision is derived from randomized object views, it encourages a stable multiview representation, even when only different view subsets are available per object.

To further enhance multiview recognition performance, this randomization is complemented by an explicit invariance constraint, which encourages the classifier parameters to remain stable under changes of view-point. We denote the resulting MV-SSL embeddings as *view invariant stochastic prototype embeddings* (VISPE). Experimental results on popular 3D recognition datasets show that self-supervised VISPE embeddings combine 1) better performance outside the training set than standard classification embeddings, and 2) faster convergence than metric learning embeddings. Furthermore, for multiview recognition, VISPE embeddings outperform previous SSL methods.

Overall, this work makes three main contributions. The first is the LWUMOR formulation of MV-SSL. This enables multiview recognition without object class or pose labels, and generalizes well to objects unseen at training time. The second is a new surrogate task that relies on randomization of object views to encourage stable multiview embeddings, and out-



performs previous SSL surrogates for multiview recognition. The third is the combination of randomization and invariance constraints implemented by VISPE to learn embeddings of good LWUMOR performance. Extensive experiments validate the ability of these embeddings to learn good invariants for multiview recognition.

## 4.2 Related work

This work is related to multiview recognition, SSL, and regularization by network randomization.

### 4.2.1 Multiview recognition

Multiview recognition is a 2D image-based approach to 3D object recognition. One of the earliest methods is the multiview CNN (MVCNN)[176], which takes multiple images of an object as input and performs view aggregation in the feature space to obtain a shape embedding. Representing 3D objects by 2D images has been shown effective for classification[91, 46] and retrieval[69, 120]. Subsequent research extended the idea by performing hierarchical view aggregation[49, 46]. However, because view aggregation disregards the available supervision for neighboring relationships between views [62], recurrent neural networks [62, 29, 129] and graph convolutional neural networks [44] have been proposed to model multiview sequences. Aside from multiview modeling, [91] treats viewpoint as a latent variable during optimization and achieves better classification accuracy and pose estimation. In the retrieval setting, [120, 69] combine the center loss [202] with a triplet loss [165] to form compact clusters for features from the same object class.

All these methods share several assumptions that make them impractical for LWUMOR. The MVCNN[176] assumes that all object views are presented at both training and inference. Methods that model view sequences [62, 29] require even more detailed viewpoint supervision. Previous works[46, 91, 76] found that these methods experience a significant performance drop when only partial views are available for inference. [91] minimized this drop by treating view-

point as an intermediate variable, while [76] proposed to overcome it with hierarchical multiview embeddings. All these methods assume a full set of training views.

In this work, we relax this constraint, investigating the LWUMOR setting, where only partial object views are available for both training and inference, and no view or image class labels are given. This forces the use of SSL techniques to learn the “implicit” shape information present in a set of object views, and encourages embeddings that generalize better to unseen classes.

## 4.2.2 Self-supervised Learning

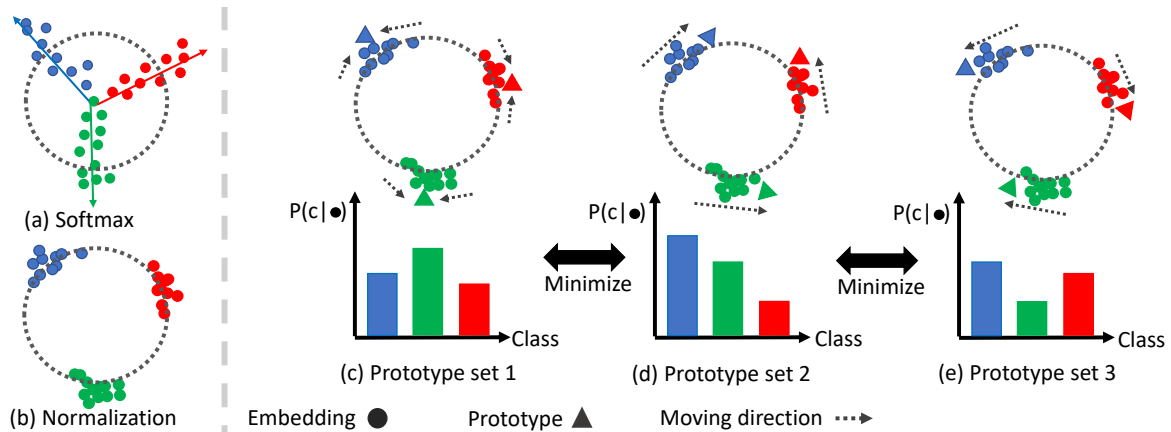
SSL leverages free labels for a surrogate task to train a deep network. Many surrogate tasks have been proposed in the literature. While we provide a brief review of many of these in what follows, most do not seek object invariants and are unsuitable for MV-SSL.

**Context:** based approaches [154, 38, 143] seek to reconstruct images. Autoencoders[55] map images to a low dimensional latent space, from which they can be reconstructed. Similarly, a context encoder[154] reconstructs missing patches from an image conditioned on their surroundings. [38] further leverages spatial image context by predicting the relative positions of randomly cropped patches. Image coloring techniques, which recover the colors of grey scale images [222] or predict pixelwise hue and chromatic distributions[107, 108] leverage color as a form of image context.

**Motion:** based approaches [2, 82, 199, 153] exploit the spatiotemporal coherence of images captured by a moving agent, in terms of relative position[2], optical flow[153] or temporal video structure [199, 81]. The surrogate task becomes to predict camera transformations[2] or segmenting objects[153].

**Sequence:** sorting is another popular task, where sequences can consist of randomly cropped image patches [146] and video clips[4, 138]. Similarly, there have been proposals to remove some color channels from image patches[95] or adding various types of jitter to video clips[111, 196, 195].

**Data augmentation:** type of tasks transform images, leveraging the difference after



**Figure 4.2.** Regularization of a self-supervised embedding by prototype randomization. In all figures, each color represents a single object and views of the same object are marked with same color. (a) softmax embedding, unnormalized features. Solid arrows represent the weight vectors  $w_i$  learned per instance  $i$ . (b) Normalized embedding. (c-e) randomization: 3 different sets of prototypes are used for training. Dashed lines show how view embeddings are encouraged to move towards the corresponding object prototype. Bar plots illustrate how the posterior class probabilities of a given image change when the prototypes are switched. The proposed multiview consistency regularization seeks to further improve the generalization power of the embedding by minimizing these variations.

transformation to define surrogate tasks. [52] predicts the rotation angle of the transformed image, while [212] learns a transformation invariant feature.

**View:** based tasks have been proposed for multiple applications, such as object recognition [80], hand [189] and human [99] pose estimation. Our work is similar to [80] as both consider object recognition. However, [80] requires image sequences while our approach has no such constraint and tackles the problem in an entirely different manner .

**Cluster:** based methods group data with visual similarities into clusters and discriminate different clusters. While [18, 11] group multiple images into the same cluster, [1, 207] treat each image as a cluster. Our work shares the high level idea of the latter, by treating each object as a cluster, but differs in terms of memory usage and efficiency. While [1] is known to be computational demanding, [207] is both memory expensive and inefficient, by requiring storage of features from *all* dataset instances. Furthermore, each feature is updated only once per epoch, which leads to noisy optimization. Our method leverages multiple object views to avoid these problems and is more suitable for the LWUMOR setting.

### 4.2.3 Network randomization

Randomization has been shown to improve network performance and robustness. It can be explained as a form of model ensembling [104, 110], by combining models trained under different conditions. One of the simplest yet most practical randomization procedures is dropout [174, 9], which removes units in the network during training. Dropmax [110] proposed to instead remove classes, training a stochastic variant of the softmax for better classification. The proposed method explores an orthogonal randomization direction, where feature vectors from different object views are chosen as object prototypes during training.

## 4.3 Multiview Self Supervised Learning

In this section, we discuss the proposed MV-SSL approach.

### 4.3.1 Light Weight Unsupervised Multiview Object Recognition

We start by defining the LWUMOR problem and introducing a surrogate task for its solution. Consider a set of objects  $\mathcal{O} = \{o_i\}_{i=1}^N$ , where  $o_i \in \mathcal{O}$  is the  $i^{\text{th}}$  object instance. This consists of a set  $o_i = \{x_i^j\}_{j=1}^{V_i}$  of variable  $V_i$  image views, captured from *unspecified* viewpoints.  $x_i^j \in \mathcal{X}$  denotes the  $j^{\text{th}}$  view of object  $o_i$ . The goal of LWUMOR is to learn an embedding that supports recognition of new views, objects, and object classes from  $\mathcal{O}$ . In this work, this is addressed with SSL, defining the surrogate task as object instance classification. Each object instance is treated as a different class, establishing a labelled image dataset  $\mathcal{D} = \{(x_i^j, y_i^j) | y_i^j = i, \forall j \in V_i\}$ . The surrogate task is solved by a classifier based on an embedding  $f_\theta: \mathcal{X} \rightarrow \mathbb{R}^k$  of parameters  $\theta$ , which maps image  $x$  into  $k$ -dimensional feature vector  $f_\theta(x)$ . This is implemented by a convolutional neural network (CNN). It should be emphasized that this surrogate task requires *no view alignment or labels*. This is unlike previous multiview SSL approaches, which require either view [80] or camera transformation labels [2].

### 4.3.2 Modeling

As is common for CNNs, a classifier can be implemented with a softmax layer

$$P_{Y|X}(i|x) = \frac{\exp(w_i^T f_\theta(x))}{\sum_{k=1}^N \exp(w_k^T f_\theta(x))}, \quad (4.1)$$

where  $w_i$  is the parameter vector of instance  $i$ . This is referred as the “instance classifier” in what follows and is trained by cross-entropy minimization, using the free instance labels for supervision. The optimal embedding and classifier parameters are learned by minimizing the risk

$$\mathcal{R} = \sum_{i,j} -\log P_{Y|X}(i|x_i^j) \quad (4.2)$$

over the image dataset  $\mathcal{D}$ .

Even though it is a strong baseline, the softmax classifier is most successful for closed-set classification, where train and test object classes are the same. In general, the learned embedding  $f_\theta$  does not have a good metric structure beyond these classes. For this reason, alternative approaches have been more successful for open set problems, such as image retrieval [30], face identification [165, 191], or person re-identification [225]. These are usually based on metric learning embeddings, such as pairwise [60] or triplet embeddings [165]. However, these techniques have problems of their own. Because there are many more example pairs or triplets than single examples in  $\mathcal{D}$ , they require sampling techniques that are not always easy to implement and lead to slow convergence.

### 4.3.3 Randomization

In this work, we explore an alternative approach to learn embeddings that generalize beyond the set of training classes, based on the softmax classifier of (4.1). This consists of randomizing the surrogate task and is inspired by previous work in low-shot learning [169], where meta-learning techniques re-sample the classes for which the embedding is trained. The

intuition is that, when the task is changed, the metric structure of the embedding changes as well. This forces the embedding to have a good metric structure over larger regions of the feature space, therefore generalizing better to unseen classes. In this work, we consider randomization strategies that leverage the view richness of multiview datasets to achieve better generalization to unseen classes during training. This is critical in the LWUMOR setting, where the goal is to enable the learning of multiview embeddings without dense view datasets or even view labels. We propose to randomize the embedding by using random feature vectors as classifier parameters  $w_i$  in (4.1).

The idea is summarized in Fig. 4.2 for a problem where  $N = 3$ . Fig. 4.2 (a) shows the vectors  $w_i$  (solid arrows) learned in feature space with the combination of (4.1) and (4.2). Since cross-entropy minimization only aims to separate the seen instances, this embedding leads to feature distributions such as shown in Fig. 4.2 (a). Embeddings of images of the same object are not tightly clustered and can be close to those from other objects. A common procedure to encourage better metric structure (in this case Euclidean) is to normalize the embedding to unit norm [157, 191, 142, 165], i.e. add a normalization layer at the output of  $f_\theta(x)$  such that  $\|f_\theta(x)\|_2 = 1$ . As shown in Fig. 4.2 (b), this maps all feature vectors to the unit norm ball. For simplicity,  $f_\theta(x)$  is assumed to be normalized in all that follows.

In this work, we propose to replace the classifier weight  $w_i$  by the embedding of a *randomly chosen view of object instance*  $o_i$ . This is implemented by defining a *view sampler*  $v_i \in \{1, \dots, V_i\}$  per object instance  $i$ , which outputs a number between 1 and  $V_i$ . This sampler is then used to draw a feature vector  $f_\theta(x_i^{v_i})$  that serves as the parameter vector  $w_i$  of (4.1). The sampled feature vectors are called “prototypes” for  $o_i$ , as shown in Fig. 4.2 (c). A softmax temperature parameter  $\tau$  is also introduced to control the sharpness of the posterior distribution. Larger temperatures originate sharper distributions, smaller temperatures originate more uniform ones. All these transform the softmax layer into

$$P_{Y|X}^s(i|x) = \frac{\exp(f_\theta(x_i^{v_i})^T f_\theta(x) / \tau)}{\sum_{k=1}^N \exp(f_\theta(x_i^{v_i})^T f_\theta(x) / \tau)}, \quad (4.3)$$

---

**Algorithm 1.** Randomization schedule

---

- 1: **Input** Threshold  $t$
  - 2: Use the view samplers  $v_i, \forall i$  to select a set of random prototypes  $\mathcal{W} = \{f_\theta(x_1^{V_1}), \dots, f_\theta(x_N^{V_N})\}$  to use in (4.3).
  - 3: **while** Not convergence **do**
  - 4:   Minimize the risk of (4.2)
  - 5:   **for all**  $i \in N$  **do**
  - 6:      $u \sim \text{Unif}(0,1)$
  - 7:     **if**  $u < t$  **then**
  - 8:       Use  $v_i$  to resample a new prototype  $f_\theta(x_i^{V_i})$
  - 9:        $w_i \leftarrow f_\theta(x_i^{V_i})$
  - 10:     **end if**
  - 11:   **end for**
  - 12: **end while**
- 

where  $s = \{f_i^{V_i}\}_{i=1}^N$  denotes the set of prototypes used to compute the probability.

### 4.3.4 Multiview embeddings

The prototype classifier has the ability to learn a more stable multiview representation than the instance classifier. This, however, depends on the sampling of the prototypes  $f_i^{V_i}$  of (4.3). To study the impact of prototype sampling, we consider different *randomization schedules*, where the view sampler  $v$  is called more or less frequently during learning, using Algorithm 1. In this algorithm, the threshold  $t \in [0,1]$  controls the frequency with which prototypes are changed. If  $t=0$ , prototypes are fixed, and the embedding is denoted a *prototype embedding* (PE). If  $t=1$ , the prototypes can change at every iteration. Fig.4.2 (c-e) illustrates the idea. Starting from an initial prototype set (Fig.4.2 (c)), prototypes are randomized by choosing embeddings of different views of each instance to play the role of prototypes (Fig.4.2 (d-e)) as training progresses.

Mathematically, prototypes belong to the set  $\mathbb{S} = \prod_{i=1}^N \{f_i^j\}_{j=1}^{V_i}$  of all possible combinations of view embeddings across the  $N$  object instances. This set has cardinality  $|\mathbb{S}| = \prod_{i=1}^N V_i$ . The randomization of Algorithm 1 can thus be seen as replacing (4.1) by an ensemble of  $|\mathbb{S}|$  classifiers, during training. This is similar to the dropout[174], but applied to prototypes only. However, unlike dropout, the randomization is structured in the sense that all the prototypes

used to replace  $w_i$  are embeddings  $f_\theta(x_i^y)$  of views from the same object  $o_i$ . This ensembling over views makes  $f_\theta(x)$  a more stable multiview representation. For this reason, the learned embedding is referred to as a *multiview stochastic prototype embedding* (MVSPE).

### 4.3.5 Multiview consistency regularization

The regularization above can be further strengthened by considering the posterior probability distributions of (4.1). During training, the feature embedding is guided to move toward the prototypes used at each iteration, as illustrated by the dashed arrows of Fig. 4.2 (c-e). This causes the variations in the distributions also shown in the figure. The magnitude of these variations is a measure of the view sensitivity of the embedding. For effective LWUMOR, the feature distributions should not vary significantly with the prototype. This would imply that the different views of the instance were effectively mapped into a view invariant representation. It follows that it should be possible to strengthen the invariance of the embeddings by minimizing these variations, i.e. encouraging the distributions  $P_{Y|X}(i|x)$  to remain stable as the set of prototype is varied. This regularization can be enforced by minimizing the average Kullback-Leibler divergence [28]

$$L_{KL} = K \sum_{s_p, s_q \in \mathcal{S}, p \neq q} \sum_{k=1}^N P^{s_p}(k|x) \log \left( \frac{P^{s_p}(k|x)}{P^{s_q}(k|x)} \right), \quad (4.4)$$

where  $K = \frac{2}{|\mathcal{S}|(|\mathcal{S}|-1)}$ , between all pairs of distributions  $P_{Y|X}^{s_p}(i|x)$  and  $P_{Y|X}^{s_q}(i|x)$  of prototype sets  $s_p$  and  $s_q$ , where  $p \neq q$ . When this regularization is used, the resulting embedding is denoted as *view invariant stochastic prototype embedding* (VISPE).

### 4.3.6 Scalable Implementation

In practice, the number of instances in the unlabeled dataset  $\mathcal{D}$  can be as large as 30,000. Given the memory capacity of current GPUs, it is impractical to load all object prototypes in memory at each each training iteration. One of the benefits of randomization as a regularization strategy is that it is fully compatible with the sampling of a small subset of object prototypes.



In our implementation we use  $m=32$  object instances per minibatch. A set  $\mathcal{I} = \{\xi_1, \dots, \xi_m\}$  of distinct instance indexes is randomly sampled, defining a subset of view embedding combinations  $\mathcal{S}' = \prod_{i=1}^m \{f_{\xi_i}^j\}_{j=1}^{V_{\xi_i}}$  from which prototypes are drawn. Prototype sets are then defined as  $s' = \{f_{\xi_i}^{V_{\xi_i}}\}_{i=1}^m$  and the posterior probabilities with

$$P_{Y|X}^{s'}(\xi_i|x) = \frac{\exp(f_{\theta}(x_{\xi_i}^{V_{\xi_i}})^T f_{\theta}(x)/\tau)}{\sum_{k=1}^m \exp(f_{\theta}(x_{\xi_i}^{V_{\xi_i}})^T f_{\theta}(x)/\tau)}, \quad (4.5)$$

where  $i \in \{1, \dots, m\}$ . At each iteration, a pair of prototypes  $s'_1$  and  $s'_2$  is sampled from the subset of prototype combinations  $\mathcal{S}'$ , the risk of classifying a training view  $x_{\xi_i}^j$  of object instance label  $\xi_i$ , using prototype set  $s'_p$ , is computed with

$$L_{s'_p}(i, j) = -\log\left(P_{Y|X}^{s'_p}(\xi_i|x_{\xi_i}^j)\right) \quad (4.6)$$

for  $p \in \{1, 2\}$ , and the KL divergence with

$$L_{KL} = \sum_{k=1}^m P^{s'_1}(k|x_{\xi_i}^j) \log\left(\frac{P^{s'_1}(k|x_{\xi_i}^j)}{P^{s'_2}(k|x_{\xi_i}^j)}\right). \quad (4.7)$$

Finally, the stochastic gradient descent (SGD) loss for training example  $(x_{\xi_i}^j, \xi_i), i \in \{1, \dots, m\}, j \in \{1, \dots, V_{\xi_i}\}$  is

$$L = L_{s'_1} + L_{s'_2} + \alpha L_{KL}. \quad (4.8)$$

In all experiments, we use a temperature  $\tau = 0.05$  and  $\alpha = 5$ . The implementation is based on Pytorch [152], using the VGG16 [167] model as feature extractor and the output of the last layer as feature vector. A standard SGD was used with learning rate 0.001 to train the network for 300 epochs using batch size of 32.

**Table 4.1.** KNN classification results for various baselines, solving different surrogate tasks. RSPE outperforms all self-supervised learning methods, VGG16 pretrained model and instance classifiers.

Datasets Methods / Classes	Surrogate Task	ModelNet		ShapeNet		ModelNet-S	
		seen	unseen	seen	unseen	seen	unseen
Chance	N/A	3.3	10.0	3.3	4.0	3.3	10.0
Pretrained [167]	N/A	62.7	52.7	63.9	58.1	58.2	55.2
Autoencoder[55]	Context	31.8	37.2	29.8	26.3	34.7	38.8
Egomotion [2]	Motion	32.4	34.7	72.6	47.1	33.0	35.2
Puzzle [146]	Sequence	34.4	41.5	67.8	48.6	34.8	42.4
UEL [212]	Data Aug.	47.9	46.5	68.7	53.4	46.4	48.2
ShapeCode [80]	View	39.4	46.5	67.1	42.3	38.8	47.2
MVCNN[176]	N/A	39.6	48.1	30.3	32.4	36.7	44.8
Triplet[165]	N/A	70.1	62.4	81.2	61.2	64.7	62.1
Instance classifier	N/A	57.7	58.9	69.3	60.4	52.3	54.6
PE	Object	69.7	61.7	81.6	63.8	62.1	60.4
MVSPE	Object	70.3	63.2	82.4	64.6	64.6	62.1
VISPE	Object	<b>71.2</b>	<b>64.4</b>	<b>82.9</b>	<b>65.5</b>	<b>66.2</b>	<b>64.3</b>

## 4.4 Experiments

In this section, we evaluate the different self-supervised learning algorithms on three multiview datasets.

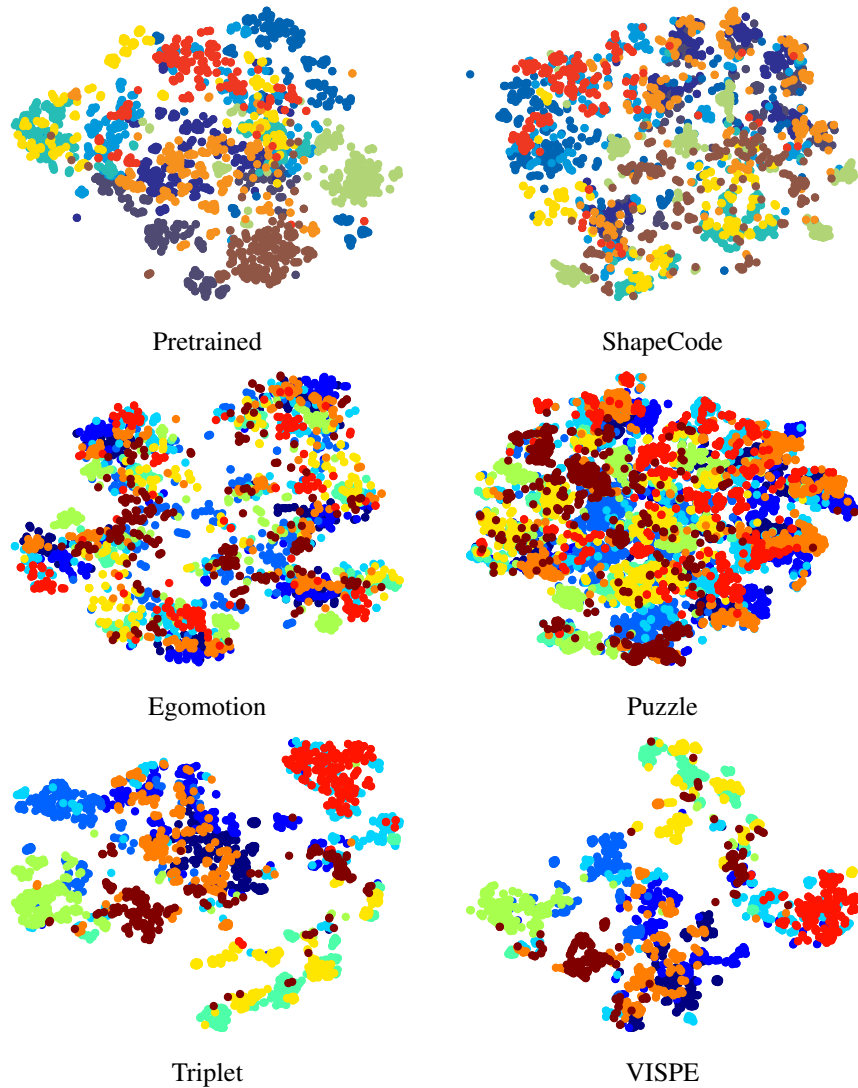
### 4.4.1 Dataset

Three datasets, Modelnet40[206], Shapenet[19] and ModelNet-S, were used in all experiments. Instead of the rendering process of [80]<sup>1</sup>, we adopted the rendering approach and dataset<sup>2</sup> widely used in the multiview literature[176, 46, 91, 120, 69, 76]. Given a synthetic CAD model, 12 views are rendered around it at every 30 degrees. The virtual camera elevates 30 degrees and points to the center of the model. Please see [176] for more details.

**Modelnet**[206] is a synthetic dataset of 3,183 CAD models from 40 object classes. We follow the seen/unseen class split of [80], where unseen classes are those of Modelnet10, a subset of Modelnet40. The standard training and testing partitions [176, 46, 91, 76] are adopted.

<sup>1</sup>We do not have access to the rendered images.

<sup>2</sup><https://github.com/suhangpro/mvcnn>



**Figure 4.3.** TSNE visualization of **unseen class** embeddings. Each color represents a class. RSPE produces more structured embedding.

**ShapeNet**[19] is a synthetic dataset of 55 categories following the Wordnet[136] hierarchy. We use the rendered images from [176], which contains 35,764 training objects and 5,159 test objects. The seen/unseen class split procedure is identical to [80], using the 30 largest categories as seen and the remaining 25 as unseen classes. **Modelnet-S** is sampled by ourselves to resemble a dataset with missing views. This is a subset of Modelnet and shares its train/test setup as well as seen/unseen classes.

## 4.4.2 Baselines

We consider SSL baselines that solve different surrogate tasks, ranging from context, to motion, view, data augmentation and sequence based, as discussed in Section 4.2. All baselines except Jigsaw puzzle[146] use the same backbone (VGG16[167]) and features are extracted from the last network layer. All methods are trained from scratch. For [146], we refer to the original paper for architecture details and the use of pool5 feature.

**Pretrained:** [167] is a VGG16 model pre-trained on ImageNet[31] using class supervision. **Autoencoder** [55] is trained to reconstruct the input image, using an L2 loss to measure the difference between input and reconstruction. **Egomotion** [2] predicts the camera motion between 2 images. Given a pair of images, features are extracted, concatenated and fed into stacked fully connected layers to predict the relative view point difference. Assuming only  $V$  viewpoints exist in the dataset, the model will output  $V - 1$  probabilities, corresponding to the  $V - 1$  viewpoints differences. **Jigsaw puzzle** [146] crops 9 patches from the  $255 \times 255$  input images and shuffles them. The surrogate task is to solve the puzzle. Based on the public source code<sup>3</sup>. **UEL** [212] treats each image as a class and learns a data augmentation invariant feature. Based on the author’s code<sup>4</sup>. **ShapeCode**[80] reconstructs the subsequent views given an object view. We use the loss function proposed in the original paper to train the network. To accommodate the different rendering conditions, the network inputs and generates  $224 \times 224$  images instead of  $32 \times 32$ . **MVCNN**[176] inputs all views of an object and averages their feature vectors, feeding the result to a fully connected classifier that predicts the object identity. **Triplet**[165] is a metric learning approach that learns from triplets of examples: an anchor (input) image, a positive image (from the same object as the anchor) and a negative image (from a different object). Margin 1 performed best in this setting. **Instance classifier** treats each object as a class and trains a VGG16 classifier to minimize (4.2).

---

<sup>3</sup><https://github.com/bbrattoli/JigsawPuzzlePytorch>

<sup>4</sup>[https://github.com/mangye16/Unsupervised\\_Embedding\\_Learning](https://github.com/mangye16/Unsupervised_Embedding_Learning)

### 4.4.3 Classification

All baselines are tested on the 3 datasets, using no labels for training. Inference is based on  $k$  nearest neighbor classification, where  $k$  is the number of images of the class with fewest objects in the dataset. This is 960, 468 and 500 for ModelNet, Shapenet and ModelNet-S, respectively. Each experiment is repeated for seen and unseen classes per dataset.

Table 4.1 shows that all previous surrogate tasks perform poorly for MV-SSL. All proposed methods outperform all baselines, regardless of surrogate task. The only competitive baselines are methods that distinguish objects: instance classifier and triplet embedding. The triplet loss has results comparable to MVSPe but, as discussed in Sec. 4.3.2 and shown in Fig. 4.4 (a), much slower training convergence. While all proposed methods converge in around 80 training epochs for ModelNet, it requires more than 200. Overall, VISPE has the best performance in all datasets, for both seen and unseen classes. This shows that the surrogate task of learning *object invariants* leads to more robust SSL for multiview data. Fig. 4.4 (b) shows the effect of the randomization threshold of Algorithm 1, presenting the average accuracy on unseen classes over ten experiments per threshold. Despite the variance of these results, it is clear that randomization during training strengthens model generalization to unseen classes.

### 4.4.4 Retrieval and Clustering

Ideally, the learned embedding should map images from the same class close together and images from different classes apart, *even for **unseen** classes*. To test this, Kmeans[63] is used to cluster the image embeddings of unseen classes. Two metrics are used to evaluate clustering quality: recall @ K and normalized mutual information (NMI)[132]. NMI is defined as  $\frac{2I(A,C)}{H(A)+H(C)}$ , where  $I$  denotes mutual information,  $H$  entropy,  $A = \{a_1, \dots, a_n\}$  where  $a_i$  is the set of images assigned to class  $i$ , and  $C = \{c_1, \dots, c_n\}$ , where  $c_j$  is the set of images of ground truth class  $j$ . Both metrics are popular in the metric learning literature[142, 173, 171].

Table 4.2 shows results for Modelnet. Again, triplet is the only baseline competitive with

the proposed embeddings, although weaker, and VISPE clearly achieves the best performance. Its effectiveness is highlighted by the large NMI gains. The tightness of its clusters can also be verified in Fig. 4.3, which presents a TSNE visualization of the feature embeddings. Note how VISPE clustering better separates the different colors, which identify the different object classes.

#### **4.4.5 Few-shot object recognition**

The generalization strength of the different embeddings is further tested by experiments with few shot classification. Table 4.2 shows classification accuracy of unseen classes when  $k$  images are labeled per object class. A linear SVM is trained on the labelled feature vectors of Modelnet [206] unseen classes, and used to classify its test set. Similarly to the previous experiments, only the triplet embedding is competitive with MVSPE and VISPE, and VISPE achieves the best performance.

#### **4.4.6 Dependence on number of objects and views**

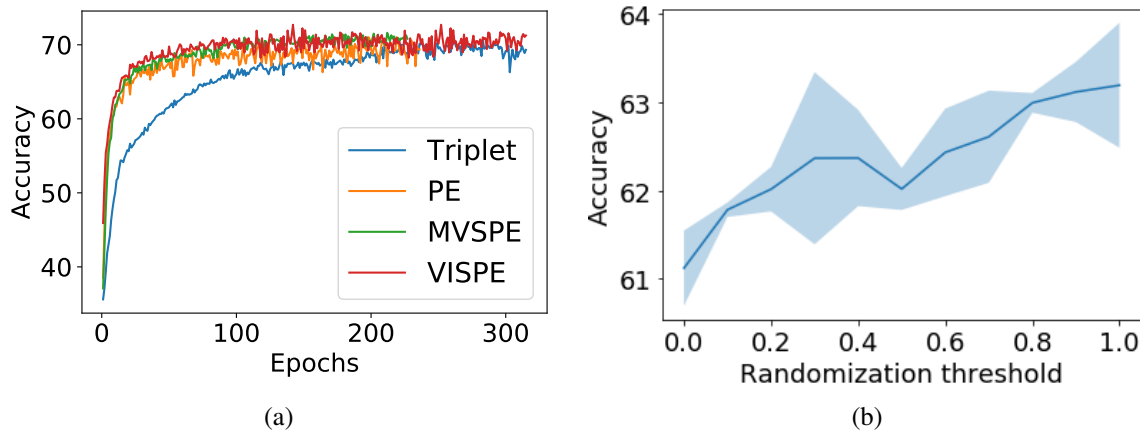
We next consider how many views and objects are needed to learn an embedding that generalizes to unseen classes. To study this, we sample a subset of ModelNet objects and a subset of views per object. The VISPE embedding is then trained on the sampled data and tested on the unseen classes. Fig. 4.5 (a) shows that classification accuracy saturates around 40 objects per class and 8 views per object. Interestingly, when the number of objects is small, capturing more views per object compensates for the lack of object diversity. This is of importance for applications, since it suggests that the embedding could be quickly retrained on a relatively small set of objects, e.g. when a robot has to be deployed on a totally new environment.

#### **4.4.7 Trade-off of training with labels**

Even though supervised learning requires more labeling effort, it performs better on predefined classes. For example, training VGG16[167] with labels on seen classes of ModelNet and ShapeNet yields 84.1% and 87.5% on its test set. However, the performance of KNN on

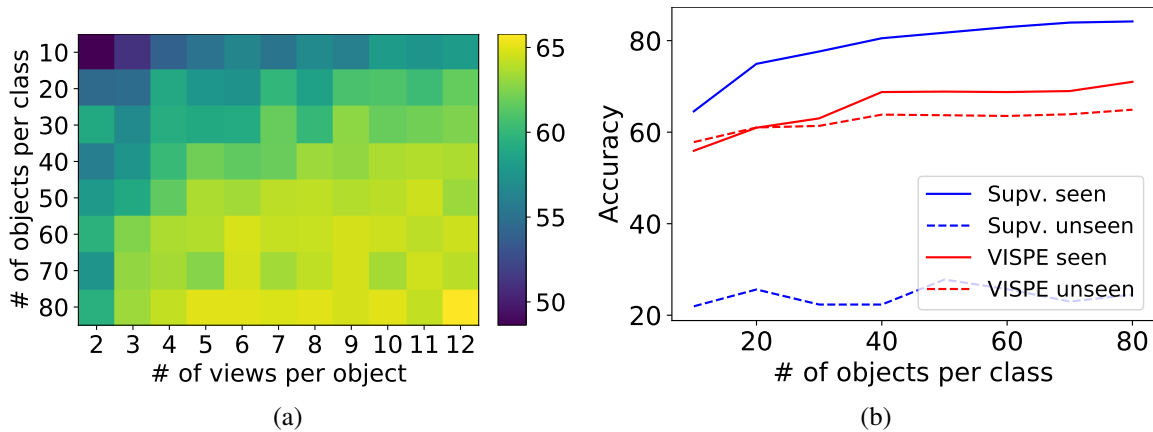
**Table 4.2.** Left: Recall @ k and NMI on Modelnet unseen classes. Right: low shot accuracy for k labeled images.

Methods	Retrieval and Clustering					Low shot k Images		
	Recall				NMI	1	3	5
@1	@2	@4	@8					
Pretrained [167]	94.5	96.6	98.2	<b>99.3</b>	46.7	34.3	46.8	51.2
Autoencoder[55]	81.7	86.8	92.3	95.2	25.4	25.0	30.0	28.0
Egomotion [2]	73.4	80.7	88.0	92.9	7.5	15.1	18.1	19.9
Puzzle [146]	77.8	84.1	89.8	94.0	21.9	21.1	26.8	29.3
UEL [212]	77.8	85.4	91.6	95.7	24.6	23.8	30.9	34.2
ShapeCode [80]	83.4	88.5	93.4	96.2	27.4	28.8	36.1	39.5
MVCNN [176]	80.3	86.7	91.7	95.0	19.3	21.6	27.0	29.5
Triplet [165]	90.8	94.7	97.4	98.8	48.2	41.4	50.3	54.5
Instance classifier	89.1	92.5	95.6	97.4	37.1	28.3	42.2	48.4
PE	91.2	95.0	97.2	98.5	48.2	40.2	49.7	52.9
MVSPE	92.4	95.4	97.7	98.9	48.4	41.5	50.8	54.2
VISPE	<b>95.5</b>	<b>97.7</b>	<b>98.6</b>	99.2	<b>51.1</b>	<b>43.1</b>	<b>52.5</b>	<b>55.9</b>



**Figure 4.4.** (a) Convergence rate of proposed methods and triplet loss. (b) Effect of different randomization threshold on unseen class accuracy.

unseen classes drops significantly to 20.9% and 35.1%, which is much worse than most SSL results in Table 4.1. This begs the question of when SSL should be used. As shown in Fig. 4.5 (b), when few objects per class are available for ModelNet, VISPE is a better choice than supervised learning, because it generalizes much better ( $> 30\%$ ) on unseen classes and maintains comparable performance ( $< 10\%$ ) on seen classes.



**Figure 4.5.** (a) Accuracy (represented by color) of VISPE on unseen classes, as a function of views per object and objects per class in training set. (b) Trade-off of training with labels as a function of object per class.

## 4.5 Conclusion

In this work, we made several contributions to MV-SSL. We started by discussing the current impractical assumption of fully supervised multiview recognition, which requires intensive labeling. We then relaxed this assumption by investigating MV-SSL methods, where only “free labels” (image to object association) are required. Embeddings that generalize to both seen and unseen data were then learned with variants of this MV-SSL surrogate task. These variants differ in the regularization used to encourage object invariant representations. We started by leveraging view information by choosing the embedding of a random object view as the object prototype. A randomization schedule was then proposed to sample prototypes stochastically. This can be seen as an ensembling over views, to encourage stable multiview embeddings. To strengthen the learning of object invariants, we finally proposed a multiview consistency constraint. The combination of all these contributions produced a new class of view invariant stochastic prototype embeddings (VISPE). These embeddings were shown to outperform other SSL methods on seen and unseen data for both multiview classification and retrieval. While we have not studied the semi-supervised setting, where few labels are provided, in great detail, this setting is also supported by VISPE. We believe that these are important contributions for the much needed



extension of multiview recognition to the LWUMOR setting of Figure 4.1, which is of interest for many real world applications.

Chapter 4 is, in full, based on the material as they appear in the publication of “Exploit Clues from Views: Self-Supervised and Regularized Learning for Multiview Object Recognition”, Chih-Hui Ho, Bo Liu, Tz-Ying Wu, and Nuno Vasconcelos, In Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2020 . The dissertation author was the primary investigator and author of this paper.

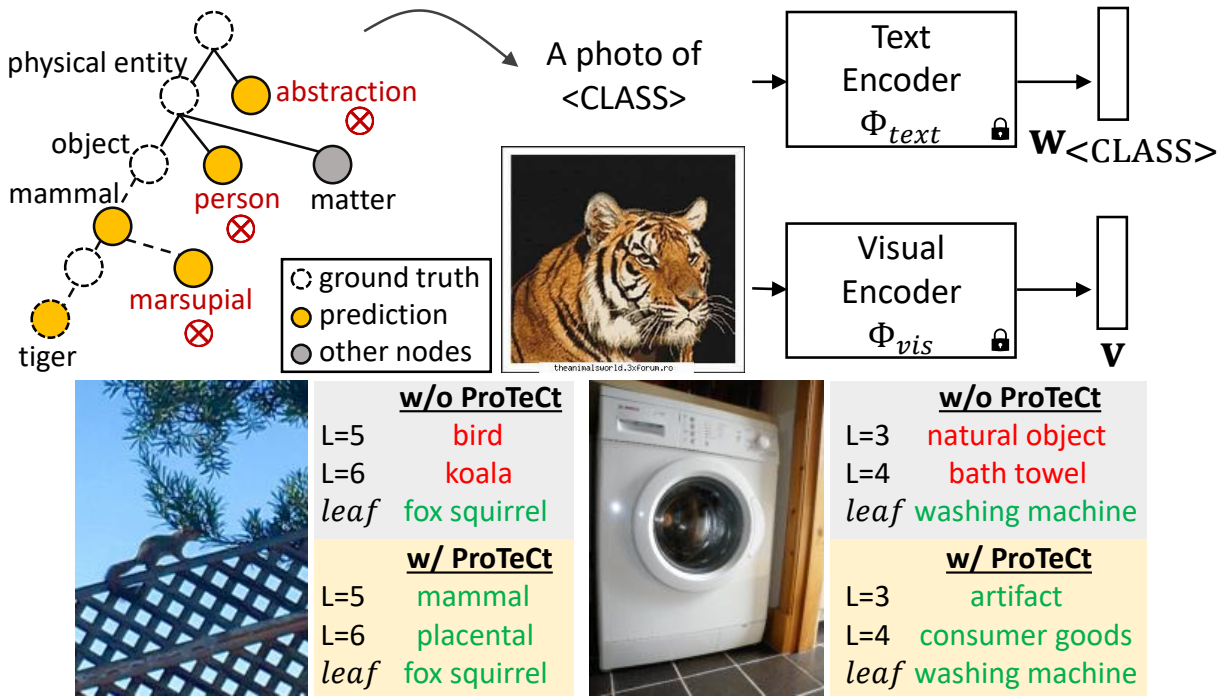
## **Chapter 5**

# **ProTeCt: Prompt Tuning for Taxonomic Open Set Classification**

## 5.1 Introduction

Vision-language foundation models (FMs) have opened up new possibilities for image classification. They are large models, trained on large corpora, to learn aligned representations of images and text. For example, CLIP [159] combines text and image encoders trained with 400M image-text pairs in an open vocabulary fashion, using a contrastive loss [23, 24, 172, 186]. Zero-shot classification can then proceed by leveraging the alignment of image and text features. Each class name is first converted to a text prompt, e.g., “a photo of [CLASS],” which is fed to the text encoder. The resulting text feature is then used as the parameter vector of a softmax classifier of image feature vectors. Since the training does not emphasize any particular classes, CLIP supports open set classification. Several works [228, 227, 94, 218] have shown that classification accuracy can be enhanced by fine-tuning the FM on the few-shot setting (i.e. few examples per class). To adapt the model and maintain the alignment between text and images, these works augment the FM with a few learnable prompts [228, 227, 218, 94]. The model parameters are then frozen and only the prompts are optimized. This process is known as **prompt tuning** and can achieve significant improvement over the zero-shot performance, on the dataset of interest.

While prompting enables classifiers to be designed for virtually any classes with minimal dataset curation effort, it should not compromise the open set nature and generality of the FM representation. In this work, we consider the setting where “open set” means the ability to refer to concepts at different levels of granularity. Consider, for example, an educational application in biology. While at grade school level it will teach students to classify animals into (“cat”, “dog”, “lizard”), at the high-school level the **exact same images** should be classified into much more detailed classes, e.g. (“iguana”, “anole”, “komodo”, etc.) for lizards. A classifier that classifies an image as a “komodo” lizard for high schoolers but “dog” for gradeschoolers is not useful and trustworthy. Advanced biology students should even learn about the taxonomic relations between the different species. This requires a representation that supports hierarchical classification [210, 205, 112, 134], where the classifier understands the relations between the



**Figure 5.1.** (Top) An example of class hierarchy, where CLIP predicts the tiger image as “person” at the internal hierarchy level. (Bottom) Correct/incorrect model predictions (green/red) of CoOp w/ and w/o ProTeCt, respectively.  $L$  denotes the tree level.

superclasses and subclasses that compose a class hierarchy, and provide correct predictions *across* hierarchy levels.

Fig. 5.1 shows an example of a class hierarchy built from ImageNet [33] classes, according to the WordNet [136]. When faced with the image of a tiger, the classifier should provide a correct prediction under the label sets  $\mathcal{Y}_1 = (\text{“dog”}, \text{“cat”}, \text{“tiger”})$ ,  $\mathcal{Y}_2 = (\text{“person”}, \text{“animal”}, \text{“insect”})$  or  $\mathcal{Y}_3 = (\text{“physical entity”}, \text{“abstraction”})$ , where the correct prediction is shown in bold. Note that, given a classifier with this property, teachers have the ability to define many different classification problems, for many levels of granularity, tailoring the same app to different uses. We refer to this setting as **taxonomic open set (TOS)** classification. In many real-world applications, support for this restricted form of open set classification is much more important than support for unbounded open set classification. In the example above, biology teachers do not really care if the classifier can still discriminate between cars and trucks, or soda cans and wine bottles. Hence, these classes are irrelevant for the app developer.

**Table 5.1.** TOS classification performance of CLIP-based classifiers.

Method	$Acc_{leaf}$	HCA	MTA
CLIP [159]	68.36	3.32	48.21
CoOp [228]	71.23	2.99	46.98
MaPLe [94]	70.70	4.15	48.29

In principle, TOS should be trivially supported by FMs. Even at zero-shot level, it should suffice to specify [CLASS] names at the desired levels of granularity. However, our experiments show that this does not work because the representation of most FMs fails to capture taxonomic relations. This is illustrated for CLIP in Fig. 5.1. While the model knows that the object is a tiger, it fails to know that it is “a physical entity” and not an “abstraction” or that it is a “placental mammal” and not a “marsupial,” indicating that it only understands class relations locally. It can perform well for the leaf class label set  $\mathcal{Y}_1$ , but cannot reason across abstraction levels, and can thus not support TOS classification. To enable TOS, we introduce the notion of **hierarchical consistency**, and a new *hierarchical consistency accuracy* (HCA) metric, where classification is defined with respect to a taxonomic tree and its success requires the correct prediction of all superclasses (e.g., mammal, object and physical entity) of each ground truth leaf class (e.g., tiger). This is complemented by the notion of **TOS classification**, where classifiers can have any set of nodes in the class hierarchy as the label set, and a new *mean treecut accuracy* (MTA) metric, which estimates classification accuracy in this setting.

Our experiments show that neither CLIP nor existing prompt tuning methods [228, 227, 94] perform well under the HCA and MTA metrics of the TOS setting. Fig. 5.1 illustrates the problem and the *inconsistent* CLIP class predictions (orange dots) across hierarchy levels. Table 5.1 compares the standard (leaf) accuracy of the model with HCA/MTA, under both the zero-shot and two prompt-tuning settings. While the leaf accuracy is quite reasonable, hierarchical consistency is very poor. To address this problem, we propose a novel prompt-tuning procedure, denoted *Prompt Tuning for Hierarchical Consistency* (ProTeCt), that explicitly targets the TOS setting. Given a dataset of interest, a class hierarchy is extracted from the associated metadata, a generic public taxonomy (e.g. WordNet [136]), or a special purpose taxonomy

related to the application (e.g. scientific taxonomies). Since FMs support classification with open vocabulary, any node in the hierarchy can be used in the label set of the classifier. Prompts are then learned with the help of two new regularization losses that encourage hierarchical consistency. A *dynamic treecut loss* (DTL) encourages correct classification at all tree levels by sampling random tree cuts during training. A *node-centric loss* (NCL) contributes additional supervision to each internal tree node to increase classification robustness for all granularities of the hierarchy.

Experiments show that ProTeCt significantly improves the performance of prompt tuning methods, like CoOp [228] and MaPLe [94], under TOS setting. Fig. 5.1 shows the predictions of CoOp at different hierarchy levels before/after adding ProTeCt. Under the HCA/MTA metrics, the improvement can be more than 15/25 points on Cifar100, SUN and ImageNet datasets. Following [228, 227, 94], we show that these gains hold for zero-shot domain generalization to several variants of ImageNet [161, 192, 75, 71], showing that hierarchical consistency transfers across datasets. Furthermore, ablations show that ProTeCt can be used with different CLIP architectures, parameter tuning methods and taxonomies.

Overall, this work makes four contributions. First, we introduce the TOS setting, including two novel metrics (HCA and MTA) that evaluate the consistency of hierarchical classification. Second, we show that neither zero-shot CLIP nor existing prompting methods fare well in this setting. Third, we propose a novel prompt-tuning method for the TOS setting, ProTeCt, which improves hierarchical consistency by combining DTL and NCL losses. The former relies on a dynamic stochastic sampling of label sets involving multiple levels of the hierarchy, while the latter regularizes the classification of every node in the hierarchy. Finally, ProTeCt is shown to outperform vanilla prompt tuning methods on three datasets with different hierarchies. Extensive ablations demonstrate that ProTeCt is applicable to different parameter tuning methods, CLIP architectures, taxonomies and the learned hierarchical consistency transfers to unseen datasets from different image domains.

## 5.2 Related Work

**Prompt Tuning of Vision-Language Models:** Many large vision-language FMs have been proposed in recent years [220, 41, 198]. Despite their promising zero-shot performance, several works [227, 228, 84, 94] have shown that their few-shot finetuning with a dataset from the target application can further improve performance. Unlike conventional finetuning methods that optimize the entire model, these methods are designed to (a) be parameter efficient and (b) maintain the general purpose feature representation of the FM. Several such tuning methods have been proposed for CLIP [159]. Inspired by prompt tuning techniques from the language literature [114, 117, 123], CoOp [228] inserts learnable prompts at the CLIP text input. CoCoOp [227] further learns a meta-network to generate an image-conditioned prompt. The idea of connecting image and text prompts is further extended by UPT [218] and MaPLe [94]. The former learns a unified transformer for generating an image and text prompt, the latter learns a coupling function to generate image prompts from text prompts. LASP [16] proposed a text-to-text cross-entropy loss to regularize the distribution shift when different prompts are used. Unlike these works, we investigate the TOS problem, where labels can be drawn from any level in a class taxonomy, and propose prompting techniques to improve hierarchical classification consistency. This is shown to be compatible with several of the above prompt-tuning methods without degrading their leaf classification accuracy.

**Hierarchical Classifiers:** Hierarchical classification aims to predict labels at different levels of a class hierarchy. Early works [134, 223, 32, 163, 166, 35] date back to the era before deep learning and are not directly applicable to deep learning-based models. Several works [210, 54, 229, 125, 3, 96] propose hierarchical classifiers for CNN-based deep models. For example, [54, 229, 125] use additional convolutional modules to learn a hierarchical feature space. It is unclear how these approaches generalize to the recent transformer-based architectures [39, 127, 126]. Furthermore, prior works [210, 54, 229, 125, 3, 205] finetune the entire model, which requires substantial data and computation, especially at the FM scale. In this work, we study

the problem of hierarchical consistency for foundational vision-language models (e.g., CLIP). While CLIP-based classifiers [159, 227, 228] have outstanding zero/few-shot performance, we show that they produce inconsistent predictions for label sets of different granularity and cannot be used in the TOS setting. We propose an efficient prompt tuning method to address this.

### 5.3 Preliminaries

**Foundation Models (FMs):** Visual-language FMs are composed by a text  $\Phi_{text}$  and a visual  $\Phi_{vis}$  encoder, which extract features from text and images, respectively. The two encoders are optimized by contrastive training [186, 172, 24, 23] to create a joint representation for the two modalities. Since the encoders are learned from a large-scale corpus of image-text pairs, the features are general and support various downstream tasks, e.g., image classification [228, 227, 94, 218] and segmentation [200, 128]. While in this work we use the CLIP [159], ProTeCt should generalize to other FMs.

**Image Classification with FMs:** Given a label set  $\mathcal{Y} = \{t_y\}_{y=1}^C$ , a zero-shot classifier can be designed in the FM representation space by introducing a weight vector  $\mathbf{w}_y$  per class  $y$ . These weight vectors are obtained by simply using the class name  $t_y$  (e.g., “dog”) as a text encoder prompt, i.e.,  $\mathbf{w}_y = \Phi_{text}(Emb_t(t_y)) \in \mathbb{R}^k$ , where  $Emb_t(\cdot)$  is a word embedding. Given these weight vectors, an image classifier of label set  $\mathcal{Y}$  can be implemented by computing class posterior probabilities with

$$p(t_y|\mathbf{x};\mathcal{Y}) = \frac{\exp(\cos(\mathbf{w}_y, \mathbf{v})/\tau)}{\sum_{t_j \in \mathcal{Y}} \exp(\cos(\mathbf{w}_j, \mathbf{v})/\tau)}, \quad (5.1)$$

where  $p(t_y|\mathbf{x};\mathcal{Y})$  is the probability of class label  $t_y$  given image  $\mathbf{x}$ ,  $\mathbf{v} = \Phi_{vis}(Emb_v(\mathbf{x})) \in \mathbb{R}^k$  the visual feature vector,  $Emb_v(\cdot)$  an image embedding,  $\cos(\cdot, \cdot)$  the cosine similarity metric, and  $\tau$  a temperature hyperparameter. Classification performance can usually be improved by inferring the classifier parameters  $\mathbf{w}_y$  from multiple text prompts, e.g. by including context words such as a prompt prefix  $p$  = “a photo of”, or  $p$  = “a drawing of”, computing  $\mathbf{w}_y = \Phi_{text}(Emb_t(\{p, t_y\}))$ , and



ensembling the vectors  $\mathbf{w}_y$  obtained from multiple prompts [159, 228]. This, however, requires multiple forward passes through  $\Phi_{text}$  during inference and can be undesirable for downstream applications.

More efficient inference can be achieved with prompt tuning [228, 227, 94, 218], which leverages a set of learnable parameters  $\{\mathbf{c}_m^t\}_{m=1}^M$  as context features. These are prepended to each class name embedding  $Emb_t(t_y)$  as text prompts, to produce the weight vectors  $\mathbf{w}_y = \Phi_{text}(\{\mathbf{c}_1^t, \dots, \mathbf{c}_M^t, Emb_t(t_y)\})$ . Note that each  $\mathbf{c}_i^t$  has the same dimension as the word embedding. Given a training dataset  $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^N$ , context features can be end-to-end optimized with the cross-entropy loss

$$L_{\mathcal{D}}(\mathbf{C}^t) = \frac{1}{N} \sum_{i=1}^N \sum_{t_j \in \mathcal{Y}} -\mathbb{1}(t_j = t_{y_i}) \log p(t_j | \mathbf{x}_i; \mathcal{D}, \mathbf{C}^t) \quad (5.2)$$

for the classifier of (5.1), where  $\mathbb{1}(\cdot)$  is the indicator function, and  $\mathbf{C}^t$  the matrix of context features. Similarly, learnable prompts  $\mathbf{c}_i^v$  can be inserted into the image branch, i.e.  $\mathbf{v} = \Phi_{vis}(\{\mathbf{c}_1^v, \dots, \mathbf{c}_M^v, Emb_v(\mathbf{x})\})$ , for better visual adaptation [84, 218, 94]. To prevent compromising the generalization of the FM embeddings, the parameters of the two encoders (i.e.,  $\Phi_{text}, \Phi_{vis}$ ) are frozen in the few-shot setting. In this paper, we consider two prompt tuning variants, CoOp [228] and MaPLe [94], the former using learnable prompts in the text branch, and the latter on both branches.

**Class Taxonomy:** A class taxonomy  $\mathcal{Y}^{tax}$  organizes classes into a tree where classes of similar semantics are recursively assembled into superclasses, at each graph node (e.g. “dog” is a superclass of “Chihuahua” and “Corgi”). For a tree hierarchy,  $\mathcal{T}$ , each node  $n \in \mathcal{N}$  has a single parent and multiple child nodes  $Chd(n)$ , where  $\mathcal{N}$  is the set of tree nodes. Given a set of classes  $\{t_y\}_{y=1}^C$ , a tree hierarchy  $\mathcal{T}$  can be built by treating  $\{t_y\}_{y=1}^C$  as leaf nodes (where  $Chd(t_y) = \emptyset$ ), i.e.,  $Leaf(\mathcal{T}) = \{t_y\}_{y=1}^C$ , and recursively grouping classes in a bottom-up manner until a single root node is created, according to the similarity relationships defined by the taxonomy  $\mathcal{Y}^{tax}$ . For example, ImageNet [33] classes are organized into a tree of 1,000 leaf nodes derived from

the WordNet [136] taxonomy. Nodes that are not at the leaves are denoted as internal nodes  $\mathcal{N}^{int} = \mathcal{N} \setminus \text{Leaf}(\mathcal{T})$ .

## 5.4 Taxonomic Open Set Classification

**Definition:** A significant advantage of FMs for practical applications is their support for open set classification. Since the classifier of (5.1) can be implemented with any class names  $t_y$ , and the FM is trained with an open vocabulary, it is possible to perform classification for virtually any classes. Prompting methods improve the classification of the classes defined by the label set  $\mathcal{Y}$ , but attempt to maintain this generality. However, for most applications “open set” does not mean the ability to recognize “any possible word.” On the contrary, the whole point of prompt tuning is to enhance the FM performance for a given application *context*. This context defines what “open set” truly means for the application. In practice, it frequently means “all the possible ways” to refer the classes in  $\mathcal{Y}$ .

One important component of this requirement is the ability to describe classes at different levels of granularity. For example, while user A (a car mechanic) may need to know if an image depicts a “Fan Clutch Wrench” or a “Box-Ended Wrench,” user B (a retail store worker) may need to know if the exact same image depicts a “a mechanic’s tool” or a “plumber’s tool.” A FM-based classification app should be deployable in both the car garage or the retail store. However, because the app is a tool classification app, the prompted model does not need to be good at recognizing “lollipops,” which are beyond the context of the app. On the other hand, it is undesirable to have to prompt-tune the app for every specific use or user group. Ideally, it should be possible to prompt tune the FM *once*, with respect to the *entire* class taxonomy  $\mathcal{Y}^{tax}$  of tools. The app can then be deployed to each user base without any retraining, by simply drawing the most suitable class names  $t_y$  from  $\mathcal{Y}^{tax}$ . We refer to this problem as **Taxonomic Open Set (TOS)** classification and introduce a formal definition in the remainder of this section.

**Datasets:** Most existing classification dataset can be used to study the TOS problem,

since the very nature of taxonomies is to group objects or concepts into semantic classes of different levels of granularity. Hence, most vision datasets are already labeled taxonomically or adopt classes defined by a public taxonomy, usually WordNet [136]. We consider three popular datasets: Cifar100 [100], SUN [209] and ImageNet [33]. ImageNet is complemented by the ImageNetv2 [161], ImageNet-S [192], ImageNet-A [75] and ImageNet-R [71] to enable the study of generalization across image domains. For each dataset, the K-shot setting is considered, where K images per class are sampled for training. We consider  $K = \{1, 2, 4, 8, 16\}$ .

**Label sets:** Given a dataset  $\mathcal{D}$  and class hierarchy  $\mathcal{Y}^{tax}$  a label set  $\mathcal{Y}$  is defined at each level of granularity, according to the latter. The leaf label set  $\mathcal{Y}_{leaf}$  is defined as the set of classes of  $\mathcal{D}$  and the class hierarchy  $\mathcal{T}$  is built recursively, denoting by  $\mathcal{Y}_n = Chd(n)$  the set of class labels for the children of node  $n$ . In our experiments, we adopt the default hierarchy of the SUN dataset and use WordNet [136] to build the hierarchy for Cifar100 and ImageNet. The resulting class hierarchies are as follows. Cifar100 [100] contains 100 leaf nodes and 48 internal nodes. SUN contains 324 leaf nodes and 19 internal nodes (after pruning 73 leaf classes that have confusing superclasses). ImageNet [33], ImageNetv2 [161] and ImageNet-S [192] share a class hierarchy of 1,000 leaf nodes and 368 internal nodes. ImageNet-A [75] and ImageNet-R [71] only contain 200 subclasses and the corresponding internal nodes from the ImageNet hierarchy.

**Metrics:** Given a classifier

$$\hat{y}(\mathbf{x}; \mathcal{Y}) = \operatorname{argmax}_{t_y \in \mathcal{Y}} p(t_y | \mathbf{x}; \mathcal{Y}) \quad (5.3)$$

using a label set  $\mathcal{Y}$ , several metrics are proposed to evaluate TOS performance.

**Leaf Accuracy** is defined as

$$Acc_{leaf} = \frac{1}{N} \sum_{i=1}^N \mathbb{1}[\hat{y}(\mathbf{x}_i; \mathcal{Y}_{leaf}) = t_{y_i}] \quad (5.4)$$

and measures the classification accuracy at the leaves of the taxonomic tree. These are usually

defined as the “dataset classes”. This metric enables comparison of hierarchical classifiers to standard, or *flat*, classifiers which only consider the leaf classes.

**Hierarchical Consistent Accuracy (HCA)** is defined as

$$HCA = \frac{1}{N} \sum_{i=1}^N (\mathbb{1}[\widehat{y}(\mathbf{x}_i; \mathcal{Y}_{leaf}) = t_{y_i}] \prod_{n \in \mathcal{A}(t_{y_i})} \mathbb{1}[\widehat{y}(\mathbf{x}_i; \mathcal{Y}_n) \in \mathcal{A}(t_{y_i}) \cup \{t_{y_i}\}]), \quad (5.5)$$

where  $\mathcal{A}(n)$  denotes all the ancestors of node  $n$ , and  $t_{y_i}$  is the leaf node corresponding to class label  $y_i$ . While  $Acc_{leaf}$  considers successful any correct classification at the leaf level of the tree, the  $HCA$  is stricter. It declares a success only when all the ancestors of the leaf node are correctly classified. In other words, each sample needs to be classified correctly at each tree level to be viewed as correctly classified under the  $HCA$ .  $Acc_{leaf}$  is an upper bound for the  $HCA$ .

**Mean Treecut Accuracy (MTA)** estimates the expected accuracy under the TOS classification setting. It computes the average accuracy over a set of treecuts  $\mathcal{T}_c \in \Omega$ ,

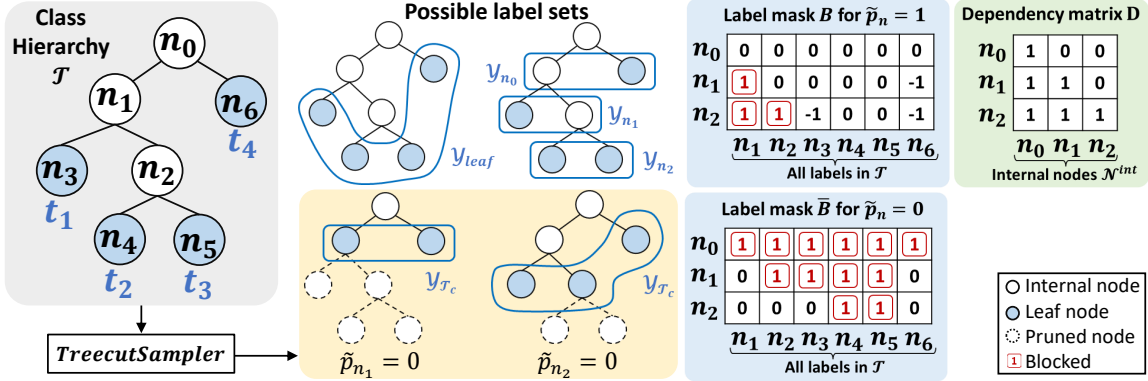
$$MTA = \frac{1}{|\Omega|} \sum_{\mathcal{T}_c \in \Omega} \frac{1}{N} \sum_{i=1}^N \mathbb{1}[\widehat{y}(\mathbf{x}_i; \mathcal{Y}_{\mathcal{T}_c}) = t_{y_i}], \quad (5.6)$$

where  $\mathcal{Y}_{\mathcal{T}_c} = Leaf(\mathcal{T}_c)$ . However, as shown by the following lemma the set of all possible tree cuts in the hierarchy  $\mathcal{T}$  is usually very large.

**Lemma 5.4.1.** *For a balanced  $M$ -ary tree with depth  $L$  (root node is excluded and is at depth 0), the number of all valid treecut is  $L + \sum_{l=2}^L \sum_{k=1}^{N-1} \frac{N!}{k!(N-k)!} |_{N=M^l-1}$ .*

For example, a tree with  $M = 2$  and  $L = 6$  has more than 4 billion treecuts. For a dataset like ImageNet ( $L = 15$ ) this number is monumental. Thus, we randomly sampled  $|\Omega| = 25$  treecuts from  $\mathcal{T}$  in all experiments. Our preliminary experiments have shown that the MTA is already fairly stable for this sample size.

**State-of-the-art:** To test TOS performance of the CLIP model with existing prompting



**Figure 5.2.** (Left) Multiple possible label sets are available in a class hierarchy. The label set can cover nodes at same level or across different hierarchy levels. (Right) Predefined matrices for efficient treecut sampling used in Algorithm 2.

techniques, we performed an experiment on ImageNet. Table 5.1 summarizes the performance of the different methods under the three metrics. Two conclusions are possible. First, the sharp drop from  $Acc_{leaf}$  to  $HCA$  shows that none of the methods make consistent predictions across the class hierarchy. Second, the low MTAs show that the expected accuracy of TOS classification is dramatically smaller than that of flat classification (leaf classes).

## 5.5 Prompt Tuning for Hierarchical Consistency

To enhance TOS performance of FMs, we propose *Prompt Tuning for Hierarchical Consistency* (ProTeCt). ProTeCt can be implemented with many existing prompt tuning methods (e.g., CoOp, MaPLE). These methods optimize context prompts using the cross-entropy loss of (5.2) with leaf label set  $\mathcal{Y}_{leaf}$ . While this optimizes leaf accuracy  $Acc_{leaf}$ , it is not robust to label set changes, even for label sets comprised of superclasses of  $\mathcal{Y}_{leaf}$ . A simple generalization would be to replace (5.2) with  $\mathcal{L}(\mathbf{C}^t) = \sum_{\mathcal{Y}_p \in \mathcal{T}} \mathcal{L}_{\mathcal{Y}_p}(\mathbf{C}^t)$ , i.e., to consider all the partial label sets  $\mathcal{Y}_p$  of the tree  $\mathcal{T}$ . However, for sizeable taxonomies, this involves a very large number of label sets and is not feasible. ProTeCt avoids the problem by dynamically sampling label sets from  $\mathcal{T}$  during training, with a combination of two learning objectives, a *node-centric loss* (NCL) and a *dynamic tree-cut loss* (DTL).

**Node-Centric Loss (NCL):** NCL is the aggregate cross-entropy loss of (5.2) over all node-centric label sets  $\mathcal{Y}_n = \text{Chd}(n)$  defined by each internal node  $n \in \mathcal{N}^{int}$  of the hierarchy, i.e.,

$$\mathcal{L}_{NCL}(\mathbf{C}^t) = \frac{1}{|\mathcal{N}^{int}|} \sum_{n \in \mathcal{N}^{int}} L_{\mathcal{Y}_n}(\mathbf{C}^t). \quad (5.7)$$

NCL optimization encourages prompts that robustify the classification at the different granularities. For example, ‘‘Corgi’’ should be classified as ‘‘mammal’’ within the animal label set  $\mathcal{Y}_{n_1} = \{\text{mammal, reptile, bird}\}$ , as a ‘‘dog’’ within the mammal label set  $\mathcal{Y}_{n_2} = \{\text{dog, cat, elephant, tiger}\}$ , and so forth.

**Dynamic Treecut Loss (DTL):** While NCL calibrates node classification, guaranteeing consistency within each node, the label sets of TOS classification can also span different sub-trees of the hierarchy, including nodes at different levels, e.g.,  $\mathcal{Y} = \{\text{dog, cat, elephant, tiger, reptile, bird}\}$ . DTL seeks to calibrate such label sets, by aggregating the cross-entropy loss of (5.2) dynamically, i.e., on an example basis, over randomly sampled label sets  $\mathcal{Y}_{\mathcal{T}_c} = \text{Leaf}(\mathcal{T}_c)$  comprised of the leaves of the tree cuts  $\mathcal{T}_c$  (sub-trees) of  $\mathcal{T}$ . At each training iteration, a random tree cut  $\mathcal{T}_c$  is sampled with the *TreeCutSampler* procedure of Algorithm 2, as illustrated on the middle of Fig. 5.2, to define the loss

$$\mathcal{L}_{DTL}(\mathbf{C}^t) = L_{\mathcal{Y}_{\mathcal{T}_c}}(\mathbf{C}^t) \quad \mathcal{T}_c \sim \text{TreecutSampler}(\mathcal{T}, \beta), \quad (5.8)$$

where  $\beta \in [0, 1]$  is a rate of tree dropout. For this, a Bernoulli random variable  $P_n \sim \text{Bernoulli}(\beta)$  of dropout rate  $\beta$  is defined for each internal node  $n \in \mathcal{N}^{int} \setminus n_0$ . The algorithm descends the tree  $\mathcal{T}$ , sampling a binary drop-out variable  $p_n$  at each node. If  $p_n = 1$ , node  $n$  is kept in the pruned tree  $\mathcal{T}_c$ . Otherwise, the sub-tree of  $\mathcal{T}$  rooted with  $n$  is dropped from  $\mathcal{T}_c$ . The parameter  $\beta$  controls the degree of pruning. Larger  $\beta$  induces the pruning of more tree nodes, while  $\beta = 0$  guarantess that  $\mathcal{Y}_{\mathcal{T}_c} = \mathcal{Y}_{\text{leaf}}$ . The root node  $n_0$  is excluded, as  $p_{n_0} = 0$  would imply discarding the whole  $\mathcal{T}$ .

The *TreeCutSampler* algorithm is an efficient procedure to sample tree cuts  $\mathcal{T}_c$  from

---

**Algorithm 2.** Treecut Sampler
 

---

**Input:** The tree hierarchy  $\mathcal{T}$  of the dataset, tree dropout rate  $\beta$

**Output:** The treecut label set  $\mathcal{Y}_{\mathcal{T}_c}$

```

// sampling  $\mathbf{p}$  for internal nodes; prune the sub-tree rooted at  $n$  if  $p_n=0$ 
 $p_{n_0} \leftarrow 1$ ; // always keep the root node
└   for  $n \in \mathcal{N}^{int} \setminus n_0$  do
       $p_n \leftarrow \text{Bernoulli}(\beta)$ 
       $\mathbf{p} \leftarrow (p_{n_1^{int}}, \dots, p_{n_K^{int}})$ 
      // correct  $\mathbf{p}$  based on the dependency between internal nodes
       $\tilde{\mathbf{p}} \leftarrow \mathbf{p} \otimes \mathbb{1}[\mathbf{D}\mathbf{p} = \mathbf{D}\mathbf{1}]$ 
      // obtain blocked labels with predefined masks and the sampled  $\tilde{\mathbf{p}}$ 
       $\mathbf{b} \leftarrow \min(\mathbf{B}, 0)^T \tilde{\mathbf{p}} + \tilde{\mathbf{B}}^T (\mathbf{1} - \tilde{\mathbf{p}})$ 
      // gather available (unblocked) labels as the sampled label set
       $\mathcal{Y}_{\mathcal{T}_c} \leftarrow \{n_j : n_j \in \mathcal{N} \setminus n_0, \mathbf{b}_j = 0\}$ 
      return  $\mathcal{Y}_{\mathcal{T}_c}$ 

```

---

$\mathcal{T}$ . It starts by sampling a vector  $\mathbf{p} = (p_{n_1^{int}}, \dots, p_{n_K^{int}})$ , where  $n_i^{int}$  denotes the  $i$ -th internal node and  $K = |\mathcal{N}^{int}|$ , containing pruning flags  $p_n$  for all internal nodes  $n \in \mathcal{N}^{int}$ . The next step is to enforce consistency between these flags, according to the tree structure. If any node in  $\mathcal{A}(n)$  is pruned, then node  $n$  should be pruned even if  $p_n = 1$ . This is efficiently enforced across all the flags by defining a dependency matrix  $\mathbf{D} \in \{0, 1\}^{K \times K}$  where  $\mathbf{D}_{ij} = \mathbb{1}[n_j^{int} \in \mathcal{A}(n_i^{int}) \cup \{n_i^{int}\}]$  indicates whether the  $i$ -th internal node  $n_i^{int}$  is a child of the  $j$ -th internal node  $n_j^{int}$ . An example is provided on the right of Fig. 5.2 for the tree on the left. The sampled flags are then corrected by computing  $\tilde{\mathbf{p}} = \mathbf{p} \otimes \mathbb{1}[\mathbf{D}\mathbf{p} = \mathbf{D}\mathbf{1}]$ , where  $\mathbf{1}$  is the vector of  $K$  ones and  $\otimes$  the Hadamard product. Note that both  $\mathbf{D}$  and  $\mathbf{D}\mathbf{1}$  are pre-computed, making the complexity of this step roughly that of one matrix-vector multiplication.

To identify the leaves of the sampled treecut ( $\mathcal{Y}_{\mathcal{T}_c} = \text{Leaf}(\mathcal{T}_c)$ ) efficiently, a mask  $\mathbf{B} \in \{0, 1, -1\}^{K \times |\mathcal{N} \setminus \{n_0\}|}$  is defined, where each row corresponds to an internal node, and the columns contain all possible labels in  $\mathcal{T}$ , i.e., all nodes except the root  $n_0$ . Entry  $B_{ij}$  flags that  $n_j$  cannot

appear in the sampled label set, given that  $n_i \in \mathcal{N}^{int}$  has not been pruned (i.e.,  $\tilde{p}_{n_i^{int}} = 1$ ), as follows

$$B_{ij} = \begin{cases} 1, & \text{if } n_j \in \mathcal{A}(n_i^{int}) \cup \{n_i^{int}\} \quad (n_j \text{ is an ancestor of } n_i^{int}) \\ 0, & \text{if } n_i^{int} \in \mathcal{A}(n_j) \quad (n_j \text{ is a descendant of } n_i^{int}) \\ -1, & \text{otherwise} \quad (n_j \text{ is outside of the sub-tree rooted at } n_i^{int}) \end{cases}. \quad (5.9)$$

Similarly, a matrix  $\bar{\mathbf{B}}$ , of entries  $\bar{B}_{ij} = 1 - |B_{ij}|$ , is defined to flag that  $n_j$  cannot appear in the label set, given that  $n_i \in \mathcal{N}^{int}$  has been pruned, i.e.  $\tilde{p}_{n_i^{int}} = 0$ . A mask of the nodes unavailable to the label set is then computed by accumulating the masks corresponding to the values of  $\tilde{\mathbf{p}}$ ,

$$\mathbf{b} = \min(\mathbf{B}, 0)^T \tilde{\mathbf{p}} + \bar{\mathbf{B}}^T (\mathbf{1} - \tilde{\mathbf{p}}), \quad (5.10)$$

where the mask in  $\min(\mathbf{B}, 0)$  is selected if  $\tilde{p}_n = 1$ , and that in  $\bar{\mathbf{B}}$  if  $\tilde{p}_n = 0$ . Note that  $\min(\mathbf{B}, 0)$  clips  $B_{ij} = -1$  to 0. The mask  $\mathbf{b}$  can then be used to obtain  $\mathcal{Y}_{\mathcal{T}_c} = \text{Leaf}(\mathcal{T}_c) = \{n_j : n_j \in \mathcal{N} \setminus n_0, b_j = 0\}$ . Fig. 5.2 gives an example. When  $\tilde{\mathbf{p}} = (\tilde{p}_{n_0}, \tilde{p}_{n_1}, \tilde{p}_{n_2}) = (1, 0, 0)$ , then  $\mathbf{b} = \min(\mathbf{B}_1, 0) + \bar{\mathbf{B}}_2 + \bar{\mathbf{B}}_3 = (0, 1, 1, 2, 2, 0)$ , signaling that only  $n_1$  and  $n_6$  are available to the label set (as  $b_1, b_6 = 0$ ), resulting in  $\mathcal{Y}_{\mathcal{T}_c} = \{n_1, n_6\}$ .

**Optimization:** The overall loss used for prompt tuning is a combination of the two losses

$$\mathcal{L}(\mathbf{C}^t) = \mathcal{L}_{DTL}(\mathbf{C}^t) + \lambda \mathcal{L}_{NCL}(\mathbf{C}^t) \quad (5.11)$$

where  $\lambda$  is a hyperparameter. Note that, like previous prompting approaches, ProTeCt optimizes the learnable prompts  $\{\mathbf{c}_m\}_{m=1}^M$  while keeping the parameters of  $\Phi_{text}$ ,  $\Phi_{vis}$  frozen.

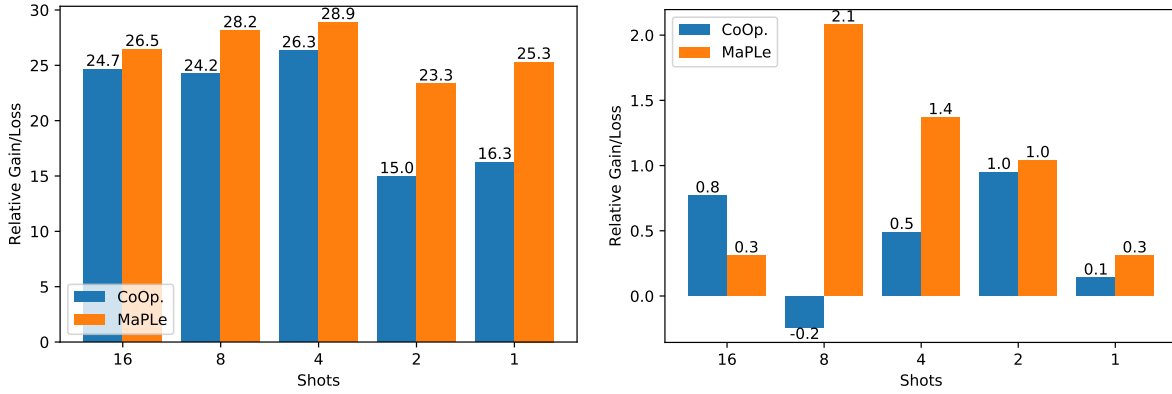
## 5.6 Experiments

In this section, we discuss experiments for evaluating the effectiveness of ProTeCt. To demonstrate that ProTeCt is a plug-an-play method, it was applied to two SOTA prompt tuning

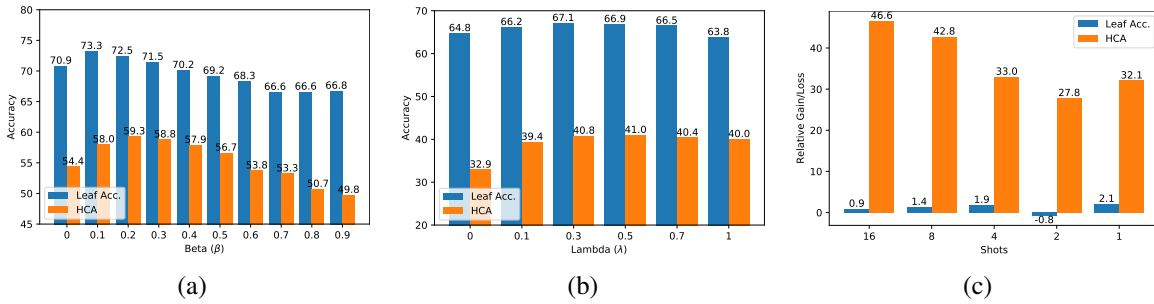


**Table 5.2.** TOS performance with/without ProTeCt on Cifar100 ( $\lambda = 0.5$ ), SUN ( $\lambda = 0.5$ ) and ImageNet ( $\lambda = 1$ ) dataset.  $\beta = 0.1$  for all datasets.

Method	K-Shot	w/ ProTeCt	Cifar100				SUN			ImageNet		
			$Acc_{leaf}$	HCA	MTA (25)	MTA (100)	$Acc_{leaf}$	HCA	MTA (25)	$Acc_{leaf}$	HCA	MTA (25)
CoOp	16		72.88	10.04	50.64	51.14	73.82	38.28	52.99	71.23	2.99	46.98
	16	✓	72.94	56.85	87.69	87.30	74.59	62.94	83.51	69.92	37.74	88.61
			(+0.06)	(+46.81)	(+37.05)	(+36.16)	(+0.77)	(+24.66)	(+30.52)	(-1.31)	(+34.75)	(+41.63)
	1		65.03	7.81	41.78	44.17	63.65	33.36	51.20	63.67	1.59	40.52
MaPLE	16		75.01	17.54	52.21	50.82	71.86	33.25	54.29	70.70	4.15	48.29
	16	✓	75.34	61.15	88.04	88.33	72.17	59.71	82.27	69.52	31.24	87.87
			(+0.33)	(+43.61)	(+35.83)	(+37.51)	(+0.31)	(+26.46)	(+27.98)	(-1.18)	(+27.09)	(+39.58)
	1		68.75	4.65	50.60	54.99	63.98	25.15	50.31	68.91	2.97	48.16
	1	✓	69.33	48.10	83.36	83.78	64.29	50.45	76.73	66.16	20.44	85.18
			(+0.58)	(+43.45)	(+32.76)	(+28.79)	(+0.31)	(+25.30)	(+26.42)	(-2.75)	(+17.47)	(+37.02)



**Figure 5.3.** Relative gain/loss after adding ProTeCt. (Left) HCA ; (Right)  $Acc_{leaf}$ .



**Figure 5.4.** Ablation of (a) tree dropout rate  $\beta$ , (b) NCL strength  $\lambda$  and (c) CLIP ViT B32 architecture.

methods: CoOp [228] and MaPLE [94]. All experiments were conducted on a single Nvidia A10 GPU, using Pytorch [152]. ProTeCt code builds on the publicly available codebases for CoOp and MaPLE and will be released upon publication.

**Metrics:**  $Acc_{leaf}$  of (5.4), HCA of (5.5) and MTA of (5.6) are considered. MTA uses 5 tree dropout rates ( $\beta \in \{0.1, 0.3, 0.5, 0.7, 0.9\}$ ) to sample treecuts of various granularities. For

**Table 5.3.** The gain of hierarchical consistency after adding ProTeCt generalizes across datasets in unseen domains. All methods are fine-tuned on ImageNet and evaluated on its 4 variants.

Method	K-Shot	w/ ProTeCt	ImageNetv2 [161]			ImageNet-S [192]			ImageNet-A [75]			ImageNet-R [71]		
			$Acc_{leaf}$	HCA	MTA (25)	$Acc_{leaf}$	HCA	MTA (25)	$Acc_{leaf}$	HCA	MTA (25)	$Acc_{leaf}$	HCA	MTA (25)
CoOp	16		64.01	2.31	43.74	47.82	1.39	38.58	50.28	2.97	52.56	75.83	18.49	64.13
	16	✓	62.60	32.84	86.66	46.80	20.73	82.60	49.08	22.45	78.21	74.94	31.18	75.59
			(-1.41)	(+30.53)	(+42.92)	(-1.02)	(+19.34)	(+44.02)	(-1.20)	(+19.48)	(+25.65)	(-0.89)	(+12.69)	(+11.40)
	1		56.43	1.51	38.27	41.38	1.11	33.61	45.92	1.76	47.54	69.84	11.74	55.31
MaPLe	16		64.15	1.97	45.93	48.97	1.58	43.37	50.61	2.31	54.88	76.61	20.67	63.06
	16	✓	62.77	27.86	86.14	47.47	17.77	82.52	47.41	19.75	77.46	75.70	32.58	77.99
			(-1.38)	(+25.89)	(+40.21)	(-1.50)	(+16.19)	(+39.15)	(-3.20)	(+17.44)	(+22.58)	(-0.91)	(+11.91)	(+14.93)
	1		61.78	2.18	45.50	46.79	1.70	45.26	47.55	3.52	55.48	74.55	18.85	62.48
MaPLe	1	✓	59.14	17.89	83.27	44.92	11.24	79.94	47.15	16.03	76.81	74.60	25.20	75.72
			(-2.64)	(+15.71)	(+37.77)	(-1.87)	(+9.54)	(+34.68)	(-0.40)	(+12.51)	(+21.33)	(+0.05)	(+6.35)	(+13.24)



(a): [Boxer, Person]

(b): [Trolleybus, Animal]

(c): [Lion, Koala]

(d): [Cheeseburger, Ice cream]

**Figure 5.5.** ProTeCt correctly predicts examples from ImageNet (a,b) and its variants (c,d) at all levels. [GT, Prediction] shows the groundtruth and incorrect prediction by vanilla prompt tuning.

each  $\beta$ ,  $T$  treecuts are sampled without repetition to obtain a total of  $5T$  treecuts.  $MTA(5T)$  indicates the result is averaged over these  $5T$  treecuts. We ablate  $T = 5$  and  $T = 20$  on Cifar100 and use  $T = 5$  for all datasets by default.

**Training Details:** All vanilla prompt tuning and their ProTeCt counterparts are trained under the same setting. The following configuration is used unless noted. All experiments use SGD optimizer and the learning rate is set to 0.02 with a cosine learning rate scheduler. By default, a pretrained ViT-B/16 CLIP model is used as initialization. For Cifar100 and SUN, we train both CoOp and MaPLe prompts for 200 epochs, using a batch size of 128 and 32, respectively. For ImageNet, CoOp is trained for 30 epochs with a batch size of 8, while MaPLe is trained for 10 epochs with a batch size of 2. Note that the setting is slightly different from the original paper due to our GPU availability.

### 5.6.1 TOS Classification Performance

Table 5.2 shows that vanilla CoOp and MaPLe have reasonable leaf accuracy for both 1-shot and 16-shot classification on Cifar100, SUN, and ImageNet. However, their very low HCA shows that their predictions are not consistent over the class hierarchy. As a result, their TOS classification performance (MTA) is much weaker than their leaf accuracy. For example, 16-shot classification with CoOp on ImageNet has a leaf accuracy of 71.23, but expected TOS accuracy of 46.98. This is explained by the very low HCA of 2.99. Similar observations hold for different few-shot configurations. In all cases, ProTeCt (results on rows with a checkmark) significantly improves HCA and MTA(25). For example, it boosts the HCA of 16-shot classification with CoOp on ImageNet by 34.75 (2.99 vs 37.74), leading to an increase of MTA(25) of 41.63 (46.98 to 88.61).

Note that, in all cases, MTA(25) after ProTeCt training is *higher* than leaf accuracy. This is expected for a well-calibrated classifier, since decisions at intermediate levels of the tree are coarser-grained than those at the leaves, which can require very fine class distinctions. These results show that ProTeCt robustifies the model for use in the TOS classification setting. The table also shows that ProTeCt maintains leaf accuracies comparable to those of the vanilla methods. Furthermore, the MTA results when 25 and 100 treecuts are sampled (corresponding to  $T = 5$  and  $T = 20$ ), are compared on Cifar100. It can be seen that the performances are similar, showing that sampling 25 treecuts is sufficient to achieve good estimation. Fig. 5.3 compares the **relative** gains in HCA and leaf accuracy of training with ProTeCt, as compared to vanilla prompt tuning. These gains are shown for both CoOp and MaPLe, under several few shot configurations, on SUN dataset. In all cases, ProTeCt increases HCA by more than 15 points, while maintaining a leaf accuracy comparable to that of vanilla CoOp/MaPLe.

**Table 5.4.** Comparison of CoOp with/without ProTeCt on FGVC Aircraft [131] dataset.

K-shot	w/ ProTeCt	$Acc_{leaf}$	HCA	MTA (25)
16		41.88	17.82	21.11
16	✓	42.00 (+0.12)	29.94 (+12.12)	32.95 (+11.84)
1		23.61	11.55	16.77
1	✓	27.30 (+3.69)	16.47 (+4.92)	24.67 (+7.90)

### 5.6.2 Domain Generalization of TOS Classification

Following the domain generalization setting of [228, 227, 94, 218], we investigate whether TOS classification performance generalizes across datasets. The CLIP model with ProTeCt prompts trained on ImageNet (source) is applied to 4 ImageNet variants (target) with domain shift: ImageNetv2 [161], ImageNet-Sketch [192], ImageNet-A [75] and ImageNet-R [71]. Table 5.3 summarizes the three metrics on these datasets for CoOp and MaPLe. Similarly to Table 5.2, ProTeCt enables significant gains in HCA and MTA(25) over the baselines for all datasets. Note that since ImageNet-A and ImageNet-R only contain 200 ImageNet subclasses, their hierarchy is different from that of ImageNet. These results demonstrate the flexibility and robustness of ProTeCt, even when transferring the model to a target domain whose class hierarchy is different from that of the source domain.

### 5.6.3 Ablation Study and Visualization

In this section, we discuss ablations of ProTeCt components and visualize the predictions of different models.

**Tree Dropout Rate  $\beta$ :** Fig. 5.4 (a) plots Cifar100  $Acc_{leaf}$  and HCA as a function of the drop-out rate  $\beta$ , for 16-shot CoOp + ProTeCt training ( $\lambda = 1$ ). Larger values of  $\beta$  reduce the likelihood of sampling the leaf nodes of the tree, resulting in shorter trees and weaker regularization. Hence, both leaf accuracy and HCA degrade for large  $\beta$ . However, always using

**Table 5.5.** CoOp ablation on Cifar100 dataset. Both DTL and NCL loss improve the hierarchical consistency.

DTL	NCL	16-shot			1-shot		
		$Acc_{Leaf}$	HCA	MTA (25)	$Acc_{Leaf}$	HCA	MTA (25)
		72.88	10.04	50.64	65.03	7.81	41.78
✓		72.81	47.97	87.32	64.77	32.93	81.38
	✓	64.20	51.69	79.44	61.22	38.02	62.16
✓	✓	<b>72.94</b>	<b>56.85</b>	<b>87.69</b>	<b>66.88</b>	<b>41.01</b>	<b>81.64</b>

the full tree ( $\beta = 0$ ) also achieves sub-optimal results. The two metrics peak at  $\beta = 0.1$  and  $\beta = 0.2$ , respectively.  $\beta = 0.1$  is selected for all experiments.

**NCL strength  $\lambda$ :** Fig. 5.4(b) summarizes the Cifar100 performance of 1-shot classification with CoOp and  $\beta = 0.1$ , as a function of the hyperparameter  $\lambda$  that balances the two losses of (5.11). The introduction of NCL improves leaf accuracy/HCA from 64.8/32.9 ( $\lambda = 0$ ) to 66.9/41 ( $\lambda = 0.5$ ). We adopt  $\lambda = 0.5$  for CIFAR100 and SUN. For ImageNet,  $\lambda = 0.5$  and  $\lambda = 1$  have comparable performance.

**Architecture:** Fig. 5.4 (c) shows the plug-and-play properties of ProTeCt, by showing that the gains obtained for CoOp+ProTeCt in Fig. 5.3 with CLIP ViT B16 also hold for ViT B32 features.

**Taxonomies:** To investigate the robustness of ProTeCt across hierachies, we consider the FGVC Aircraft [131] dataset. This has its built-in hierarchy, which beyond differing from those of SUN [209] and WordNet [136], is a technical hierarchy of fine-grained aircraft classes. Table 5.4 summarizes the CoOp results for these experiments, showing that ProTeCt improves performance under all metrics. This illustrates its taxonomy robustness.

**Loss:** Table 5.5 ablates the model performance with/without NCL and DTL loss, for two few-shot settings, using CoOp on Cifar100. Both losses improve TOS performance individually and there is a significant additional gain when they are combined. Using NCL alone can degrade leaf performance, due to the lack of regularization across different levels of the hierarchy. The combination of the two losses overcomes this problem.

**Table 5.6.** Improving other adapter-based tuning methods, including CLIP-Adapter and CLIP+LORA with ProTeCt on Cifar100.

K-Shot	w/ ProTeCt	CLIP-Adapter [48]			CLIP+LORA [40]		
		$Acc_{leaf}$	HCA	MTA (25)	$Acc_{leaf}$	HCA	MTA (25)
16		71.96	5.59	42.93	70.45	4.57	47.19
16	✓	72.47 (+0.51)	57.15 (+51.56)	87.67 (+44.83)	70.64 (+0.19)	51.06 (+46.49)	77.29 (+30.10)
1		65.35	8.35	48.25	63.57	2.89	38.63
1	✓	67.29 (+1.94)	36.21 (+27.86)	78.49 (+30.24)	63.62 (+0.05)	24.66 (+21.8)	56.42 (+17.79)

**Adapter-based tuning methods:** To investigate how ProTeCt affect the TOS performance of adapter-based tuning method on FM, we use the ProTeCt losses to train the CLIP adapter of [48] and the CLIP+LORA method of [40]. Table 5.6 shows that this again produces large consistency gains, indicating that ProTeCt losses generalize to both prompt-based and adapter-based methods.

**Visualization:** Fig. 5.5 shows examples from ImageNet (a,b), ImageNet-A (c) and ImageNet-R (d). While ProTeCt can correctly classify these examples at all hierarchy levels, vanilla prompt tuning fails at certain levels.

## 5.7 Conclusion

In this work, we formulated the TOS classification setting, including datasets, performance metrics, and experiments. For a given dataset, a class hierarchy is built by assigning the dataset classes to leaf nodes and superclasses to internal nodes. The TOS classifier is then expected to support classification with label sets drawn throughout the taxonomy. We have shown that existing prompting methods fail to address this setting and proposed ProTeCt training for enhancing the TOS performance of FMs, like CLIP, with existing prompt tuning methods. ProTeCt includes two losses. A dynamic treecut loss, based on an efficient treecut sampler, dynamically regularizes labels of varying granularity. A node-centric loss encourages correct predictions at all hierarchy levels. Experiments show that ProTeCt enhances the TOS perfor-

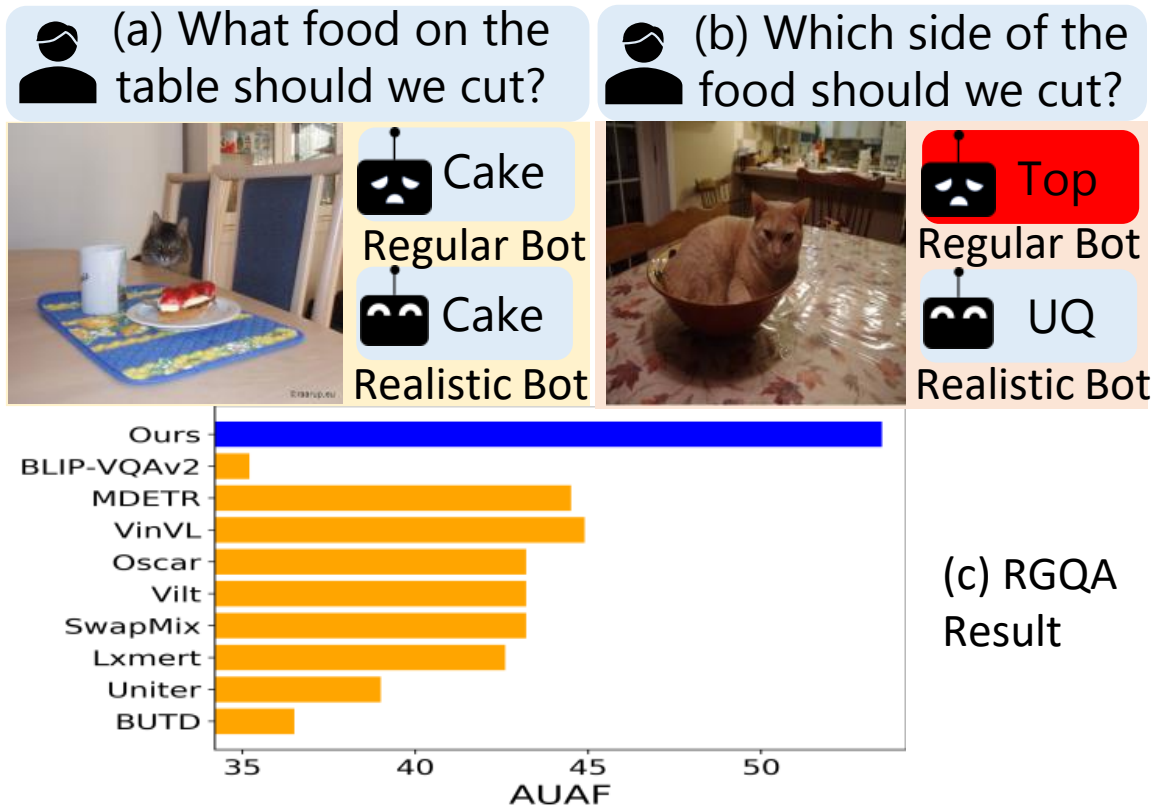
mance of existing prompt tuning techniques, both in the adaptation datasets and across unseen domains. Finally, it was shown that ProTeCt is successful for various architectures, hierarchies and parameter tuning methods.

Chapter 5 is, in full, based on the material as they appear in the publication of “ProTeCt: Prompt Tuning for Hierarchical Consistency”, Chih-Hui Ho\*, Tz-Ying Wu\*, and Nuno Vasconcelos, In Proceedings of IEEE Conference on Computer Vision and Pattern Recognition xiii (CVPR), 2024 . The dissertation author was the primary investigator and author of this paper.

## **Chapter 6**

# **Toward Unsupervised Realistic Visual Question Answering**





**Figure 6.1.** Realistic VQA. In VQA, a vision system answers a question by inspection of an image. However, existing approaches have no awareness if the question is *answerable* (AQ), such as in (a), or *unanswerable* (UQ), as in (b). A realistic VQA system only answers AQs. (c) RVQA performance of prior (yellow) vs. proposed (blue) models.

## 6.1 Introduction

Visual Question Answering (VQA) is a challenging task that requires a machine to understand a question in natural language, perceive an image, and produce an answer. Despite extensive research in VQA [7, 57, 79, 133, 87, 181, 25, 221], little attention has been given to VQA robustness. In this work, we consider robustness to *unanswerable questions* (UQs), which cannot be answered by image inspection, as in Fig. 6.1(b). This is opposed to the traditional answerable questions (AQ), such as in Fig. 6.1(a).

Lack of robustness to UQs is problematic because, in the absence of image information, the VQA system frequently resorts to the answer statistically most correlated with the question.

In the figure, the absence of food in (b) entices the robot to pick the answer corresponding to the “side of food” most commonly “cut” in the dataset, which happens to be the “top” (perhaps because the dataset is rich in cake images). The problem is that a decision by the robot to act on this answer would be catastrophic for the cat in the scene. More generally, the inability to reject UQs signals a deeper perceptual deficiency and exposes VQA systems to attacks.

Vulnerability to UQs can create safety hazards for indoor robots [6] or assistants for the visually impaired [59] and reduces user trust in VQA models. When faced with a UQ, the VQA system should refuse to answer or ask for more information. More precisely, it should assess the question, decide to (a) “accept” or “reject” it, and only (b) answer the accepted questions. Since this resembles the idea of a “realistic model” for classification [205, 197], we denote it *realistic VQA* (RVQA).

Although some prior works have addressed RVQA, existing formulations are not conducive to practical RVQA systems, for three reasons. First, existing formulations address the *supervised* training of RVQA models. This, however, requires a significant number of annotated UQs [160, 130, 59]. The collection of a set of annotated UQs large enough to train a modern VQA network is expensive, frequently not even plausible. This is compounded by the existence of many types of UQs: training on one type does not guarantee generalization to another. Second, prior datasets generate UQs by randomly paring images and questions from an existing VQA dataset [160, 130, 89, 184]. This, however, tends to produce obviously unrelated pairs of images and questions with low semantic similarity, that are easy to reject. In the real world, RVQA models must be able to handle both simple and challenging UQs. Finally, the VQA datasets that support RVQA, such as VizWiz [59], are designed for a specific application domain, frequently containing images with few objects. This prevents the modeling of complex image-question relationships.

To address these drawbacks, we consider the problem of *unsupervised RVQA*. We start by curating a new evaluation dataset for this task, based on *testdev* set of the widely used GQA dataset [79]. The new dataset, denoted as *realistic GQA* (RGQA), is composed of 26,591 AQs in

the *testdev* set of GQA and 29,046 additional human-annotated UQs. To penalize RVQA models that overfit on a specific type of UQs, we generate candidate UQ by two methods. *CLIP-based* UQ generation produces candidate UQs by retrieving questions sorted by CLIP [158] similarity score between image and question. *Perturbation-based* (PT-based) UQ generation perturbs the object, attribute, and relation phrases in a question. For each method, we further generate a set of easy and a set of hard candidate UQs, leading to a total of four RGQA subsets. All candidate UQs are finally annotated by humans, to guarantee they are unanswerable.

Since each AQ in RGQA is complemented by its answer, the dataset enables measuring the accuracy of both AQ/UQ detection and VQA accuracy. For this, we propose the ACC-FPR curve [36], a joint measure of VQA accuracy for AQs and UQ rejection performance. This is complemented by introducing 3 new unsupervised RVQA methods that establish a set of baselines for future RVQA work. These are classifiers with a binary output per class, which elicit a rejection when all class outputs are below a threshold. Three methods differ in training strategy and are shown capable of producing RVQA models that both reject UQs and answer AQs correctly, outperforming prior RVQA methods.

The first is to train the classifier with pseudo UQs, obtained by randomly pairing images and questions. This suffers from the fact that pseudo UQs are noisy and not always challenging. The second improves the sampling of image-question pairs, by using a RoI Mixup strategy to encourage the model to spot fine-grained mismatches between image and question during training. The third address the limitations of random sampling at the classifier output, by ensembling multiple RVQA models. Experiments show that all strategies enhance RVQA performance and that they can be combined to achieve best results. As shown in Fig. 6.1(c), this combination (blue) significantly exceeds the performance of existing VQA models (yellow) under the joint objective of rejecting UQs and correctly answering AQs.

Overall, three contributions are made to VQA. First, we introduce RGQA, a new challenging testing dataset for evaluating RVQA. It contains both fine- and coarse-grained image-question pairs which better align with real-world scenarios than previous datasets. Second, we propose



**Figure 6.2.** Examples of CLIP based (a,b,e,f) and Perturbation (PT) based UQs (c,d,g,h) in RGQA. For the PT-based UQs, the red words are modified from the original question.

an unsupervised training strategy that uses free pseudo UQs, combining random sampling, RoI Mixup, and model ensembling. Finally, extensive experiments demonstrate the effectiveness of the proposed methods over prior RVQA methods. We also show that the proposed models under-perform humans, which encourages future work in the RVQA problem.

## 6.2 Related Work

In this section, we review related works.

**Realistic VQA (RVQA):** The study of RVQA is still in its infancy. A central question is how to assemble datasets of UQs, i.e. unrelated pairs of images and questions. Most methods start from a VQA dataset. VTFQ [160] collected a RVQA dataset by randomly pairing images and questions. QRPE [130] uses question-derived object/attribute premises. The associated image is then replaced by its Euclidean nearest neighbor in a set of images without the extracted premises. These approaches are limited by the inability of random pairing or Euclidean similarity

to guarantee a fine-grained semantic mismatch between image and UQ.

VizWiz [59] is a VQA dataset from the visually impaired setting, with UQs asked by people. However, its images are of poor quality and contain one or a few objects, which prevents complex interaction between objects, scenes, and language. TDIUC [89] and C2VQA [183] are created by checking if objects mentioned in questions also appear in images. While UQ cardinality can be easily scaled up [89] by randomly paring images and questions without common objects, this assumes that the only reason for a UQ is object mismatch. In comparison, the proposed RGQA dataset considers both coarse- and fine-grained mismatches, based on stronger measures of image-question similarity. No constraints of image content are also imposed on UQ generation, producing a more challenging and diverse dataset.

All previous works address supervised RVQA, using annotated UQs, which is expensive and limits dataset sizes. For instance, [160] generates a caption per image with NeuralTalk2 [93] and measures question-caption similarity with a binary LSTM classifier. [130] further extracts the question premise and uses the concatenated question-premise-caption triplet as classifier input. [116] uses this architecture to reject UQs in VQA. [109] uses the maximum attention score between objects and text tokens for rejection and regularizes attentions by training on UQs. In this work, we explore an unsupervised training strategy that is model-agnostic and does not rely on annotated UQs.

**Out of Distribution Detection (OOD)** RVQA is closely related to OOD in classification [72, 105, 121, 74, 203, 73, 113] which aims to detect samples on which a classifier has not been trained. This has been addressed by temperature scaling of classifier logits [121], using Mahalanobis distance [113] or energy scores [122] to measure the distance to the training distribution, ensembling predictions from multiple models [105, 188], or regularizing in-distribution (ID) features [26]. It is also possible to use a background dataset, with different distribution from the training dataset, during training [37, 74, 203, 119]. While background datasets can significantly improve OOD, prior works in RVQA [116, 109] show a performance degradation for AQs. We devise sampling strategies that address this problem.

**Table 6.1.** Comparison to previous datasets. The proposed RGQA dataset has longer and more fine-grain UQs and requires a multi-task classifier to solve the RVQA problem. RGQA is only for evaluation purposes.

Dataset	Supervised UQ	Type	UQ Annotation	Image Source	Question Source	UQ(%)	# Test Pair	Avg. Length
VTFQ [160]	✓	UQ det.	human	MSCOCO	VQAv1	89.24	31464	7.53
QRPE [130]	✓	UQ det.	generated	MSCOCO	VQAv1	50.87	35476	7.76
C2VQA [184]	✓	UQ det.	generated	Visual Genome	Visual Genome	50.00	29106	7.10
TDIUC [89]	✓	VQA+UQ det.	generated	MSCOCO+Visual Genome	VQAv1+Visual Genome	22.17	538868	7.92
VizWiz [59]	✓	VQA+UQ det.	human	VizWiz	VizWiz	27.84	8000	8.10
RGQA	✗	VQA+UQ det.	human	GQA <i>testdev</i>	GQA <i>testdev</i>	52.22	55637	10.33

The classification and OOD performance are usually reported by combining Area Under ROC curve (AUROC) and accuracy on ID samples [148, 10, 226, 216]. However, separate metrics increase the difficulty to compare models. We introduce a unified metric for the RVQA problem.

## 6.3 RGQA Dataset

In this section, we introduce the RGQA dataset for evaluating RVQA systems. It is a human-annotated dataset with  $\sim 29K$  UQs and built upon the *testdev* set of GQA [79].

### 6.3.1 Dataset Curation

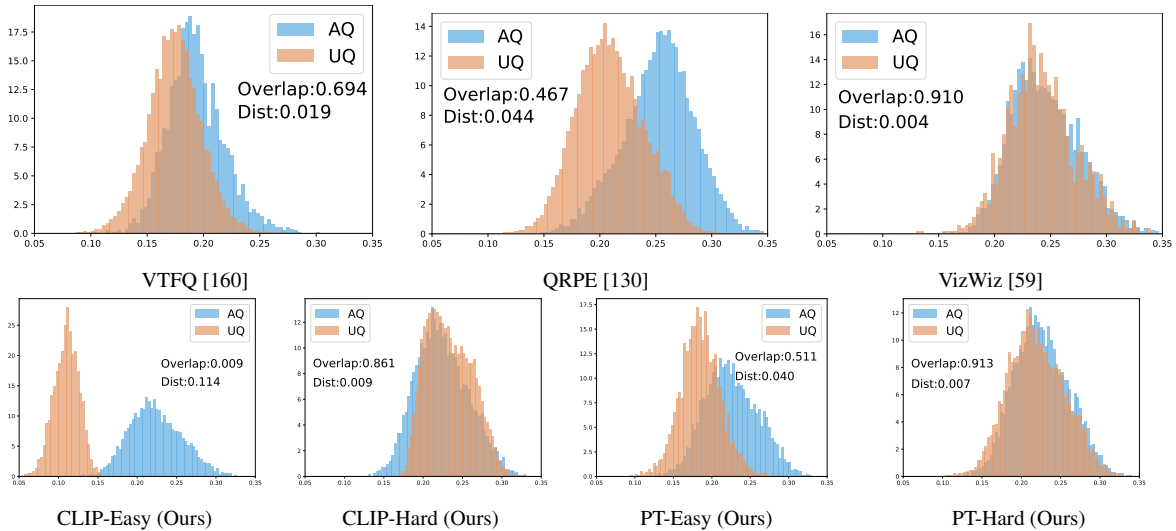
RGQA has a balanced coverage of AQs and UQs. AQs are image-question pairs with answers from the GQA *testdev* set. For UQs, we first generate a *candidate set* using two different approaches, *CLIP-based* and *Perturbation-based*, to mitigate potential UQ generation biases. Human annotators then decide which candidates are true UQs.

**CLIP-based Candidate UQs:** Leveraging recent advances in image-text pre-training, we use CLIP [158] to measure similarity between images and questions. Given an image  $I$ , we consider the set of questions  $\mathcal{Q}(I)$  in the *testdev* dataset, excluding 1) existence questions (e.g. “Are there any ...?”), which can never be UQs, and 2) the questions originally paired with  $I$ . We then feed all pairs  $(I, Q), Q \in \mathcal{Q}(I)$  to the CLIP model and rank the questions by similarity score. To cover the spectrum from simple to hard UQs, 85 questions sampled from the top 2,500 are used as candidate UQs for CLIP-Hard, while the last 50 questions are used as candidate UQs for CLIP-Easy. Fig. 6.2 shows images from each set. The pairs of CLIP-Hard (Fig. 6.2 (a,b)) have more subtle mismatches than those of CLIP-Easy (Fig. 6.2 (e,f)).

**Perturbation-based Candidate UQs:** Given an AQ in GQA *testdev*, a candidate UQ counterpart is generated by perturbing its objects and adjectives. This is implemented by first collecting a set of candidate objects and their attributes from the scene graphs of GQA *train* and *valid* set. For each AQ, objects and adjectives are extracted by POS tagging. Similar to the CLIP-based approach, both easy and hard UQs are generated by the perturbation-based approach, resulting in the subsets PT-Easy and PT-Hard. For PT-Easy, each object in the AQ is replaced by a random but different object sampled from the candidate object set. For PT-Hard, the objects in AQ are kept but their attributes are replaced by different candidate attributes of the same object. Finally, the spatial relation terms in PT-Hard are replaced by antonyms, such as “left/right” and “top/bottom”. Conflicting questions, like “What color are the black shoes?” are then eliminated. Fig. 6.2(g,h) and Fig. 6.2(c,d) show examples from PT-Easy and PT-Hard, with the perturbed text in red.

**Human Annotation:** Human annotators analyze all image-question candidates and decide which are true UQs. Following [182, 64, 106, 20], we use 8 expert annotators with experience in visual language research. The annotator is shown an image and two questions, and asked to choose from “*valid*” (corresponding to AQs) and “*invalid*” (UQs) options for each question. We instruct the annotator to choose “*valid*” if the decision is ambiguous, due to unclear images, confusing wording, or any other reason. These annotations are discarded.

This process produced 11,264 UQs for CLIP-Hard, 5,689 for CLIP-Easy, 6,130 for PT-Easy and 5,963 for PT-Hard. The next step aimed to sample a similar number of AQs, to balance the dataset. For CLIP-Hard and CLIP-Easy, we randomly sample AQs to pair with UQs. For each UQ, we consider the associated image and retrieve the AQs originally paired with this image in GQA. We then randomly sample one of these AQs. This produced 11,158 questions for CLIP-Easy and 20,325 for CLIP-Hard. For PT-Easy and PT-Hard, we pair with the original AQs for each perturbed UQ which results in 12,241 questions in total for PT-Easy and 11,913 for PT-Hard.



**Figure 6.3.** CLIP image-question similarity distribution of both AQs and UQs. The overlap area between 2 normalized histograms (sum of overall area=1) and the distance between the means are computed.

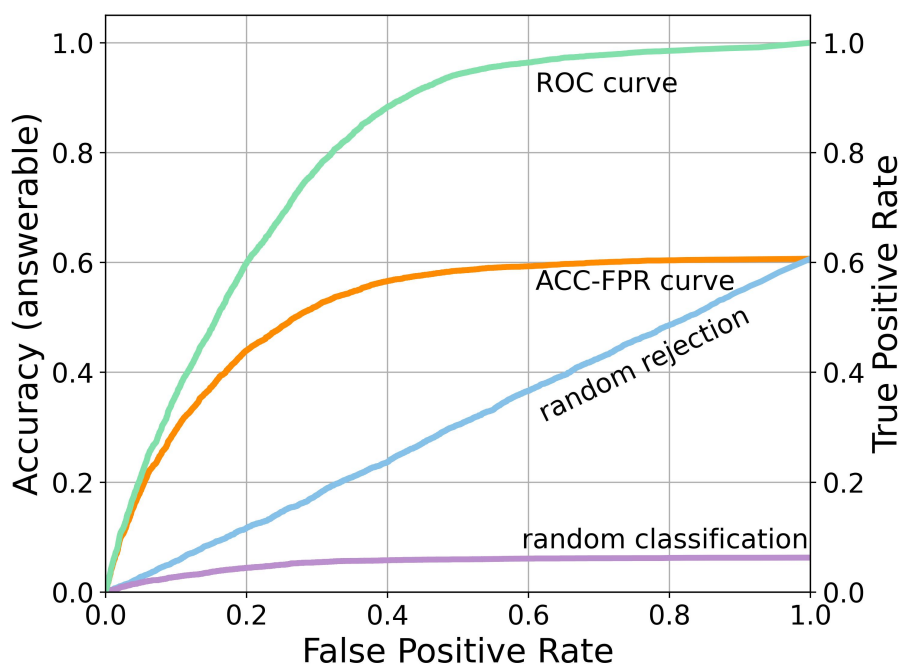
### 6.3.2 Dataset Analysis

**UQ Categories:** RGQA covers a wide spectrum of UQs, including questions without valid answers (*e.g.* Fig. 6.2 (b)), with false premise at object (*e.g.* Fig. 6.2 (e)) or attribute level (*e.g.* Fig. 6.2 (d)), and underspecified questions (*e.g.* “Do the snowpants look black and long?” for Fig. 6.2 (f)). Many UQs also have subtle mismatches with the image, which can only be spotted via high-level understanding of image semantics. For instance, in Fig. 6.2(b), both the predicate “wearing” and the object “shoes” exist in the image, so the model needs to understand the semantics of “wearing” and search for their subject. Hence, beyond evaluating robustness, RGQA also measures how strongly VQA models learn semantics.

**Dataset Comparison:** Table 6.1 compares RGQA to previous VQA datasets with UQs [160, 130, 59, 184, 89]. Several of these only address UQ detection. RGQA combines this with VQA, which better matches real-world applications. It also contains higher-quality human annotations, a better balance between AQs and UQs, and longer and more complex questions (last column) than previous datasets. Overall, it poses a greater challenge to model reasoning skills.

**AQs vs UQs:** To gain insight on the differences between AQs and UQs, we performed an





**Figure 6.4.** Comparison between ROC curve (green; right axis) and ACC-FPR curve (orange; left axis). See text for details.

analysis from two aspects. The first is to plot the distributions of image-question CLIP similarity scores, as shown in Fig. 6.3. Clearly, for VTFQ [160] and QRPE [130] the scores are smaller, indicating simpler questions, and the AQ/UQ distributions have less overlap, showing that they can be easily separated. VizWiz [59], CLIP-Hard, and PT-Hard have larger scores and stronger overlap between the two distributions, indicating that their UQs have finer-grained mismatch between image and question. However, while the CLIP score measures semantic similarity, it does not capture the answerability of UQs. The second strategy addresses this limitation, by plotting the distribution of questions by the first three words. Other than a different order for the three most popular words (“Are”, “Who” and “Which”) and a few changes on the proportions, there are no major differences between the AQ and UQ distributions. This shows that AQs/UQs cannot be easily separated by question structure.

### 6.3.3 Evaluation Metrics

Since UQ detection is an OOD problem, we leverage well-established OOD practices for evaluation. However, because RVQA requires jointly solving UQ detection and VQA, the common OOD practice of reporting close-set accuracy and AUROC is not satisfying. We instead proposed to use the ACC-FPR curve, introduced as CCR-FPR curve in [36], which measures the joint performance. Given a VQA classifier  $f$  and a UQ detector  $g$ , ACC is the proportion of AQs with correct VQA prediction and accepted as AQ, i.e.

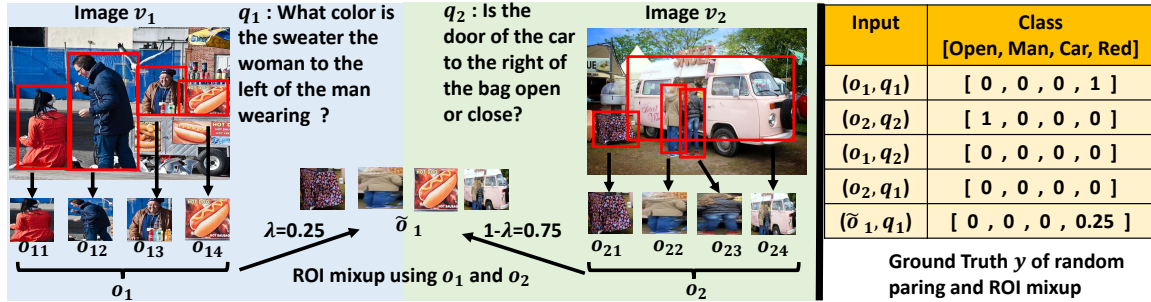
$$\text{ACC} = \frac{|\{x_i | f(x_i) = a_i, g(x_i) = \text{AQ}, (x_i, a_i) \in \mathcal{D}^{aq}\}|}{|\mathcal{D}^{aq}|}, \quad (6.1)$$

where  $x_i = (v_i, q_i)$  denotes image-question pair,  $a_i$  is the corresponding VQA answer and  $\mathcal{D}^{aq}$  is the dataset of AQs. FPR is the proportion of UQs falsely accepted as AQ, i.e.

$$\text{FPR} = \frac{|\{x_i | g(x_i) = \text{AQ}, x_i \in \mathcal{D}^{uq}\}|}{|\mathcal{D}^{uq}|}, \quad (6.2)$$

where  $\mathcal{D}^{uq}$  is the dataset of UQs. The ACC-FPR curve is drawn by connecting ACCs (y-axis) at different FPRs (x-axis) as in Fig. 6.4. We define the maximum value of the curve on the y axis (best accuracy the model can achieve on AQs) as full accuracy (FACC).

A RVQA model with a strong VQA classifier  $f$  and a UQ detector  $g$  (orange line) has higher FACC than a model with the same  $g$  but random  $f$  (purple line). On the other hand, a model with the same  $f$  but random  $g$  (blue line) has the same FACC but underperforms the RVQA model for all FPRs less than 1. Note that the ROC curve (green line) is the special case of ACC-FPR curve with FACC=1. As a single evaluation metric, we use *Area Under* ACC-FPR curve (AUAF), for joint performance, FPR at 95% FACC (FF95) for rejection, and FACC for classification.



**Figure 6.5.** Illustration of the pseudo UQ and ROI Mixup. The right table shows the label for different visual question inputs.

## 6.4 Unsupervised RVQA Learning

In this section, we introduce unsupervised RVQA and three model-agnostic methods for unsupervised training.

### 6.4.1 Unsupervised RVQA

Unsupervised RVQA learns a model, VQA classifier  $f$  and UQ detector  $g$ , from a dataset of AQs  $\mathcal{D}_{tr}^{aq} = \{(x_i, a_i)\}$ , *without* annotated UQs. At testing,  $g(x)$  decides whether a pair  $x$  is accepted. If so,  $f(x)$  then predicts an answer.

### 6.4.2 Training with Pseudo UQ

Inspired by recent OOD works using an auxiliary background dataset [36, 74, 119, 137] for training, we investigate training the RVQA model with a background dataset. For image classification, choosing a background dataset of reasonable scale and effective performance is non-trivial [119]. However, this is much simpler for RVQA: a simple and natural choice is to randomly pair images and question  $\{(v_i, q_i)\}$  already available in the VQA dataset. Given an image  $v_i$ , we randomly sample a question  $q_k$  belonging to a different image  $v_k \neq v_i$  to form a *pseudo* unanswerable image-question pair  $(v_i, q_k)$ . Fig. 6.5 illustrates an example of this random paring, where image  $v_1$  is paired with question  $q_2$ . Like this example, most randomly sampled pairs are

unanswerable<sup>1</sup>. The *pseudo UQs* are used to construct the unsupervised background dataset  $\widehat{\mathcal{D}}_{tr}^{uq}$ .

With  $\mathcal{D}_{tr}^{aq}$  and  $\widehat{\mathcal{D}}_{tr}^{uq}$ , the VQA classifier  $f$  and binary UQ detector  $g$  can be trained to minimize the risk

$$\begin{aligned} \mathcal{R} &= E_{(x_i, a_i) \in \mathcal{D}_{tr}^{aq}} \mathbb{1}(f(x_i) \neq a_i) \\ &+ E_{x_i \in \mathcal{D}_{tr}^{aq}} \mathbb{1}(g(x_i) \neq \text{AQ}) + E_{x_i \in \widehat{\mathcal{D}}_{tr}^{uq}} \mathbb{1}(g(x_i) \neq \text{UQ}), \end{aligned} \quad (6.3)$$

where the first term is the classification error and the last two are the detection error. Different from most OOD methods, which use softmax outputs [74], VQA models are usually trained as multi-label models. Let  $\mathcal{Y} = \{1, \dots, K\}$  be the set of possible answers. Then, the ground truth for  $i^{\text{th}}$  example  $x_i = (v_i, q_i)$  and  $k^{\text{th}}$  answer is a binary variable,  $y_{i,k} \in \{0, 1\}$ , with  $y_{i,k} = 1$  if the answer holds for  $x_i$  and  $y_{i,k} = 0$  otherwise. The VQA model  $f$  has  $K$  binary outputs, where  $f_k(x)$  is the predicted probability for  $k^{\text{th}}$  answer, implemented with sigmoid functions and trained with the *binary cross entropy* (BCE) loss

$$l_i = \sum_{k=1}^K y_{i,k} \log f_k(x_i) + (1 - y_{i,k}) \log(1 - f_k(x_i)). \quad (6.4)$$

In Sec. 6.5.2, several configurations of models  $f$  and  $g$  are ablated. Best results were obtained with an *integrated* model, where both  $f$  and  $g$  share the network according to

$$g(x) = \mathbb{1}(\max_k f_k(x) > \theta) \rightarrow y^* = \operatorname{argmax}_k f_k(x), \quad (6.5)$$

where  $\rightarrow$  means that the second equation is only implemented if  $g(x) = 1$ . The rejection step first checks that there is at least one  $f_k$  above threshold  $\theta$ . If so, VQA is performed. Otherwise, the example  $x$  is identified as a UQ and rejected. This model minimizes (6.3) by simply assigning labels  $y_{i,k} = 0, \forall k \in \mathcal{Y}$  to each UQ  $x_i$ , leading to

$$\mathcal{L}^{rvqa} = \frac{1}{N_{tr}^{aq} + N_{tr}^{uq}} \sum_{i=1}^{N_{tr}^{aq} + N_{tr}^{uq}} l_i, \quad (6.6)$$

---

<sup>1</sup>We inspected 100 pairs on GQA *train* and found 77% to be UQs.

where  $N_{tr}^{aq}$ ,  $N_{tr}^{uq}$  is the size of  $\mathcal{D}_{tr}^{aq}$  and  $\widehat{\mathcal{D}}_{tr}^{uq}$ , respectively.

### 6.4.3 RoI Mixup

While random pairing is effective for constructing a background dataset of UQs, it tends to produce coarse-grained UQs, where (see Fig. 6.5) image and question are weakly related. To increase the coverage of fine-grained mismatches, we propose an additional sampling strategy denoted as *RoI Mixup*, motivated by mixup data augmentation [219, 217, 21]. Most VQA models have an object-based architecture [181, 25, 47, 118, 221], where image  $v_i$  is represented as a set of  $M$  (usually fixed) objects features  $o_i = \{o_{i,m}\}_{m=1}^M$  detected by a pre-trained object detector [162]. In training, RoI Mixup randomly replaces a portion  $1 - \lambda$ , where  $\lambda \in (0, 1)$ , of the objects in image  $v_i$  with objects from another image  $v_j \neq v_i$ . This leads to a new and mixed set of objects  $\tilde{o}_i$

$$\tilde{o}_i = \{o_{i,m}\}_{m=1}^{\lambda M} \cup \{o_{j,n}\}_{n=1}^{(1-\lambda)M} \quad (6.7)$$

with a new target one-hot vector  $\tilde{y}_i = \lambda y_i$ . Note that  $y_i$  can either be a correct answer, for AQs, or a zero vector, for UQs. Intuitively, by reducing the percentage  $\lambda$  of original objects, the probability of the question being AQ should also shrink by  $\lambda$ . Fig. 6.5 illustrates the mixing of two sets of visual features  $o_1$  and  $o_2$  with  $\lambda = 0.25$  to synthesize the object set  $\tilde{o}$ . Following [219],  $\lambda$  is sampled as  $\lambda \sim \text{Beta}(1, \beta)$  where  $\beta$  is a tunable hyper-parameter.

### 6.4.4 Model Ensembling

Random pairing and RoI Mixup are sampling strategies to create a background UQ dataset with a mix of coarse- and fine-grained UQs. It is also possible to improve the performance by regularizing the model output. As in the calibration literature [105, 188], we achieve this with model ensembles. Given  $C$  models  $\{f^c\}_{c=1}^C$ , model  $f^c$  predicts the probability of answer  $y_k$  as  $p^c(y_k = 1|x) = f_k^c(x)$ . Assuming the predictions of different models are independent, the probability predicted by the ensemble is  $p^E(y_k = 1|x) = f_c^E(x) = \prod_{c=1}^C f_k^c(x)$ . Model ensembling is then implemented by replacing  $f$  with  $f^E$  in (6.5), which produces more conservative

**Table 6.2.** RVQA comparison of recent VQA models, using MSP for the UQ detector  $g$ . \* indicates that the model is not finetuned on GQA dataset. Larger AUAF and smaller FF95 are better.

Classifiers	CLIP-Easy			CLIP-Hard			PT-Easy			PT-Hard			Avg.
	AUAF	FF95↓	FACC	AUAF	FF95↓	FACC	AUAF	FF95↓	FACC	AUAF	FF95↓	FACC	AUAF
BUTD [25]	38.45	64.75	53.50	36.13	79.14	53.08	37.83	66.05	53.02	33.60	83.11	51.31	36.50
Uniter [25]	40.03	73.15	57.08	39.42	80.48	57.10	41.45	61.76	56.82	35.17	83.52	55.08	39.01
LXMERT [181]	42.39	76.25	0.87	42.60	78.92	60.49	47.30	61.79	59.94	38.12	85.14	58.76	42.60
SwapMix [58]	46.31	71.98	61.05	42.44	78.41	60.10	46.19	62.27	59.77	37.78	82.73	58.37	43.18
Vilt [97]	46.17	69.62	58.91	40.66	79.21	57.39	48.06	60.54	60.64	37.93	82.40	57.63	43.21
Oscar [118]	45.51	72.14	62.09	41.76	80.04	61.72	46.38	64.27	<b>63.44</b>	39.16	83.15	60.20	43.2
VinVL [221]	49.86	69.87	<b>64.89</b>	46.36	78.16	<b>64.61</b>	41.68	84.27	63.38	41.67	84.26	<b>63.37</b>	44.89
MDETR [90]	47.81	70.32	62.91	43.86	78.94	62.05	47.14	70.04	62.93	39.04	84.11	60.30	44.46
BLIP-VQAv2* [115]	35.93	69.39	51.67	34.94	82.10	51.13	37.44	69.33	52.49	32.62	86.91	49.79	35.23

**Table 6.3.** Comparison between different RVQA approaches on AUAF. Cells with light cyan background denote training with pseudo UQs.

RVQA Approaches	BUTD [5]					UNITER [25]					LXMERT [181]				
	CLIP easy	CLIP hard	PT easy	PT hard	Avg.	CLIP easy	CLIP hard	PT easy	PT hard	Avg.	CLIP easy	CLIP hard	PT easy	PT hard	Avg.
FRCNN	33.58	30.73	31.43	26.94	30.67	35.81	33.09	33.67	28.82	32.84	38.43	35.22	35.73	31.00	35.09
MSP	38.45	36.13	37.83	33.60	36.50	40.03	39.42	41.45	35.17	39.01	42.39	42.60	47.30	38.12	42.60
ODIN	38.47	36.14	37.80	33.60	36.50	40.04	39.43	41.45	35.16	39.02	42.41	42.59	47.33	38.12	42.61
Maha	30.05	25.75	25.34	23.93	26.26	37.52	33.74	35.87	31.68	34.70	57.68	44.96	49.44	39.25	47.83
Energy	38.47	36.19	37.77	33.67	36.52	40.10	39.42	41.41	35.19	39.03	38.76	42.11	47.00	37.90	41.44
Q-C	53.04	36.20	47.14	29.06	41.36	56.61	38.67	50.12	30.93	44.08	60.39	41.31	53.11	33.18	46.99
Resample	40.25	37.73	39.54	34.78	38.07	58.66	48.08	53.65	39.84	50.05	60.47	50.80	55.74	42.18	52.29
RP w/ hard UQ	43.74	43.27	37.62	36.17	40.2	44.92	47.14	41.89	37.92	42.96	53.60	51.39	46.95	42.96	48.72
RP(Ours)	56.31	44.09	50.51	37.18	47.02	58.35	48.37	54.42	40.27	50.35	60.51	51.49	56.08	42.53	52.65
Mix(Ours)	56.85	44.70	51.27	37.59	47.60	59.08	49.00	54.63	41.50	51.05	60.79	51.91	56.83	43.56	53.27
Ens(Ours)	<b>57.25</b>	<b>45.46</b>	<b>51.95</b>	<b>38.46</b>	<b>48.28</b>	<b>59.62</b>	<b>49.65</b>	<b>55.79</b>	<b>42.14</b>	<b>51.8</b>	<b>61.03</b>	<b>52.42</b>	<b>56.90</b>	<b>43.75</b>	<b>53.52</b>

predictions and rejects more UQs.

## 6.5 Experiments

In this section, we discuss a set of experiments that leverage the proposed RGQA dataset and metrics to evaluate the RVQA performance of both existing VQA models and proposed unsupervised RVQA training techniques. In what follows, “RP” means the model is trained with pseudo UQs, “Mix” means RoI Mixup examples are also used, and “Ens” is the ensembling of RP and Mix.

### 6.5.1 Experimental Set-up

An RVQA model consists of a VQA model  $f$  and a UQ detector  $g$ . RVQA methods vary along three dimensions: VQA model  $f$ , RVQA architecture, which determines how  $f$  and  $g$  are combined, and RVQA approach, which uses the architecture to implement the RVQA method.

We consider several models, architectures, and approaches.

**VQA models:** We consider the nine VQA models [5, 25, 181, 97, 90, 118, 221, 115, 58] listed in Table 6.2. These represent a sampling of the literature, ranging from smaller models like BUTD [5] to recent large scale models, like VinVL [221]. All models are fine-tuned on GQA [79], except BLIP [115] whose finetuning requirements exceed our resources. BUTD/UNITER/LXMERT were trained for 1/7/7 epochs, respectively, with the original hyperparameters. For MDETR/ OSCAR/ VinVL/ SwapMix, we used VQA checkpoints fine-tuned on GQA from the authors’ githubs. Since Vilt [97] does not have a GQA checkpoint, it was fine-tuned on GQA using its pre-trained weights and fine-tuning procedure from prior works [90, 118].

**RVQA approaches:** We group RVQA approaches into two categories. *Post-hoc, training free methods* use the finetuned VQA model  $f$  directly, implementing  $g$  with post-hoc operations. These frequently involve thresholding a confidence score derived from the model predictions, a popular approach in the OOD literature. *Training based methods* retrain the VQA model, using unlabeled data (pseudo-UQs), to learn  $g$ . The proposed RP, Mix, and Ens methods are of this type. We considered the following approaches.

**Post-hoc, training free methods:** **MSP [72]:** Confidence score is the largest probability output by VQA model; **ODIN [121].** Extension of MSP that uses temperature scaling and input processing to boost performance. For RVQA, input processing is only applied to visual features. The temperature is  $1e5$  and the noise  $1e-4$  for all datasets; **Maha [113].** Estimate class-conditional Gaussian distribution of the VQA model features and use the Mahalanobis distance with respect to the closest class as confidence score. **Energy [122, 193].** Energy scoring method, initially proposed for Softmax based models [122] and recently adapted to multi-label models [193]. We find that only considering the top-2 classes improves performance. **FRCNN.** A rule-based method, which compares object names detected by Faster-RCNN [162] with the nouns in the question. All object names and nouns are converted into word stems. If there exist nouns that are not in the object names, the question is declared as UQ.

**Training based methods:** **Resample [119].** An OOD method that performs iterative

adversarial weighting of background examples (i.e. pseudo UQs), assigns higher weights to harder examples and the reweighted dataset is trained. **Q-C [160]**. A caption is generated per image and its similarity to the question is measured. While [160] adopts NeuralTalk2 [93], we use BLIP [115] captions. To measure similarity, we finetune a BERT model that takes the concatenation of a caption and a question and predicts whether the two match, with a binary score.

**RVQA architectures:** Several configurations of model  $f$  and detector  $g$  were considered. **Integrated:** sequential implementation of  $g$  and  $f$  as in (6.5); **Branched:** a common backbone with decoupled classifier heads for  $f$  and  $g$ ; **Multi-branched:** generalizes Branched by taking features from multiple layers; **Separated:** trains  $g$  and  $f$  separately, with different models [116]. **K+1:** [226] defines UQs as an additional  $(K+1)^{th}$  VQA class and trains  $f$  as a  $K+1$ -class classifier. The integrated approach is applicable to all methods discussed above. The remaining architectures are only possible for training-based methods since they require pseudo-UQs to train separate  $g$  heads, models, or classes.

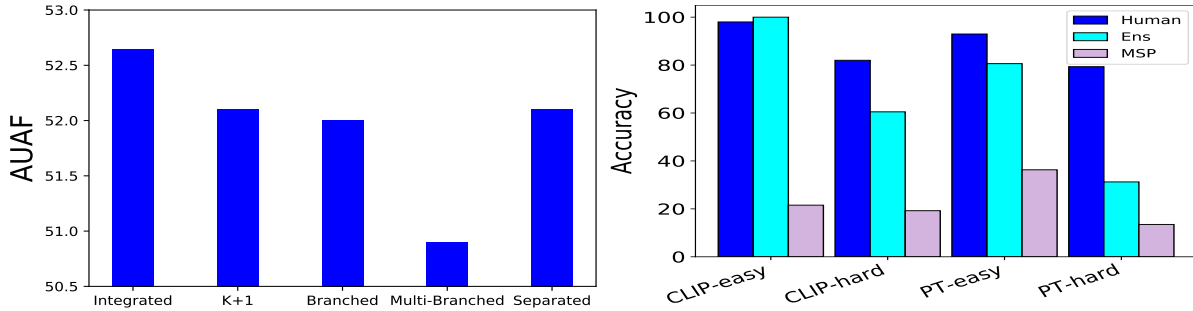
## 6.5.2 Quantitative Results

The combinatorial space of RVQA methods, VQA models, and RVQA architectures makes a comparison of all possibilities infeasible. We instead use a factorial experiment: start by ablating the architecture given a model and method, then compare models given the best architecture, and finally compare different methods for a few models.

**RVQA Architecture:** We started by investigating if the multiple architectures possible for trained models have any benefit over the integrated architecture of (6.5), which can be universally used. These experiments used the LXMERT VQA model and RP training. Fig. 6.6 left compares the averaged AUAF of the different architectures on RGQA. The *integrated* architecture has top performance, followed by Separated that, besides not being universal, doubles parameter sizes and inference time. We use the integrated architecture in the following experiments.

**VQA Model:** We next compared the UQ robustness of the different VQA models, using the MSP RVQA approach. Table 6.2 shows that all models are quite vulnerable to UQs, with





**Figure 6.6.** Left: RVQA architecture ablation. Right: Human evaluation.

average AUAF across datasets below 45. This shows that there is significant room for improvement. Interestingly, larger and more recent models do not fare significantly better than smaller models. Despite their superior AQ performance (FACC), they have similar FF95 and AUAF to the smaller models at the top of the table. Since the smaller models are much easier to train, we use them in the remaining experiments.

**RVQA Approach:** We finally compared the proposed RP, Mix, and Ens to all prior RVQA approaches discussed above. In these experiments, all approaches use BUTD, UNITER or LXMERT models. Non-trainable approaches use models learned from AQs alone, trainable methods leverage a background dataset of pseudo UQs. For Mix, we empirically find the best  $\beta$  value per model and use it for all subsets. Table 6.3 summarizes the performance of all models on the 4 RGQA subsets. The last column is the averaged AUAF across subsets. The table allows several conclusions.

**Post-hoc approaches do not help.** While MSP outperforms FRCNN, post-hoc approaches like ODIN, Maha, and Energy, which do not leverage pseudo-UQ, fail to improve on MSP. Surprisingly, these approaches have similar performance for CLIP-Easy and CLIP-Hard, even though CLIP-Easy has much coarser-grained image-question pairs.

**Pseudo UQs are effective.** The cyan cells of Table 6.3 show that training based methods, which leverage pseudo UQs, have significantly better RVQA performance (AUAF) than methods that do not. This is mainly due to a decrease of FF95 without sacrifice of FACC. Q-C consistently improves upon MSP by 5–10 pts. Resample further improves performance

for most models. However, the proposed RP improves on both, outperforming Q-C by  $\sim 5.9$  pts and Resample by  $\sim 3.4$  pts on average. This is somewhat surprising, since Resample is a more sophisticated sampling strategy. We hypothesize that Resample is unsuitable for the noisy background data generated by random pairing, likely applying larger weights to noisy examples (AQs) and hurting RVQA performance. The proposed Mix and Ens approaches have additional gains, producing the best results across VQA models. Finally, unlike prior RVQA works [116, 109], RP, Mix, and Ens do not harm VQA performance, even improving FACC.

**Impact of VQA model.** Comparing the 3 models of Table 6.3, shows that RVQA approaches are more beneficial for models of higher VQA accuracy (FACC). For instance, for MSP on CLIP-Hard, from BUTD to LXMERT a FACC increase from 53.08 to 60.49 is accompanied by an AUAF increase from around 36 to 42. This shows that better VQA reasoning skills help the model detect UQs. However, note that these gains saturate quickly, as shown in Table 6.2. Together, the two tables show that RVQA benefits more from pseudo-UQ than from large models.

**UQ Diversity.** Most approaches achieve higher AUAF on CLIP-Easy and PT-Easy, because these 2 subsets have either low CLIP score or object level mismatch between image and question. Conversely, most approaches underperform on CLIP-Hard and PT-Hard, where UQs have subtle mismatches at attribute or relation level. This trend holds across VQA models and subsets. We also consider RP training only on hard pseudo UQs, selected by CLIP score, (RP w/ hard UQs in Table 6.3), which produced a weaker AUAF than standard RP, especially on CLIP-Easy and PT-Easy. These results show the importance of UQ diversity.

### 6.5.3 Qualitative results

**Confidence score distribution:** Fig. 6.8 compares the confidence score distribution of the post-hoc MSP approach to the proposed RP and Ens training-based methods. It shows that MSP tends to be over-confident for both AQs (blue) and UQs (orange), while RP and Ens have higher (lower) scores for AQs (UQs). MSP is also not able to capture fine-grained mismatches. For instance, it assigns to UQ C a higher score than to AQ A. Finally, the confidence scores of

Q: How is the vehicle to the left of the ambulance called?



MSP:  
Train

RP:  
UQ

ENS:  
UQ

CLIP-Easy

Q: What are the jars sitting on top of?



MSP:  
Desk

RP:  
UQ

ENS:  
UQ

CLIP-Hard

Q: Does the hair that is not long look small?



MSP:  
Yes

RP:  
UQ

ENS:  
UQ

PT-Easy

Q: On which side is the wood mirror?



MSP:  
Right

RP:  
Right

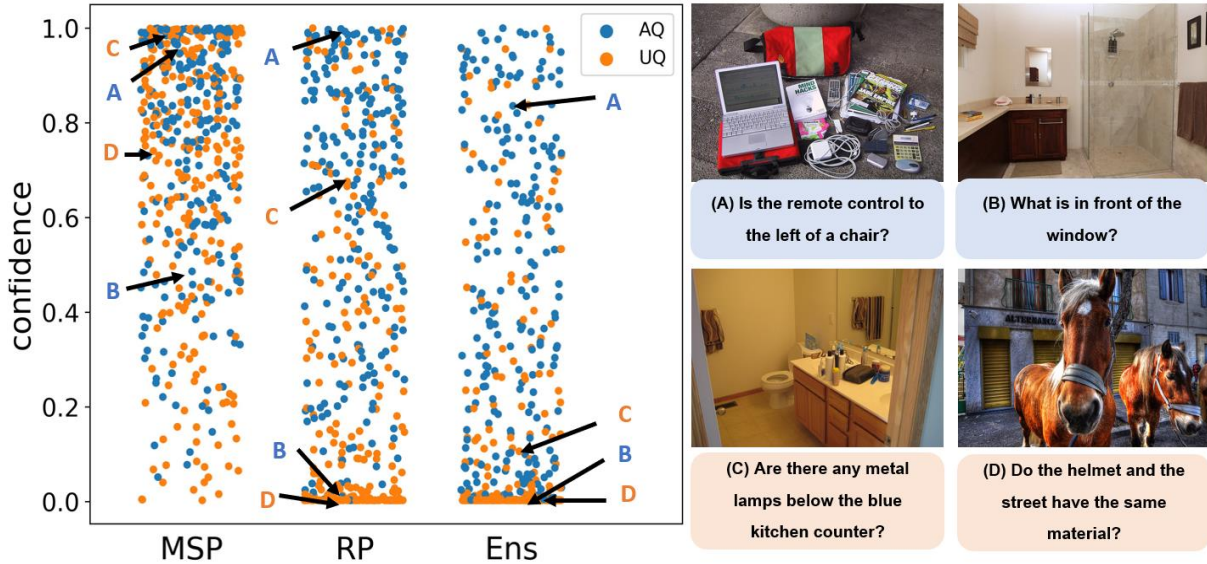
ENS:  
UQ

PT-Hard

Figure 6.7. Qualitative examples for a threshold such that all models achieve 55% accuracy.

AQ B show that RP and Ens can even detect incorrect annotations in the original GQA dataset.

**Model prediction:** Fig. 6.7 shows some qualitative examples from the four subsets of RGQA. The rejection threshold is set such that all models have accuracy of 55%. Ens correctly rejects all UQs, and RP three of the four, while MSP fails in all cases. Note that, for the fine-grained mismatches of the hard subsets, the VQA system tends to respond by statistical



**Figure 6.8.** Left: confidence scores of MSP, RP, and Ens methods for 500 random samples. Right: qualitative examples. AQs/UQs are shown in blue/orange. B is an annotation error in the original GQA dataset.

association -the missing jars are “sitting on the desk” and the nonexistent wood mirror is on the “right,” which is the side of the bike closest to the camera.

### 6.5.4 Human Evaluation

To assess the challenge posed by the UQs in RGQA dataset, we conducted a human evaluation on MTurk. Workers were asked to perform the binary rejection on 50 AQs and 50 UQs for each subset. Fig. 6.6 right shows the rejection accuracy on UQs, comparing to models thresholded so as to achieve the same true positive rate on AQs. As expected, annotators found CLIP-Hard and PT-Hard more challenging. While Ens approaches human performance on the easier subsets, the gap on harder subsets is large.

## 6.6 Conclusion

We studied the problem of realistic VQA (RVQA) that aims to both reject UQs and answer AQs. Prior RVQA methods assume labeled UQs for training. It was argued that prior datasets are insufficient because they contain poor-quality images or lack of UQ diversity. To

address this, we assembled the RGQA dataset, using 2 approaches to generate candidate UQs for human annotation. This allowed RGQA to cover broader granularities in image-question mismatch. A combination of pseudo UQs, RoI Mixup, and model ensembles was then proposed for unsupervised training of RVQA models. Experiments show that the resulting models outperform RVQA baselines for both easy and hard UQs. Comparison to human performance shows that more research is needed in RVQA.

Chapter 6 is, in full, based on the material as they appear in the publication of “Toward Unsupervised Realistic Visual Question Answering”, Chih-Hui Ho\*, Yuwei Zhang\*, and Nuno Vasconcelos, International Conference on Computer Vision (ICCV), 2023 . The dissertation author was the primary investigator and author of this paper.

## **Chapter 7**

### **Discussion and Conclusion**

In this thesis, we investigate three aspects of learning an object invariant representation, including the visual invariance, semantic invariance and the detection of outliers of object invariance. To tackle the challenges in each aspects, we have collected datasets to probe the robustness of existing deep models, proposed new loss functions and design novel architectures. First, we curated two large multiview datasets (one in the lab and one in the wild) and showed that the classifiers fail to provide consistent prediction when different view of an object are shown to the classifier. To address this, we proposed, PIE, a novel loss function that learns a structured feature space. More specifically, the feature space is organized in a hierarchical manner by pushing the feature of objects from the same object class closer and the feature of image views from the same object even closer. With the use of the proposed loss function, the existing model becomes more robust when partial views or single views are given during inference time. While the proposed PIE loss function requires label data for supervised training, we found that the model can be equipped with the view invariance property with self-supervised learning, where no labeled dataset is required. We proposed VISPE, which is a self-supervised learning method to pretrain the model, and we show that the resulting model can be applied for both classification and retrieval task.

Second, we extend the learning of object invariance by adding language modality. We first show that the existing models are not applicable to the open granularity classification, which aims to support any label set that spans across different granularity. To address this, we proposed ProTeCt, which is a prompt tuning method that can rectify the existing visual language model and allow them to support open granularity classification. Results shows that the models trained with ProTeCt provide more consistent prediction when the label set changes and such consistency can be transferred to other unseen dataset without further training. Finally, we study the outlier case where the image and text does not match with each other by investigating the visual question answering task. We proposed the notion of unanswerable question, which are questions that the model cannot answer by inspecting the image. A large evaluation dataset, RGQA, is curated to examine the existing models and a novel fine-tuning method is proposed. Experiments show the proposed method can better recognize these unanswerable questions without reducing the

capability to answer those answerable ones.

Beyond the above aspects, the study of object invariance can be extended to 3D visual language task. While visual language has been a popular task in 2D domain, the research on 3D visual language has been less investigated, which is quite surprising since we are living in a 3D world. With the emergence of large image-text foundational models, it remains a challenge to adopt these large-scale image-text foundational models for 3D visual language tasks. This can be potentially implemented by using multi-view images as a medium to transfer the knowledge from 2D image-text foundational models to the 3D world. Furthermore, the learning of object invariance can be used for downstream task, such as object generation through text. The integration of 2D visual language and single view 3D reconstruction is to jointly understand the text and reconstruct the 3D object shape. This requires a more sophisticated optimization of the object invariant feature and the text feature to directly generate high-fidelity 3D object shapes. Furthermore, as a human, we are living in the multi-modal world, where we can receive signal of all kinds, such as image, audio, thermal, language and etc. This equipped us with the capability of recognizing the object with all different clues. For example, we can know that there is a dog when we hear a bark, without even seeing it. While human can easily achieve this, this is not the case for existing models. In fact, to achieve a robust embodied AI, the alignment across modalities is a fundamental challenge. The learning of object invariant representation will continue to play an important role for modeling the complex world in the future.



# Bibliography

- [1] A.Dosovitskiy, J.T.Springenberg, M.Riedmiller, and T.Brox. Discriminative unsupervised feature learning with convolutional neural networks. In *Advances in Neural Information Processing Systems 27 (NIPS)*, 2014.
- [2] P. Agrawal, J. Carreira, and J. Malik. Learning to see by moving. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 37–45, Dec 2015.
- [3] Karim Ahmed, Mohammad Haris Baig, and Lorenzo Torresani. Network of experts for large-scale image categorization. In *European Conference on Computer Vision (ECCV)*, 2016.
- [4] Unaiza Ahsan, Rishi Madhok, and Irfan A. Essa. Video jigsaw: Unsupervised learning of spatiotemporal context for video action recognition. *CoRR*, abs/1808.07507, 2018.
- [5] Peter Anderson, Xiaodong He, Chris Buehler, Damien Teney, Mark Johnson, Stephen Gould, and Lei Zhang. Bottom-up and top-down attention for image captioning and visual question answering. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6077–6086, 2018.
- [6] Peter Anderson, Qi Wu, Damien Teney, Jake Bruce, Mark Johnson, Niko Sünderhauf, Ian Reid, Stephen Gould, and Anton Van Den Hengel. Vision-and-language navigation: Interpreting visually-grounded navigation instructions in real environments. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3674–3683, 2018.
- [7] Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C. Lawrence Zitnick, and Devi Parikh. Vqa: Visual question answering. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 2425–2433, 2015.
- [8] Anish Athalye, Logan Engstrom, Andrew Ilyas, and Kevin Kwok. Synthesizing robust adversarial examples. *CoRR*, abs/1707.07397, 2017.
- [9] Lei Jimmy Ba and Brendan Frey. Adaptive dropout for training deep neural networks. In *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2, NIPS’13*, pages 3084–3092, USA, 2013. Curran Associates Inc.
- [10] Wentao Bao, Qi Yu, and Yu Kong. Evidential deep learning for open set action recognition. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 13349–13358, 2021.

- [11] Miguel Ángel Bautista, Artsiom Sanakoyeu, Ekaterina Tikhoncheva, and Björn Ommer. Cliquecnn: Deep unsupervised exemplar learning. In *NIPS*, 2016.
- [12] Sean Bell and Kavita Bala. Learning visual similarity for product design with convolutional neural networks. *ACM Transactions on Graphics (TOG)*, 34(4):98, 2015.
- [13] A. Borji, S. Izadi, and L. Itti. ilab-20m: A large-scale controlled object dataset to investigate deep learning. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2221–2230, June 2016.
- [14] André Brock, Theodore Lim, James M. Ritchie, and Nick Weston. Generative and discriminative voxel modeling with convolutional neural networks. *CoRR*, abs/1608.04236, 2016.
- [15] Tom B. Brown, Dandelion Mané, Aurko Roy, Martín Abadi, and Justin Gilmer. Adversarial patch. *CoRR*, abs/1712.09665, 2017.
- [16] Adrian Bulat and Georgios Tzimiropoulos. Lasp: Text-to-text optimization for language-aware soft prompting of vision&language models. 2022.
- [17] Nicholas Carlini and David A. Wagner. Towards evaluating the robustness of neural networks. *CoRR*, abs/1608.04644, 2016.
- [18] Mathilde Caron, Piotr Bojanowski, Armand Joulin, and Matthijs Douze. Deep clustering for unsupervised learning of visual features. In *European Conference on Computer Vision*, 2018.
- [19] Angel X. Chang, Thomas A. Funkhouser, Leonidas J. Guibas, Pat Hanrahan, Qi-Xing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, Jianxiong Xiao, Li Yi, and Fisher Yu. Shapenet: An information-rich 3d model repository. *CoRR*, abs/1512.03012, 2015.
- [20] Chongyan Chen, Samreen Anjum, and Danna Gurari. Grounding answers for visual questions asked by visually impaired people. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 19098–19107, June 2022.
- [21] Jie-Neng Chen, Shuyang Sun, Ju He, Philip Torr, Alan Yuille, and Song Bai. Transmix: Attend to mix for vision transformers. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2022.
- [22] Pin-Yu Chen, Huan Zhang, Yash Sharma, Jinfeng Yi, and Cho-Jui Hsieh. Zoo: Zeroth order optimization based black-box attacks to deep neural networks without training substitute models. In *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security, AISec '17*, pages 15–26, New York, NY, USA, 2017. ACM.
- [23] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *Proceedings of the 37th International Conference on Machine Learning, ICML'20*. JMLR.org, 2020.

- [24] Xinlei Chen, Haoqi Fan, Ross B. Girshick, and Kaiming He. Improved baselines with momentum contrastive learning. *ArXiv*, abs/2003.04297, 2020.
- [25] Yen-Chun Chen, Linjie Li, Licheng Yu, Ahmed El Kholy, Faisal Ahmed, Zhe Gan, Yu Cheng, and Jingjing Liu. Uniter: Universal image-text representation learning. In *ECCV*, 2020.
- [26] Jiacheng Cheng and Nuno Vasconcelos. Learning deep classifiers consistent with fine-grained novelty detection. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1664–1673, 2021.
- [27] Sumit Chopra, Raia Hadsell, and Yann LeCun. Learning a similarity metric discriminatively, with application to face verification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, pages 539–546. IEEE, 2005.
- [28] Thomas M. Cover and Joy A. Thomas. *Elements of Information Theory*. Wiley-Interscience, New York, NY, USA, 1991.
- [29] Guoxian Dai, Jin Xie, and Yi Fang. Siamese cnn-bilstm architecture for 3d shape representation learning. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18*, pages 670–676. International Joint Conferences on Artificial Intelligence Organization, 7 2018.
- [30] Ritendra Datta, Dhiraj Joshi, Jia Li, and James Z. Wang. Image retrieval: Ideas, influences, and trends of the new age. *ACM Comput. Surv.*, 40(2):5:1–5:60, May 2008.
- [31] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR09*, 2009.
- [32] Jia Deng, Nan Ding, Yangqing Jia, Andrea Frome, Kevin Murphy, Samy Bengio, Yuan Li, Hartmut Neven, and Hartwig Adam. Large-scale object classification using label relation graphs. In *European Conference on Computer Vision (ECCV)*, 2014.
- [33] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009.
- [34] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 248–255. IEEE, 2009.
- [35] Jia Deng, Jonathan Krause, Alexander C. Berg, and Li Fei-Fei. Hedging your bets: Optimizing accuracy-specificity trade-offs in large scale visual recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.
- [36] Akshay Raj Dhamija, Manuel Günther, and Terrance Boulton. Reducing network agnostophobia. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

- [37] Akshay Raj Dhamija, Manuel Günther, and Terrance E. Boult. Reducing network agnostophobia. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, NIPS’18, page 9175–9186, Red Hook, NY, USA, 2018. Curran Associates Inc.
- [38] Carl Doersch, Abhinav Gupta, and Alexei A. Efros. Unsupervised visual representation learning by context prediction. *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 1422–1430, 2015.
- [39] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021.
- [40] Sivan Doveh, Assaf Arbelle, Sivan Harary, Rameswar Panda, Roei Herzig, Eli Schwartz, Donghyun Kim, Raja Giryes, Rogério Schmidt Feris, Shimon Ullman, and Leonid Karlinsky. Teaching structured vision & language concepts to vision & language models. *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2657–2668, 2022.
- [41] Yifan Du, Zikang Liu, Junyi Li, and Wayne Xin Zhao. A survey of vision-language pre-trained models. In *International Joint Conference on Artificial Intelligence*, 2022.
- [42] Logan Engstrom, Dimitris Tsipras, Ludwig Schmidt, and Aleksander Madry. A rotation and a translation suffice: Fooling cnns with simple transformations. *CoRR*, abs/1712.02779, 2017.
- [43] Ivan Evtimov, Kevin Eykholt, Earlene Fernandes, Tadayoshi Kohno, Bo Li, Atul Prakash, Amir Rahmati, and Dawn Song. Robust physical-world attacks on machine learning models. *CoRR*, abs/1707.08945, 2017.
- [44] Yifan Feng, Haoxuan You, Zizhao Zhang, Rongrong Ji, and Yue Gao. Hypergraph neural networks. *CoRR*, abs/1809.09401, 2018.
- [45] Yifan Feng, Zizhao Zhang, Xibin Zhao, Rongrong Ji, and Yue Gao. Gvcnn: Group-view convolutional neural networks for 3d shape recognition. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.
- [46] Yifan Feng, Zizhao Zhang, Xibin Zhao, Rongrong Ji, and Yue Gao. Gvcnn: Group-view convolutional neural networks for 3d shape recognition. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.
- [47] Zhe Gan, Yen-Chun Chen, Linjie Li, Chen Zhu, Yu Cheng, and Jingjing Liu. Large-scale adversarial training for vision-and-language representation learning. In *NeurIPS*, 2020.

- [48] Peng Gao, Shijie Geng, Renrui Zhang, Teli Ma, Rongyao Fang, Yongfeng Zhang, Hongsheng Li, and Yu Jiao Qiao. Clip-adapter: Better vision-language models with feature adapters. *ArXiv*, abs/2110.04544, 2021.
- [49] Zan Gao, Deyu Wang, Xiangnan He, and Hua Zhang. Group-pair convolutional neural networks for multi-view based 3d object retrieval. In *AAAI*, 2018.
- [50] A. Garcia-Garcia, F. Gomez-Donoso, J. Garcia-Rodriguez, S. Orts-Escolano, M. Cazorla, and J. Azorin-Lopez. Pointnet: A 3d convolutional neural network for real-time object class recognition. In *2016 International Joint Conference on Neural Networks (IJCNN)*, pages 1578–1584, July 2016.
- [51] Jan-Mark Geusebroek, Gertjan J Burghouts, and Arnold WM Smeulders. The amsterdam library of object images. *International Journal of Computer Vision*, 61(1):103–112, 2005.
- [52] Spyros Gidaris, Praveer Singh, and Nikos Komodakis. Unsupervised representation learning by predicting image rotations. *CoRR*, abs/1803.07728, 2018.
- [53] Jacob Goldberger, Geoffrey E Hinton, Sam T Roweis, and Ruslan R Salakhutdinov. Neighbourhood components analysis. In *Advances in neural information processing systems*, pages 513–520, 2005.
- [54] Wonjoon Goo, Juyong Kim, Gunhee Kim, and Sung Ju Hwang. Taxonomy-regularized semantic deep convolutional neural networks. In *European Conference on Computer Vision (ECCV)*, 2016.
- [55] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. <http://www.deeplearningbook.org>.
- [56] Ian Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In *International Conference on Learning Representations*, 2015.
- [57] Yash Goyal, Tejas Khot, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh. Making the v in vqa matter: Elevating the role of image understanding in visual question answering. *International Journal of Computer Vision*, 127:398–414, 2017.
- [58] Vipul Gupta, Zhuowan Li, Adam Kortylewski, Chenyu Zhang, Yingwei Li, and Alan Loddon Yuille. Swapmix: Diagnosing and regularizing the over-reliance on visual context in visual question answering. *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5068–5078, 2022.
- [59] Danna Gurari, Qing Li, Abigale J Stangl, Anhong Guo, Chi Lin, Kristen Grauman, Jiebo Luo, and Jeffrey P Bigham. Vizwiz grand challenge: Answering visual questions from blind people. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3608–3617, 2018.

- [60] R. Hadsell, S. Chopra, and Y. LeCun. Dimensionality reduction by learning an invariant mapping. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 2, pages 1735–1742, June 2006.
- [61] Raia Hadsell, Sumit Chopra, and Yann LeCun. Dimensionality reduction by learning an invariant mapping. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1735–1742. IEEE, 2006.
- [62] Z. Han, H. Lu, Z. Liu, C. Vong, Y. Liu, M. Zwicker, J. Han, and C. L. P. Chen. 3d2seqviews: Aggregating sequential views for 3d global feature learning by cnn with hierarchical attention aggregation. *IEEE Transactions on Image Processing*, 28(8):3986–3999, Aug 2019.
- [63] J. A. Hartigan and M. A. Wong. Algorithm as 136: A k-means clustering algorithm. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 28(1):100–108, 1979.
- [64] Sadid A. Hasan, Yuan Ling, Oladimeji Farri, Joey Liu, Henning Müller, and Matthew P. Lungren. Overview of imageclef 2018 medical domain visual question answering task. In *Conference and Labs of the Evaluation Forum*, 2018.
- [65] Kaiming He, X. Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.
- [66] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. *CoRR*, abs/1502.01852, 2015.
- [67] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.
- [68] Xinwei He, Yang Zhou, Zhichao Zhou, Song Bai, and Xiang Bai. Triplet-center loss for multi-view 3d object retrieval. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018.
- [69] Xinwei He, Yang Zhou, Zhichao Zhou, Song Bai, and Xiang Bai. Triplet-center loss for multi-view 3d object retrieval. *CoRR*, abs/1803.06189, 2018.
- [70] Vishakh Hegde and Reza Zadeh. Fusionnet: 3d object classification using multiple data representations. *CoRR*, abs/1607.05695, 2016.
- [71] Dan Hendrycks, Steven Basart, Norman Mu, Saurav Kadavath, Frank Wang, Evan Dorundo, Rahul Desai, Tyler Lixuan Zhu, Samyak Parajuli, Mike Guo, Dawn Xiaodong Song, Jacob Steinhardt, and Justin Gilmer. The many faces of robustness: A critical analysis of out-of-distribution generalization. *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 8320–8329, 2020.

- [72] Dan Hendrycks and Kevin Gimpel. A baseline for detecting misclassified and out-of-distribution examples in neural networks. *Proceedings of International Conference on Learning Representations*, 2017.
- [73] Dan Hendrycks and Kevin Gimpel. A baseline for detecting misclassified and out-of-distribution examples in neural networks. In *International Conference on Learning Representations*, 2017.
- [74] Dan Hendrycks, Mantas Mazeika, and Thomas Dietterich. Deep anomaly detection with outlier exposure. *Proceedings of the International Conference on Learning Representations*, 2019.
- [75] Dan Hendrycks, Kevin Zhao, Steven Basart, Jacob Steinhardt, and Dawn Song. Natural adversarial examples. *CVPR*, 2021.
- [76] Chih-Hui Ho, Pedro Morgado, Amir Persekian, and Nuno Vasconcelos. Pies: Pose invariant embeddings. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.
- [77] Hossein Hosseini and Radha Poovendran. Semantic adversarial examples. *CoRR*, abs/1804.00499, 2018.
- [78] Ruitong Huang, Bing Xu, Dale Schuurmans, and Csaba Szepesvári. Learning with a strong adversary. *CoRR*, abs/1511.03034, 2015.
- [79] Drew A Hudson and Christopher D Manning. Gqa: A new dataset for real-world visual reasoning and compositional question answering. *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.
- [80] Dinesh Jayaraman, Ruohan Gao, and Kristen Grauman. Unsupervised learning through one-shot image-based shape reconstruction. *CoRR*, abs/1709.00505, 2017.
- [81] Dinesh Jayaraman and Kristen Grauman. Slow and steady feature analysis: Higher order temporal coherence in video. *CoRR*, abs/1506.04714, 2015.
- [82] Dinesh Jayaraman and Kristen Grauman. Learning image representations tied to egomotion from unlabeled video. *Int. J. Comput. Vision*, 125(1-3):136–161, December 2017.
- [83] Chao Jia, Yinfei Yang, Ye Xia, Yi-Ting Chen, Zarana Parekh, Hieu Pham, Quoc V. Le, Yun-Hsuan Sung, Zhen Li, and Tom Duerig. Scaling up visual and vision-language representation learning with noisy text supervision. In *International Conference on Machine Learning*, 2021.
- [84] Menglin Jia, Luming Tang, Bor-Chun Chen, Claire Cardie, Serge Belongie, Bharath Hariharan, and Ser-Nam Lim. Visual prompt tuning. In *European Conference on Computer Vision (ECCV)*, 2022.

- [85] Huaizu Jiang, Gustav Larsson, Michael Maire, Greg Shakhnarovich, and Erik Learned-Miller. Self-supervised relative depth learning for urban scene understanding. In Vittorio Ferrari, Martial Hebert, Cristian Sminchisescu, and Yair Weiss, editors, *Computer Vision – ECCV 2018*, pages 20–37, Cham, 2018. Springer International Publishing.
- [86] Longlong Jing and Yingli Tian. Self-supervised visual feature learning with deep neural networks: A survey. *CoRR*, abs/1902.06162, 2019.
- [87] Justin Johnson, Bharath Hariharan, Laurens van der Maaten, Li Fei-Fei, C. Lawrence Zitnick, and Ross B. Girshick. Clevr: A diagnostic dataset for compositional language and elementary visual reasoning. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1988–1997, 2017.
- [88] Jung-Eun Lee, Rong Jin, and A. K. Jain. Rank-based distance metric learning: An application to image retrieval. In *2008 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, June 2008.
- [89] Kushal Kafle and Christopher Kanan. An analysis of visual question answering algorithms. In *Proceedings of the IEEE international conference on computer vision*, pages 1965–1973, 2017.
- [90] Aishwarya Kamath, Mannat Singh, Yann LeCun, Ishan Misra, Gabriel Synnaeve, and Nicolas Carion. Mdetr - modulated detection for end-to-end multi-modal understanding. *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 1760–1770, 2021.
- [91] Asako Kanezaki. Rotationnet: Learning object classification using unsupervised viewpoint estimation. *CoRR*, abs/1603.06208, 2016.
- [92] Asako Kanezaki, Yasuyuki Matsushita, and Yoshifumi Nishida. Rotationnet: Joint object categorization and pose estimation using multiviews from unsupervised viewpoints. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.
- [93] Andrej Karpathy and Li Fei-Fei. Deep visual-semantic alignments for generating image descriptions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3128–3137, 2015.
- [94] Muhammad Uzair khattak, Hanoona Rasheed, Muhammad Maaz, Salman Khan, and Fahad Shahbaz Khan. Maple: Multi-modal prompt learning. In *The IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023.
- [95] Dahun Kim, Donghyeon Cho, Donggeun Yoo, and In So Kweon. Learning image representations by completing damaged jigsaw puzzles. *CoRR*, abs/1802.01880, 2018.
- [96] Hyo Jin Kim and Jan-Michael Frahm. Hierarchy of alternating specialists for scene recognition. In *European Conference on Computer Vision (ECCV)*, 2018.



- [97] Wonjae Kim, Bokyung Son, and Ildoo Kim. Vilt: Vision-and-language transformer without convolution or region supervision. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 5583–5594. PMLR, 18–24 Jul 2021.
- [98] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014.
- [99] Muhammed Kocabas, Salih Karagoz, and Emre Akbas. Self-supervised learning of 3d human pose using multi-view geometry. *CoRR*, abs/1903.02330, 2019.
- [100] Alex Krizhevsky. Learning multiple layers of features from tiny images. 2009.
- [101] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.
- [102] Alexey Kurakin, Ian J. Goodfellow, and Samy Bengio. Adversarial examples in the physical world. *CoRR*, abs/1607.02533, 2016.
- [103] Alexey Kurakin, Ian J. Goodfellow, and Samy Bengio. Adversarial machine learning at scale. *CoRR*, abs/1611.01236, 2016.
- [104] Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 6402–6413. Curran Associates, Inc., 2017.
- [105] Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS’17, page 6405–6416, Red Hook, NY, USA, 2017. Curran Associates Inc.
- [106] Matthew Lamm, Jennimaria Palomaki, Chris Alberti, Daniel Andor, Eunsol Choi, Livio Baldini Soares, and Michael Collins. QED: A framework and dataset for explanations in question answering. *Transactions of the Association for Computational Linguistics*, 9:790–806, 2021.
- [107] Gustav Larsson, Michael Maire, and Gregory Shakhnarovich. Learning representations for automatic colorization. *CoRR*, abs/1603.06668, 2016.
- [108] Gustav Larsson, Michael Maire, and Gregory Shakhnarovich. Colorization as a proxy task for visual understanding. *CoRR*, abs/1703.04044, 2017.
- [109] Doyup Lee, Yeongjae Cheon, and Wook-Shin Han. Regularizing attention networks for anomaly detection in visual question answering. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 1845–1853, 2021.

- [110] Haebeom Lee, Juho Lee, Eunho Yang, and Sung Ju Hwang. Dropmax: Adaptive stochastic softmax. *CoRR*, abs/1712.07834, 2017.
- [111] Hsin-Ying Lee, Jia-Bin Huang, Maneesh Singh, and Ming-Hsuan Yang. Unsupervised representation learning by sorting sequences. *CoRR*, abs/1708.01246, 2017.
- [112] Kibok Lee, Kimin Lee, Kyle Min, Yuting Zhang, Jinwoo Shin, and Honglak Lee. Hierarchical novelty detection for visual object recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.
- [113] Kimin Lee, Kibok Lee, Honglak Lee, and Jinwoo Shin. A simple unified framework for detecting out-of-distribution samples and adversarial attacks. *Advances in neural information processing systems*, 31, 2018.
- [114] Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3045–3059, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics.
- [115] Junnan Li, Dongxu Li, Caiming Xiong, and Steven Hoi. Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation. In *ICML*, 2022.
- [116] Mengdi Li, Cornelius Weber, and Stefan Wermter. Neural networks for detecting irrelevant questions during visual question answering. In *International Conference on Artificial Neural Networks*, pages 786–797. Springer, 2020.
- [117] Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4582–4597, Online, August 2021. Association for Computational Linguistics.
- [118] Xiujun Li, Xi Yin, Chunyuan Li, Xiaowei Hu, Pengchuan Zhang, Lei Zhang, Lijuan Wang, Houdong Hu, Li Dong, Furu Wei, Yejin Choi, and Jianfeng Gao. Oscar: Object-semantics aligned pre-training for vision-language tasks. *ECCV 2020*, 2020.
- [119] Yi Li and Nuno Vasconcelos. Background data resampling for outlier-aware classification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.
- [120] Zhaoqun Li, Cheng Xu, and Biao Leng. Angular triplet-center loss for multi-view 3d shape retrieval. *CoRR*, abs/1811.08622, 2018.
- [121] Shiyu Liang, Yixuan Li, and R. Srikant. Enhancing the reliability of out-of-distribution image detection in neural networks. In *International Conference on Learning Representations*, 2018.

- [122] Weitang Liu, Xiaoyun Wang, John Owens, and Yixuan Li. Energy-based out-of-distribution detection. *Advances in Neural Information Processing Systems*, 2020.
- [123] Xiao Liu, Kaixuan Ji, Yicheng Fu, Weng Tam, Zhengxiao Du, Zhilin Yang, and Jie Tang. P-tuning: Prompt tuning can be comparable to fine-tuning across scales and tasks. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 61–68, Dublin, Ireland, May 2022. Association for Computational Linguistics.
- [124] Yanpei Liu, Xinyun Chen, Chang Liu, and Dawn Song. Delving into transferable adversarial examples and black-box attacks. *CoRR*, abs/1611.02770, 2016.
- [125] Yuntao Liu, Yong Dou, Ruochun Jin, and Peng Qiao. Visual tree convolutional neural network in image classification. In *International Conference on Pattern Recognition (ICPR)*, 2018.
- [126] Ze Liu, Han Hu, Yutong Lin, Zhuliang Yao, Zhenda Xie, Yixuan Wei, Jia Ning, Yue Cao, Zheng Zhang, Li Dong, Furu Wei, and Baining Guo. Swin transformer v2: Scaling up capacity and resolution. In *International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.
- [127] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 10012–10022, October 2021.
- [128] Timo Lüddecke and Alexander Ecker. Image segmentation using text and image prompts. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7086–7096, June 2022.
- [129] C. Ma, Y. Guo, J. Yang, and W. An. Learning multi-view representation with lstm for 3-d shape recognition and retrieval. *IEEE Transactions on Multimedia*, 21(5):1169–1182, May 2019.
- [130] Aroma Mahendru, Viraj Prabhu, Akrit Mohapatra, Dhruv Batra, and Stefan Lee. The promise of premise: Harnessing question premises in visual question answering. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 926–935, Copenhagen, Denmark, September 2017. Association for Computational Linguistics.
- [131] Subhransu Maji, Esa Rahtu, Juho Kannala, Matthew B. Blaschko, and Andrea Vedaldi. Fine-grained visual classification of aircraft. *ArXiv*, abs/1306.5151, 2013.
- [132] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. *Introduction to Information Retrieval*. Cambridge University Press, Cambridge, UK, 2008.

- [133] Kenneth Marino, Mohammad Rastegari, Ali Farhadi, and Roozbeh Mottaghi. Ok-vqa: A visual question answering benchmark requiring external knowledge. In *Proceedings of the IEEE/cvf conference on computer vision and pattern recognition*, pages 3195–3204, 2019.
- [134] Marcin Marszałek and Cordelia Schmid. Semantic hierarchies for visual object recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2007.
- [135] D. Maturana and S. Scherer. Voxnet: A 3d convolutional neural network for real-time object recognition. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 922–928, Sept 2015.
- [136] George A. Miller. Wordnet: A lexical database for english. *Commun. ACM*, 38(11):39–41, November 1995.
- [137] Yifei Ming, Ying Fan, and Yixuan Li. Poem: Out-of-distribution detection with posterior sampling. In *International Conference on Machine Learning*. PMLR, 2022.
- [138] Ishan Misra, C. Lawrence Zitnick, and Martial Hebert. Unsupervised learning using sequential verification for action recognition. *CoRR*, abs/1603.08561, 2016.
- [139] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, Omar Fawzi, and Pascal Frossard. Universal adversarial perturbations. *CoRR*, abs/1610.08401, 2016.
- [140] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard. Deepfool: a simple and accurate method to fool deep neural networks. *CoRR*, abs/1511.04599, 2015.
- [141] Yair Movshovitz-Attias, Alexander Toshev, Thomas K Leung, Sergey Ioffe, and Saurabh Singh. No fuss distance metric learning using proxies. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017.
- [142] Yair Movshovitz-Attias, Alexander Toshev, Thomas K. Leung, Sergey Ioffe, and Saurabh Singh. No fuss distance metric learning using proxies. *CoRR*, abs/1703.07464, 2017.
- [143] T. Nathan Mundhenk, Daniel Ho, and Barry Y. Chen. Improvements to context based self-supervised learning. *CoRR*, abs/1711.06379, 2017.
- [144] Sanjeev Muralikrishnan, Vladimir G. Kim, Matthew Fisher, and Siddhartha Chaudhuri. Shape unicode: A unified shape representation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.
- [145] Sameer A Nene, Shree K Nayar, and Hiroshi Murase. Columbia object image library (coil-100).
- [146] Mehdi Noroozi and Paolo Favaro. Unsupervised learning of visual representations by solving jigsaw puzzles. *CoRR*, abs/1603.09246, 2016.
- [147] Hyun Oh Song, Yu Xiang, Stefanie Jegelka, and Silvio Savarese. Deep metric learning via lifted structured feature embedding. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4004–4012, 2016.

- [148] Poojan Oza and Vishal M Patel. C2ae: Class conditioned auto-encoder for open-set recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2307–2316, 2019.
- [149] Nicolas Papernot, Patrick D. McDaniel, and Ian J. Goodfellow. Transferability in machine learning: from phenomena to black-box attacks using adversarial samples. *CoRR*, abs/1605.07277, 2016.
- [150] Nicolas Papernot, Patrick D. McDaniel, Somesh Jha, Matt Fredrikson, Z. Berkay Celik, and Ananthram Swami. The limitations of deep learning in adversarial settings. *CoRR*, abs/1511.07528, 2015.
- [151] Eunbyung Park, Jimei Yang, Ersin Yumer, Duygu Ceylan, and Alexander C. Berg. Transformation-grounded image generation network for novel 3d view synthesis. *CoRR*, abs/1703.02921, 2017.
- [152] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017.
- [153] Deepak Pathak, Ross B. Girshick, Piotr Dollár, Trevor Darrell, and Bharath Hariharan. Learning features by watching objects move. *CoRR*, abs/1612.06370, 2016.
- [154] Deepak Pathak, Philipp Krähenbühl, Jeff Donahue, Trevor Darrell, and Alexei Efros. Context encoders: Feature learning by inpainting. 2016.
- [155] Charles Ruizhongtai Qi, Hao Su, Matthias Nießner, Angela Dai, Mengyuan Yan, and Leonidas J. Guibas. Volumetric and multi-view cnns for object classification on 3d data. *CoRR*, abs/1604.03265, 2016.
- [156] Charles Ruizhongtai Qi, Hao Su, Matthias Nießner, Angela Dai, Mengyuan Yan, and Leonidas J. Guibas. Volumetric and multi-view cnns for object classification on 3d data. *CoRR*, abs/1604.03265, 2016.
- [157] Hang Qi, Matthew Brown, and David G. Lowe. Learning with imprinted weights. *CoRR*, abs/1712.07136, 2017.
- [158] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pages 8748–8763. PMLR, 2021.
- [159] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. *CoRR*, abs/2103.00020, 2021.

- [160] Arijit Ray, Gordon Christie, Mohit Bansal, Dhruv Batra, and Devi Parikh. Question relevance in VQA: Identifying non-visual and false-premise questions. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 919–924, Austin, Texas, November 2016. Association for Computational Linguistics.
- [161] Benjamin Recht, Rebecca Roelofs, Ludwig Schmidt, and Vaishal Shankar. Do imagenet classifiers generalize to imagenet? In *International Conference on Machine Learning*, 2019.
- [162] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015.
- [163] Ruslan Salakhutdinov, Antonio Torralba, and Josh Tenenbaum. Learning to share visual appearance for multiclass object detection. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2011.
- [164] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 815–823, 2015.
- [165] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. *CoRR*, abs/1503.03832, 2015.
- [166] Babak Shahbaba and Radford M. Neal. Improving classification when a class hierarchy is available using a hierarchy-based prior. *Bayesian Analysis*, 2(1):221–238, 2007.
- [167] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014.
- [168] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.
- [169] Jake Snell, Kevin Swersky, and Richard S. Zemel. Prototypical networks for few-shot learning. *CoRR*, abs/1703.05175, 2017.
- [170] Kihyuk Sohn. Improved deep metric learning with multi-class n-pair loss objective. In *Advances in Neural Information Processing Systems*, pages 1857–1865, 2016.
- [171] Kihyuk Sohn. Improved deep metric learning with multi-class n-pair loss objective. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, pages 1857–1865. Curran Associates, Inc., 2016.
- [172] Kihyuk Sohn. Improved deep metric learning with multi-class n-pair loss objective. In *NIPS*, 2016.

- [173] Hyun Oh Song, Yu Xiang, Stefanie Jegelka, and Silvio Savarese. Deep metric learning via lifted structured feature embedding. *CoRR*, abs/1511.06452, 2015.
- [174] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.*, 15(1):1929–1958, January 2014.
- [175] Hang Su, Subhransu Maji, Evangelos Kalogerakis, and Erik Learned-Miller. Multi-view convolutional neural networks for 3d shape recognition. In *Proceedings of the IEEE international conference on computer vision*, pages 945–953, 2015.
- [176] Hang Su, Subhransu Maji, Evangelos Kalogerakis, and Erik G. Learned-Miller. Multi-view convolutional neural networks for 3d shape recognition. *CoRR*, abs/1505.00880, 2015.
- [177] Jiawei Su, Danilo Vasconcellos Vargas, and Kouichi Sakurai. One pixel attack for fooling deep neural networks. *CoRR*, abs/1710.08864, 2017.
- [178] Yi Sun, Yuheng Chen, Xiaogang Wang, and Xiaoou Tang. Deep learning face representation by joint identification-verification. In *Advances in neural information processing systems*, pages 1988–1996, 2014.
- [179] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott E. Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. *CoRR*, abs/1409.4842, 2014.
- [180] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian J. Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *CoRR*, abs/1312.6199, 2013.
- [181] Hao Tan and Mohit Bansal. Lxmert: Learning cross-modality encoder representations from transformers. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*, 2019.
- [182] Tristan Thrush, Ryan Jiang, Max Bartolo, Amanpreet Singh, Adina Williams, Douwe Kiela, and Candace Ross. Winoground: Probing vision and language models for visio-linguistic compositionality. In *CVPR*, 2022.
- [183] Andeep S Toor and Harry Wechsler. Biometrics and forensics integration using deep multi-modal semantic alignment and joint embedding. *Pattern Recognition Letters*, 113:29–37, 2018.
- [184] Andeep S. Toor, Harry Wechsler, and Michele Nappi. Question part relevance and editing for cooperative and context-aware vqa (c2vqa). In *Proceedings of the 15th International Workshop on Content-Based Multimedia Indexing*, CBMI '17, New York, NY, USA, 2017. Association for Computing Machinery.

- [185] Florian Tramèr, Alexey Kurakin, Nicolas Papernot, Ian Goodfellow, Dan Boneh, and Patrick McDaniel. Ensemble adversarial training: Attacks and defenses. In *International Conference on Learning Representations*, 2018.
- [186] Aäron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *ArXiv*, abs/1807.03748, 2018.
- [187] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9:2579–2605, 2008.
- [188] Apoorv Vyas, Nataraj Jammalamadaka, Xia Zhu, Dipankar Das, Bharat Kaul, and Theodore L. Willke. Out-of-distribution detection using an ensemble of self supervised leave-out classifiers. In *ECCV*, 2018.
- [189] Chengde Wan, Thomas Probst, Luc Van Gool, and Angela Yao. Self-supervised 3d hand pose estimation through training by fitting. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.
- [190] Chu Wang, Marcello Pelillo, and Kaleem Siddiqi. Dominant set clustering and pooling for multi-view 3d object recognition. In *BMVC*, 2017.
- [191] Feng Wang, Xiang Xiang, Jian Cheng, and Alan L. Yuille. Normface:  $L_2$  hypersphere embedding for face verification. *CoRR*, abs/1704.06369, 2017.
- [192] Haohan Wang, Songwei Ge, Zachary Lipton, and Eric P Xing. Learning robust global representations by penalizing local predictive power. In *Advances in Neural Information Processing Systems*, pages 10506–10518, 2019.
- [193] Haoran Wang, Weitang Liu, Alex Bocchieri, and Yixuan Li. Can multi-label classification networks know what they don’t know? *Advances in Neural Information Processing Systems*, 2021.
- [194] Jiang Wang, Yang Song, Thomas Leung, Chuck Rosenberg, Jingbin Wang, James Philbin, Bo Chen, and Ying Wu. Learning fine-grained image similarity with deep ranking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1386–1393, 2014.
- [195] Jiangliu Wang, Jianbo Jiao, Linchao Bao, Shengfeng He, Yunhui Liu, and Wei Liu. Self-supervised spatio-temporal representation learning for videos by predicting motion and appearance statistics. *CoRR*, abs/1904.03597, 2019.
- [196] Junyan Wang, Bingzhang Hu, Yang Long, and Yu Guan. Order matters: Shuffling sequence generation for video prediction. *CoRR*, abs/1907.08845, 2019.
- [197] Pei Wang and Nuno Vasconcelos. Towards realistic predictors. In Vittorio Ferrari, Martial Hebert, Cristian Sminchisescu, and Yair Weiss, editors, *Computer Vision – ECCV 2018*, pages 37–53, Cham, 2018. Springer International Publishing.



- [198] Xiao Wang, Guangyao Chen, Guangwu Qian, Pengcheng Gao, Xiaoyong Wei, Yaowei Wang, Yonghong Tian, and Wen Gao. Large-scale multi-modal pre-trained models: A comprehensive survey. *ArXiv*, abs/2302.10035, 2023.
- [199] Xiaolong Wang and Abhinav Gupta. Unsupervised learning of visual representations using videos. *CoRR*, abs/1505.00687, 2015.
- [200] Zhaoqing Wang, Yu Lu, Qiang Li, Xunqiang Tao, Yandong Guo, Mingming Gong, and Tongliang Liu. Cris: Clip-driven referring image segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022.
- [201] Kilian Q Weinberger, John Blitzer, and Lawrence K Saul. Distance metric learning for large margin nearest neighbor classification. In *Advances in neural information processing systems*, pages 1473–1480, 2006.
- [202] Yandong Wen, Kaipeng Zhang, Zhifeng Li, and Yu Qiao. A discriminative feature learning approach for deep face recognition. In *ECCV*, 2016.
- [203] Hang Wu and May D. Wang. Training confidence-calibrated classifier via distributionally robust learning. In *2020 IEEE 44th Annual Computers, Software, and Applications Conference (COMPSAC)*, pages 295–304, 2020.
- [204] Jiajun Wu, Chengkai Zhang, Tianfan Xue, William T. Freeman, and Joshua B. Tenenbaum. Learning a probabilistic latent space of object shapes via 3d generative-adversarial modeling. *CoRR*, abs/1610.07584, 2016.
- [205] Tz-Ying Wu, Pedro Morgado, Pei Wang, Chih-Hui Ho, and Nuno Vasconcelos. Solving long-tailed recognition with deep realistic taxonomic classifier. In *European Conference on Computer Vision (ECCV)*, 2020.
- [206] Zhirong Wu, S. Song, A. Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and J. Xiao. 3d shapenets: A deep representation for volumetric shapes. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1912–1920, June 2015.
- [207] Zhirong Wu, Yuanjun Xiong, X Yu Stella, and Dahua Lin. Unsupervised feature learning via non-parametric instance discrimination. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018.
- [208] Chaowei Xiao, Jun-Yan Zhu, Bo Li, Warren He, Mingyan Liu, and Dawn Song. Spatially transformed adversarial examples. *CoRR*, abs/1801.02612, 2018.
- [209] Jianxiong Xiao, James Hays, Krista A. Ehinger, Aude Oliva, and Antonio Torralba. Sun database: Large-scale scene recognition from abbey to zoo. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 3485–3492, 2010.
- [210] Zhicheng Yan, Hao Zhang, Robinson Piramuthu, Vignesh Jagadeesh, Dennis DeCoste, Wei Di, and Yizhou Yu. Hd-cnn: Hierarchical deep convolutional neural networks for large scale visual recognition. In *International Conference on Computer Vision (ICCV)*, 2015.

- [211] Yaoqing Yang, Chen Feng, Yiru Shen, and Dong Tian. Foldingnet: Interpretable unsupervised learning on 3d point clouds. *CoRR*, abs/1712.07262, 2017.
- [212] Mang Ye, Xu Zhang, Pong C. Yuen, and Shih-Fu Chang. Unsupervised embedding learning via invariant and spreading instance feature. *CoRR*, abs/1904.03436, 2019.
- [213] Haoxuan You, Yifan Feng, Rongrong Ji, and Yue Gao. Pvnet: A joint convolutional network of point cloud and multi-view for 3d shape recognition. *CoRR*, abs/1808.07659, 2018.
- [214] Haoxuan You, Yifan Feng, Rongrong Ji, and Yue Gao. Pvnet: A joint convolutional network of point cloud and multi-view for 3d shape recognition. *CoRR*, abs/1808.07659, 2018.
- [215] Xiaoyong Yuan, Pan He, Qile Zhu, Rajendra Rana Bhat, and Xiaolin Li. Adversarial examples: Attacks and defenses for deep learning. *CoRR*, abs/1712.07107, 2017.
- [216] Zhongqi Yue, Tan Wang, Hanwang Zhang, Qianru Sun, and Xian-Sheng Hua. Counterfactual zero-shot and open-set visual recognition. In *CVPR*, 2021.
- [217] Sangdoon Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *International Conference on Computer Vision (ICCV)*, 2019.
- [218] Yuhang Zang, Wei Li, Kaiyang Zhou, Chen Huang, and Chen Change Loy. Unified vision and language prompt learning. *ArXiv*, abs/2210.07225, 2022.
- [219] Hongyi Zhang, Moustapha Cisse, Yann N. Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. *International Conference on Learning Representations*, 2018.
- [220] Jingyi Zhang, Jiaying Huang, Sheng Jin, and Shijian Lu. Vision-language models for vision tasks: A survey. *ArXiv*, abs/2304.00685, 2023.
- [221] Pengchuan Zhang, Xiujun Li, Xiaowei Hu, Jianwei Yang, Lei Zhang, Lijuan Wang, Yejin Choi, and Jianfeng Gao. Vinvl: Revisiting visual representations in vision-language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5579–5588, June 2021.
- [222] Richard Zhang, Phillip Isola, and Alexei A. Efros. Colorful image colorization. *CoRR*, abs/1603.08511, 2016.
- [223] Bin Zhao, Li Fei-Fei, and Eric P. Xing. Large-scale category structure aware image categorization. In *Advances in Neural Information Processing Systems (NIPS)*, 2011.
- [224] Zhengli Zhao, Dheeru Dua, and Sameer Singh. Generating natural adversarial examples. *CoRR*, abs/1710.11342, 2017.

- [225] Liang Zheng, Yi Yang, and Alexander G. Hauptmann. Person re-identification: Past, present and future. *CoRR*, abs/1610.02984, 2016.
- [226] Da-Wei Zhou, Han-Jia Ye, and De-Chuan Zhan. Learning placeholders for open-set recognition. In *CVPR*, pages 4401–4410, 2021.
- [227] Kaiyang Zhou, Jingkang Yang, Chen Change Loy, and Ziwei Liu. Conditional prompt learning for vision-language models. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.
- [228] Kaiyang Zhou, Jingkang Yang, Chen Change Loy, and Ziwei Liu. Learning to prompt for vision-language models. *International Journal of Computer Vision (IJCV)*, 2022.
- [229] Xinqi Zhu and Michael Bain. B-cnn: Branch convolutional neural network for hierarchical classification. *CoRR*, abs/1709.09890, 2017.