The University of Edinburgh

Masters Dissertation Report

Name the Dataset or Name the Label: Dataset Bias Investigation

Author: Junhao Sun *Supervisor: Sotirios Tsaftaris*

A thesis submitted in fulfilment of the requirements for the degree of MSc. in Signal Processing and Communications

in the

School of Engineering

August 2025

Declaration of Authorship

I, Junhao SUN, declare that this dissertation report titled, 'Name the Dataset or Name the Label: Dataset Bias Investigation' and the work presented in it is my own. Any uses made within it of the works of other authors in any form (e.g., ideas, equations, figures, text, tables, programs) are properly acknowledged at any point of their use. A list of the references employed is included.

Signed: _____

Date: _____

Abstract

This paper investigates the dataset bias in modern neural networks, revealing its dual nature: composed of both generalizable semantic information and detrimental non-semantic information. Through systematic experimentation, the research demonstrates that convolutional and transformer architectures fundamentally differ in how they propagate and transform these components of dataset bias across network layers. Convolutional networks exhibit hierarchical purification where non-semantic signals are progressively transformed into semantic representations, while transformers maintain persistent entanglement of both information types throughout their depth.

Building on Domain-Adversarial Neural Networks (DANN), two novel bias mitigation variants are proposed by incorporating Name-the-Dataset loss into the total loss function. Evaluations demonstrate consistent performance improvements: when applied to ResNet-18 trained on 273-class classification across a dataset of 170,000 images, semantic classification accuracy increases by approximately 2% absolute. Vision Transformer (ViT-B32) shows a much more modest gain of approximately 0.5% absolute accuracy under the same dataset.

Collectively, these findings establish that different architectural choices mediate bias propagation through distinct information transformation pathways, and that explicit bias suppression need not compromise semantic classification performance and can in fact enhance it. The demonstrated improvements across architectures validate dataset bias mitigation as both a robustness necessity and performance enhancement opportunity in modern deep neural networks.

Code is available at: https://github.com/hsnT9SHtAGynt/Dataset-Bias-Investigation

Acknowledgements

I would like to express my gratitude to everyone who supported and guided me throughout my MSc. research and the writing of this dissertation.

First and foremost, my sincere thanks go to my supervisor, Professor Sotirios Tsaftaris. I am profoundly grateful for his unique and invaluable mentorship. Rather than providing direct guidelines, Professor Tsaftaris consistently challenged me with insightful and probing questions throughout the project. The process of independently exploring and seeking answers to these questions was instrumental in shaping my critical thinking and ultimately defining the research direction of this thesis.

I am also immensely grateful to Dr. Steven McDonagh for his exceptional teaching in the *Machine Learning in Signal Processing* (MSc) course (2024–2025, Semester 2). Dr. McDonagh's profound expertise and clear articulation of complex machine learning concepts provided me with the essential foundational knowledge and prerequisite understanding crucial for undertaking the research presented in this dissertation.

I extend my warmest thanks to the postdoctoral researchers and PhD students who generously shared their time and insights during my project work. Specifically, I wish to thank Konstantinos Vilouras, Jingshuai Liu, Junyu Yan, and Yuning Du (listed alphabetically by first name). Their willingness to offer helpful guidance, constructive suggestions, and support throughout the research phase was greatly appreciated and significantly contributed to my work.

Lastly, my heartfelt thanks go to my family, friends, and peers for their unwavering encouragement, constant support, and steadfast belief in my capabilities. This dissertation is not only a reflection of my efforts but also a testament to the collective support and inspiration provided by all of these remarkable individuals.

Contents

List of Figures

List of Tables

Abbreviations

AP	Average Precision
BK-tree	Burkhard–Keller tree
BoW–SIFT	Bag of Words + Scale-Invariant Feature Transform
CD	Cross-Dataset index
CDG	Cross-Dataset Generalization
CC	Common Crawl
CKA	Centered Kernel Alignment
CE	Cross-Entropy
CIFAR	Canadian Institute For Advanced Research
CNN	Convolutional Neural Network
DANN	Domain-Adversarial Neural Network
DeCAF	Deep Convolutional Activation Features
DNN	Deep Neural Network
ERM	Empirical Risk Minimization
FFT	Fast Fourier Transform
GRL	Gradient Reversal Layer
HOG	Histogram of Oriented Gradients
HSIC	Hilbert–Schmidt Independence Criterion
I.I.D.	Independently and Identically Distributed
JSON	JavaScript Object Notation
JPEG	Joint Photographic Experts Group
LLM	Large Language Model
LSH	Locality-Sensitive Hashing
MMD	Maximum Mean Discrepancy
MLP	Multi-Layer Perceptron

NLP	Natural Language Processing
O.O.D.	Out-of-Distribution
OHM	One-Hot MiniMax
REPAIR	REPresentation Bias Removal
SAD	Self-Attention Difference
SIFT	Scale-Invariant Feature Transform
SVM	Support Vector Machine
UDM	Uniform Double Mini
VEVENT	iCal `VEVENT` component
ViT	Vision Transformer
VLM	Vision-Language Model
VQA	Visual Question Answering
YFCC	Yahoo Flickr Creative Commons

Symbols

N	Number of samples / mini-batch size / number of image patches (ViT)
K	Number of datasets (classes/sources)
$\mathbf{x}_i$	i-th input sample
$\boldsymbol{x}$	Image sample (pixel values)
X	Space of observed features
$\boldsymbol{y}$	Semantic label (one-hot vector)
Y	Set of semantic labels
y_i	Label (class or dataset index) of $\mathbf{x}_i$
$f_\theta(\mathbf{x}_i)$	Classifier output for $\mathbf{x}_i$
$\mathbf{1}[\cdot]$	Indicator function
Acc	Classification accuracy
$\mathrm{AP}_{\mathrm{self}}$	Average precision on source dataset
$\mathrm{AP}_{\mathrm{target}}$	Average precision on target dataset
Drop	Relative drop in AP (see eq. 2.5.2)
α	Sample-value coefficient
CD	Cross-dataset index (eq. 2.5.4)
$H(X)$	Shannon entropy of random variable X
$H(Y)$	Entropy of label distribution
$H(Y \mid X = c)$	Conditional entropy of Y given $X = c$ (eq. 2.5.5)
$I\big(b(X); f(X)\big)$	Mutual information between bias $b(X)$ and features $f(X)$
$I(\cdot;\cdot)$	Mutual information (generic)
$\mathrm{Ent}_\mu(f)$	Functional entropy under measure μ (eq. 2.5.10)
$p(y \mid X = c)$	Conditional probability of y given $X = c$
$p(b \mid f(X))$	Conditional probability of bias b given features
p_i	Empirical probability of category i

p'_{y_i}	Reweighted class prior for label y_i
P_{dataset}	Empirical dataset distribution
P_{real}	True real-world distribution
$\boldsymbol{u}$	Uniform target distribution vector
$\text{KL}(p\|q)$	Kullback–Leibler divergence
$\mathscr{D}(\cdot\|\cdot)$	Divergence between two distributions (e.g., KL, TV)
$\boldsymbol{d}$	Dataset source identifier (one-hot vector)
D	Set of dataset sources / dimension of observed feature space / embedding dimensio
$\mathscr{D}$	Dataset (set of samples)
$\boldsymbol{g}$	Generalizable (latent) features
$\mathscr{G}$	Space of generalizable features
$\boldsymbol{b}$	Dataset-bias shortcuts (latent)
$\mathscr{B}$	Space of dataset-bias shortcuts
ϕ	Feature representation / generative mixing function
$\mathscr{B}(\mathscr{D}, \phi)$	Dataset-bias measure w.r.t. ϕ
$\mathscr{R}^*(\mathscr{D}, \phi)$	Minimum empirical risk with representation ϕ
f	Neural network mapping input to logits
$f^{(y)}$	Label-classification head
$f^{(d)}$	Dataset-classification head
h	Feature mapping on $\boldsymbol{g}$
k	Feature mapping on $\boldsymbol{b}$
φ	Coupling function combining h and k
$f_\theta(x)$	Feature extractor output for x
$g_\phi(\cdot)$	Main-task classifier
$h_\psi(\cdot)$	Bias (adversary) classifier
$\boldsymbol{z}$	Pre-softmax logits
$\hat{\boldsymbol{y}}$	Predicted label distribution (post-softmax)
$\hat{\boldsymbol{d}}$	Predicted dataset distribution (post-softmax)
softmax	Softmax normalization function
ℓ	Loss function (e.g., cross-entropy)
CE	Cross-entropy function
$\mathscr{L}$	Loss function (generic)
$\mathscr{L}_{\text{std}}$	Standard training loss

$\mathscr{L}_{\text{y}}$	Name-the-Label loss
$\mathscr{L}_{\text{d}}$	Name-the-Dataset loss
$\mathscr{L}_{\text{minimax}}$	Minimax objective loss
$\mathscr{L}_{\text{u}}$	Uniform alignment loss for dataset head
$\mathscr{L}_{\text{UDM}}$	Uniform Double Mini combined loss
λ, β, γ	Trade-off / schedule coefficients
λ	Trade-off hyperparameter
$\lambda_{\max}$	Maximum adaptation coefficient
p	Normalized training progress (e/E)
e	Current epoch index
E	Total number of epochs / Shannon evenness index (when context makes clear)
η_f, η_y, η_d	Learning rates (feature-, label-, dataset-head)
w_i	Weight of example i in re-weighted dataset
$D_{\text{real}}, D_{\text{syn}}$	Real/synthetic dataset subsets
$\lambda_g, \lambda_l, \lambda_q$	Loss weights (global, local, quality)
$g_T(x,k), g_S(x,k)$	Teacher / student explanation maps
$d(\cdot,\cdot)$	Explanation-alignment distance (eq. 2.6.1)
M	Number of scales in explanation loss
X_S, X_T	Source and target domain sample sets
$\text{MMD}(X_S, X_T)$	Maximum Mean Discrepancy
x, y	Input/output in ResNet block
$\mathscr{F}(x, \{W_i\})$	Residual function in ResNet ($y = \mathscr{F}(x) + x$)
$X^{(l)}$	Representation at Transformer layer l
MSA	Multi-head self-attention sub-layer
LN	Layer normalization
MLP	Feed-forward sub-layer in Transformer
L	Number of Transformer encoder layers
[`CLS`]	Classification token (ViT)
Δ_{sys}	Generalization gap (systematic deviation)
$\text{Bias}_{\text{output}}(\boldsymbol{x})$	Point-wise output bias
S	Number of discrete attribute bins
$\sum$	Summation operator
$\mathbb{E}$	Expectation operator

Dedicated to my family for their unwavering love and support ...

Chapter 1

Introduction

1.1 The Inevitability of Dataset Bias

It is a truth universally acknowledged that a dataset that is collected, no matter how exhaustive, represents only a sampled approximation of the real world. Inevitably, the sampling process introduces distributional discrepancies between the dataset and the complete reality it seeks to represent. This limitation manifests as *dataset bias* [Torralba and Efros, 2011]—the mismatch between a dataset's empirical distribution and the true real-world distribution—which causes models trained on such data to produce outputs that systematically diverge from actual real-world behavior. Specifically, a model may achieve strong performance on an I.I.D. (Independently and identically distributed) test set yet exhibit a pronounced drop in accuracy when evaluated on O.O.D. (out-of-distribution) data [Geirhos et al., 2020].

Much like the classic *nature versus nurture* debate [Turkheimer, 2000] in biology (are human's behaviors driven by innate genetics or shaped by environment?), a model's performance depends critically on two complementary factors: architectural (*nature*) and training (*nurture*). When the architecture lacks sufficient capacity, even high-quality data cannot reveal complex patterns; conversely, when training data contains distributional discrepancies, even the most sophisticated architectures produce biased outputs.

1.2 The Persistence of Dataset Bias

Despite remarkable advances in deep learning architectures and training methodologies, dataset bias remains a persistent challenge. As Liu and He [2025] comprehensively

demonstrate, dataset bias not only persists but often intensifies in modern deep neural networks. This resilience occurs probably because neural architectures inherently transform and propagate dataset biases rather than eliminating them; indeed, [Zeng et al., 2024] empirically confirm that these bias signatures remain robust across diverse data transformations.

1.3 The Dual Nature of Dataset Bias

Recent work by Liu and He [2025] has revealed that dataset bias is not solely detrimental noise, but also *contains generalizable and transferable semantic information*—for example, features learned to discriminate dataset origin yield non-trivial gains when fine-tuned on ImageNet classification. Inspired by the finding, the present investigation designed a *non-semantic test* experiment to demonstrate that that dataset bias also encodes *non-semantic information*—such as interpolation fingerprints, JPEG compression residues, and sensor-specific noise patterns—that serve as *spurious shortcuts* exploited by modern deep neural networks.

Consequently, dataset bias exhibits a *dual nature*:

- **Transferable semantic patterns** that can enhance downstream tasks when appropriately harnessed;
- **Detrimental non-semantic shortcuts** that inflate in-domain performance yet impair out-of-distribution generalization.

1.4 Aim and Objectives

This dissertation aims to deliver a comprehensive investigation into dataset bias in modern deep neural networks, with four primary objectives:

1. **Objective 1:** Can modern deep neural networks still capture dataset bias? (Chapter 4,7)
2. **Objective 2:** What components constitute dataset bias, and which dominate? (Chapter 4,7)
3. **Objective 3:** How do model architectures mediate dataset bias propagation? (Chapter 4,7)

4. **Objective 4:** What is Dataset Bias? How Can Dataset Bias Be Mitigated? What Challenges Remain? (Chapters 6,7)

1.5 Unique Contributions

This dissertation makes the following principal contributions:

1. **Dual Nature Dataset Bias Characterisation** showing that dataset bias primarily consists of two components—non-semantic information and semantic information. In certain cases, non-semantic information dominates the dataset bias; however, when non-semantic information is attenuated, semantic information can emerge as a significant secondary source of dataset identifiability.
2. **Architectural Mediation Insights** revealing that convolutional and transformer architectures differ fundamentally in how they propagate and transform these bias components across layers.
3. **Theoretical Framework** describe dataset bias as a divergence between empirical and real-world distributions and formalising its impact on model behavior.
4. **Mitigation Methods** introducing and validating One-hot MiniMax and Uniform Double Mini bias mitigation methods, which substantially reduce dataset bias strength while improving semantic classification performance.

1.6 Thesis Structure

The remainder of this dissertation is organized as follows:

- **Chapter 2** presents a comprehensive literature review on the origins, mechanisms, quantification and mitigation methods of dataset bias.
- **Chapter 3** details the methodology, including dataset construction, network architectures, and training protocols.
- **Chapter 4** reports experiments investigating dataset bias persistence and component decomposition.
- **Chapter 5** develops a mathematical framework formalizing dataset bias and its effect on generalization.

- **Chapter 6** introduces and evaluates mitigation strategies based on adversarial and uniform alignment objectives.
- **Chapter 7** provides a global discussion synthesizing findings, highlighting practical implications, and outlining persisting challenges.
- **Chapter 8** concludes with a synthesis of the key findings, practical recommendations, and an agenda for future work aimed at advancing bias-aware deep neural networks.

Chapter 2

Literature Review

2.1 Sectional Introduction

This chapter presents a comprehensive review of the literature concerning dataset bias in machine learning systems. The examination begins by tracing the historical development of dataset bias research, establishing its foundational concepts and seminal contributions. Subsequent sections analyze the complex mechanisms through which dataset bias manifests in modern deep neural networks, including its *dual nature (Semantic-Transportability Paradox) as both a source of detrimental shortcuts and potentially transferable patterns.*

Various methodological approaches for quantifying dataset bias are systematically surveyed, ranging from classical "Name That Dataset" accuracy to information-theoretic measures. The review then synthesizes contemporary mitigation strategies, categorizing them into data-centered and model-centered approaches while critically evaluating their respective strengths and limitations.

Significant research gaps and persistent challenges are identified, including fragmented evaluation protocols, the semantic-transportability paradox, and incomplete mechanistic understanding of bias propagation. The chapter concludes by positioning the present work within this research landscape, highlighting how it addresses several outstanding limitations through hierarchical dataset bias deconstruction, cross-layer linear probing and novel dual-objective mitigation strategies.

2.2 Historical Perspective and Foundational Work

The study of dataset bias has its origins in the early 2010s, when pioneering research began to expose how hidden structures in data significantly affect the generalization performance of machine learning models. One of the most influential works in this area was the "Name That Dataset" experiment by Torralba and Efros [2011], which revealed that shallow models trained on hand-crafted features such as Histogram of Oriented Gradients (HOG) and color histograms, combined with Support Vector Machines (SVMs), could successfully predict the dataset from which an image originated, even though that information was not provided during training. This demonstrated that datasets contain unique dataset-specific signatures,which is called *dataset bias*, making models vulnerable to overfitting to dataset-specific features rather than learning generalizable features.

Building upon this finding, Khosla et al. [2012] proposed a discriminative framework to explicitly model dataset bias. Their approach introduced dataset-specific parameters to complement a shared representation of the visual world, improving cross-dataset generalization performance to some extent. Similarly, Fang et al. [2013] advanced this line of research with their "Unbiased Metric Learning" method, which exploited multiple datasets and web images to construct more generalizable metric spaces, thereby mitigating dataset biases to some extent.

The advent of deep learning extended these early insights into a new era. Tommasi et al. [2017] revisited the dataset bias problem using deep convolutional features (DeCAF), showing that despite the superior representational power of deep models, dataset bias remained pervasive. Their experiments confirmed that most debiasing methods available at the time only partially mitigated the issue, indicating the resilience of bias even in more sophisticated feature spaces.

More recently, Liu and He [2025] offered a comprehensive retrospective in their work *A Decade's Battle on Dataset Bias: Are We There Yet*. Notably, they were among the first to suggest that dataset bias may also contain *generalizable and transferrable patterns*, opening a nuanced discussion about when bias can inadvertently benefit learning. They further highlighted that modern architectures such as ResNets and Vision Transformers have not eliminated dataset bias; in fact, in some cases, the problem has intensified [Hall et al., 2022].

Extending these insights, Zeng et al. [2024] presented *Understanding Bias in Large-Scale Visual Datasets*, one of the most comprehensive empirical investigations of dataset bias to date. The study revealed that, even after a wide range of low-level transformations—such as semantic segmentation, edge detection, depth estimation, and frequency filtering—modern models retained a strong ability to predict dataset origin. Following

the discovery of semantic information contained in dataset bias by Liu and He [2025], this work was the first to integrate Large Language Models (LLMs) into the study of dataset bias, thereby enabling a large-scale semantic perspective on the problem. By coupling ConvNeXt-based visual classifiers with LLM-driven semantic interpretation, the authors systematically examined multiple large-scale datasets, including YFCC, CC, and DataComp. Their findings highlighted that dataset bias is not merely tied to superficial visual cues but is deeply embedded in semantic and structural properties. Moreover, the experiments showed that synthetic images generated by modern diffusion models inherit these semantic and structural biases, underscoring the risks of treating synthetic data as a straightforward remedy for dataset bias.

Together, these foundational studies outline a clear trajectory: from the early recognition of dataset specific signatures in shallow models, through initial attempts to explicitly model and mitigate bias, to the realization that modern architectures and even generative data pipelines remain deeply affected by dataset bias. The recent integration of LLM-based semantic analysis marks a new frontier, highlighting not only the persistence of dataset bias but also the pressing need for more holistic frameworks that address its complex semantic and structural dimensions.

2.3 Dataset Bias Mechanism in the Deep Learning Era

2.3.1 Dataset Bias as a Source of Generalizable and Transferrable Patterns

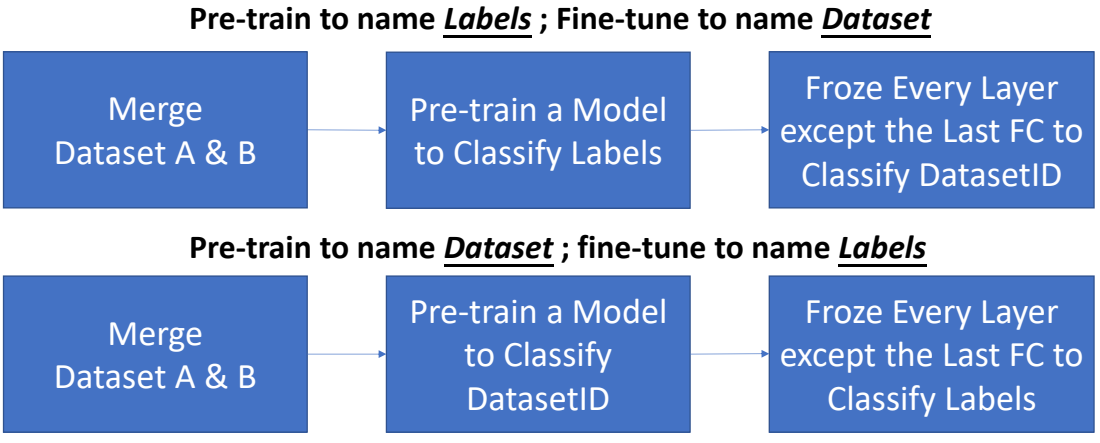


FIGURE 2.1: Workflow to Illustrate Dataset Bias Also Contain Some Semantic information that May be Generalizable and Transferable to Downstream Image Label Classification Tasks (Liu and He [2025])

Most recent research has uncovered that dataset bias, while often detrimental to fairness and robustness, may also encode generalizable and transferrable patterns. These patterns carry semantic information that can be leveraged for downstream image classification tasks, suggesting that bias is not purely noise but can contribute meaningful representations [Liu and He, 2025, Zeng et al., 2024].

Empirical studies provide two main lines of support for this perspective:

- **Dataset classification as a pretrained task uncovers transferrable semantic features.**

  The "Name That Dataset" experiment by Torralba and Efros [2011] demonstrated that a model pre-trained for semantic classification could still successfully identify the dataset of origin of an image when all layers were frozen except the new replaced final fully connected layer to classify dataset origin.

  In a complementary inversion of this idea (Fig.2.2), Liu and He [2025] showed that models pre-trained explicitly to classify dataset origin of an image were able to transfer the learned features to semantic classification tasks when only the new replaced final fully connected layer was retrained to do semantic classification. That is, their experiments revealed that features acquired during dataset classification yielded non-trivial semantic classification improvements on ImageNet-1K compared to random baselines. Furthermore, incorporating a broader variety of datasets in the pre-train task further enhanced performance, indicating that the discovery of dataset biases may facilitate the learning of semantically rich and transferrable representations.

- **Semantic and structural biases underpin persistent transferability across data modalities.**

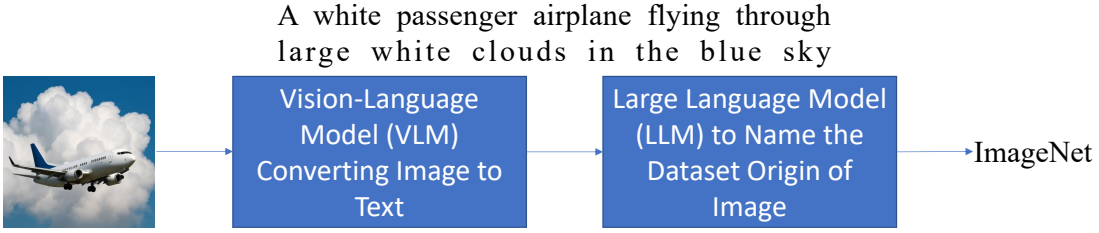


FIGURE 2.2: Workflow to Illustrate Dataset Bias Also Contain Some Semantic information that May be Generalizable and Transferable to Downstream Image Label Classification Tasks (Zeng et al. [2024])

Zeng et al. [2024] conducted a detailed analysis of large-scale visual datasets using techniques such as edge detection, depth estimation, and frequency filtering.

Even when low-level visual cues were removed, models retained the ability to identify dataset origins, indicating that dataset bias is also rooted in semantics rather than superficial features only. *To further probe these biases, the authors employed Vision-Language Models (VLMs) to convert images into descriptive textual representations and subsequently leveraged Large Language Models (LLMs) to infer the dataset of origin based on these descriptions.* This two-stage framework demonstrated that semantic bias encode contextually meaningful signals sufficient for accurate dataset identification. Moreover, the findings showed that these semantic-level patterns are directly relevant to downstream classification tasks, confirming the *transferrable nature of dataset bias across both visual and textual modalities.*

2.3.2 Dataset Bias Induces the Poor Generalization of Models to Unseen Scenarios

A widely acknowledged phenomenon in contemporary deep learning research is that dataset bias significantly impairs model generalization to unseen scenarios, particularly in out-of-distribution (O.O.D.) contexts [Geirhos et al., 2020, Liu et al., 2021], resulting in systematic errors frequently attributed to shortcut learning. This phenomenon has been shown to be robust across various model architectures, dataset combinations, and training methodologies [Geirhos et al., 2020, Liu et al., 2021, Liu and He, 2025, Mehrabi et al., 2021, Tommasi et al., 2017, Zeng et al., 2024].

The mechanisms by which dataset bias undermines O.O.D. generalization have been analyzed extensively in the literature, and the following evidence strongly supports this viewpoint:

- **Shortcut learning as the underlying mechanism of O.O.D. failures.** Geirhos et al. [2020] demonstrated that neural networks often rely on superficial correlations such as background textures or dataset-specific artifacts. This reliance constitutes of dataset bias induced shortcut learning, directly explaining why models that perform well on I.I.D. test sets collapse when exposed to O.O.D. scenarios.
- **Deep representations do not inherently overcome dataset bias.** Through cross-dataset experiments, Tommasi et al. [2017] showed that convolutional features (DeCAF) improved within-dataset performance but failed to sustain accuracy in novel test sets. This finding confirms that even advanced deep neutral networks cannot prevent O.O.D. degradation, reinforcing the robustness of the phenomenon across model architectures.

- **Bias persists despite larger and more diverse datasets.** Revisiting the "Name That Dataset" experiment, Liu and He [2025] reported that modern networks achieved over 84% accuracy in identifying dataset origin from large-scale collections such as YFCC, CC, and DataComp. The result indicates that scaling up dataset size and diversity does not eliminate bias, and instead reveals that dataset biases remain systematically exploitable across dataset combinations.

- **Bias drives systematic errors in sensitive real-world applications.** According to Mehrabi et al. [2021], dataset bias is a key factor behind systematic failures in domains such as healthcare and recruitment. This demonstrates that the negative effect of bias on O.O.D. generalization is not limited to academic benchmarks but extends to high-stakes decision-making contexts.

- **Traditional Training strategies amplify the vulnerability under O.O.D. shifts.** Liu et al. [2021] highlighted that empirical risk minimization (ERM) tends to exploit all correlations present in training data, including biased ones. Consequently, ERM-trained models often generalize poorly under distribution shifts, sometimes performing worse than random guessing, showing that the phenomenon is robust across training methods.

2.3.3 No Unbiased Dataset and Dataset Bias Is Difficult to Identify or Eliminate

A prevailing consensus in the literature is that there is no truly unbiased benchmark dataset[Fabbrizzi et al., 2022]. Biases are often difficult to identify or predict, with many forms remaining imperceptible to human observers and the concrete forms of the bias captured by neural networks remaining largely unclear. Moreover, simply enlarging or merging datasets seldom eliminates bias and may even amplify it. [Fabbrizzi et al., 2022, Hall et al., 2022, Liu et al., 2021, Liu and He, 2025, Mehrabi et al., 2021].

The persistence and complexity of dataset bias have been widely documented. The following findings illustrate how this viewpoint is consistently supported across multiple strands of research:

- **Bias amplification through training exacerbates existing imbalances.** Hall et al. [2022] demonstrated that neural networks not only inherit but also amplify existing biases in training data. Their controlled experiments revealed that larger models and certain training regimes exacerbate these effects, showing that increasing dataset size or model capacity does not resolve the issue but can make it worse.

This finding directly supports the claim that dataset bias can be intensified rather than eliminated through scaling.

- **Large-scale and diverse datasets remain deeply biased.** Revisiting the "Name That Dataset" paradigm, Liu and He [2025] found that modern models could achieve over 84% accuracy in identifying dataset origins across YFCC, CC, and DataComp. This demonstrates that even large and diverse datasets preserve dataset-specific signatures, invalidating the assumption that merging or expanding datasets naturally reduces bias.

- **Bias forms are often imperceptible and difficult to identify in advance.** According to Fabbrizzi et al. [2022], visual datasets suffer from selection, framing, and labeling biases, many of which are subtle and not easily detectable by human. These hidden distortions show that certain biases can hardly be reliably anticipated or manually corrected.

- **Bias pervades high-stakes applications, reinforcing its unpredictability.** Mehrabi et al. [2021] emphasized that biases embedded in training data often resurface as systematic errors in domains such as healthcare and recruitment. Their analysis highlights that these errors arise from feedback loops and subtle correlations that are difficult to perceive, underscoring the unpredictability of bias in practical systems.

2.4 Model Architectures & Dataset Bias Interaction

The interplay between model architectures and dataset bias represents a critical dimension in understanding how dataset biases manifest and propagate in modern deep neural networks. Rather than being passive recipients of data statistics, neural architectures actively transform, amplify, or mitigate inherent dataset biases through their structural properties and learning dynamics. This section synthesizes empirical evidence demonstrating how architectural choices-from convolutional inductive biases to attention mechanisms—mediate the relationship between biased data inputs and model outputs.

2.4.1 Bias Amplification Mechanisms

Fundamental work by Hall et al. [2022] establishes that bias amplification is neither inevitable nor random, but systematically modulated by architectural capacity. Their controlled experiments reveal a "V-shaped" relationship: both under-parameterized models

(lacking representational capacity) and over-parameterized models (prone to overconfidence) amplify synthetic biases more than optimally balanced architectures. Crucially, amplification peaks during early training when models exploit "easy" spurious correlations before learning robust features. Complementary evidence from NLP shows transformer architectures actively reshape bias through attention mechanisms. Talebpour et al. [2025] demonstrates that bias amplification occurs at distinct architectural locations: upper layers in BERT-style encoders versus lower layers in autoregressive models like GPT, quantified through their novel Self-Attention Difference (SAD) metric.

2.4.2 Feature Learning and Dataset Signatures

The sensitivity of architectures to dataset bias persists even in modern deep learning. Tommasi et al. [2017]'s cross-dataset analysis revealed that deep features (DeCAF) actually *enhanced* dataset identifiability compared to handcrafted features, indicating that convolutional architectures excel at learning dataset fingerprints. This finding is reinforced by Zeng et al. [2024], whose transformation-based framework shows contemporary ConvNeXt architectures exploit subtle cues including color statistics (84.5% accuracy), local patches (81%), and structural contours (73%) for dataset identification. Together, these studies indicate that architectural prowess in pattern recognition inherently includes learning non-semantic dataset signatures.

2.4.3 Architectural Inductive Biases

Core architectural components fundamentally shape bias susceptibility. Convolutional networks exhibit strong texture bias when trained on ImageNet Geirhos et al. [2019], classifying cue-conflict images by texture 97.7% of the time versus 1.7% for humans. This bias stems not from inherent CNN properties but from training data interactions: Hermann et al. [2020] proved texture bias primarily originates from aggressive cropping augmentations that disrupt global shape information. When CNNs train on texture-depleted data (Stylized-ImageNet), they develop human-like shape bias and improved robustness. Vision Transformers (ViTs), by contrast, exhibit fundamentally different representation structures. Raghu et al. [2021]'s CKA analysis[1]revealed that ViTs maintain uniform representation similarity across layers (mean CKA: 0.75) versus CNNs' hierarchical progression (mean CKA: 0.32), enabling early global information integration that alters bias acquisition pathways.

[1]CKA (Centered Kernel Alignment) is a statistical method for quantifying the similarity of hidden layer representations in neural networks. It measures the geometric similarity of two layers by computing the Hilbert-Schmidt Independence Criterion (HSIC) of the Gram matrices of their representations. This method is rotation-invariant and scale-invariant, enabling meaningful comparisons across different architectures or layers.

2.4.4 Scale and Contextual Mismatch

Architectural bias manifests when training-test contexts diverge. Herranz et al. [2016] identified scale-induced dataset bias: ImageNet-pre-trained CNNs excel at object-scale patches (84% accuracy) but underperform at scene scales (62%), while Places-trained models show the inverse pattern. This stems from architectural specialization to training-data statistics—ImageNet contains 3× larger objects than scene datasets. Their hybrid architecture solution, which routes different scales to specialized backbones, increased scene recognition accuracy by 8.2%, demonstrating that architectural adaptation to data statistics mitigates bias.

2.4.5 Synthesis and Research Implications

Collectively, these studies reveal architecture as both amplifier and attenuator of dataset bias. Key mechanisms include:

- **Capacity-accuracy tradeoffs**: Optimal bias mitigation occurs at intermediate model capacities Hall et al. [2022]
- **Representation geometry**: ViTs' uniform feature similarity versus CNNs' hierarchy creates distinct bias propagation pathways Raghu et al. [2021]
- **Attention mechanics**: Self-attention layers actively reshape bias with architecture-specific patterns Talebpour et al. [2025]
- **Augmentation-architecture coupling**: Data transformations interact with architectural biases to determine feature preferences Geirhos et al. [2019], Hermann et al. [2020]

These findings necessitate architecture-aware bias mitigation strategies and underscore that debiasing may require co-design of data processing pipelines and model architectures. Fundamental tensions remain between accuracy optimization and bias minimization, particularly when biased features provide predictive shortcuts. The evidence establishes that architectural interventions—from hybrid designs [d'Ascoli et al., 2021, Herranz et al., 2016] that inject or modulate inductive biases, to representation geometry constraints [Bardes et al., 2021] that regularize the structure of the learned feature space and thereby reduce over-reliance on spurious correlations—offer promising pathways toward bias-resilient systems.

2.5 Dataset Bias Quantification Methods

2.5.1 Name the Dataset Accuracy Method

The most common approach to quantifying dataset bias is the "Name the Dataset Accuracy" (Torralba and Efros [2011]), which trains a classifier to identify the source dataset of a given image. The performance of this classifier serves as a direct measure of the discernible, systematic differences—or "fingerprints"—between datasets. The accuracy of such a classifier is formally defined as:

$$\text{Acc} = \frac{1}{N} \sum_{i=1}^{N} \mathbf{1}\big[f_\theta(\mathbf{x}_i) = y_i\big], \tag{2.5.1}$$

where:

- f_θ denotes the classifier with parameters θ that maps an input to a dataset index.
- $\mathbf{x}_i$ represents the i-th input sample from a validation set composed of images from all datasets.
- $y_i \in \{1, \ldots, K\}$ is the ground-truth index of the dataset from which $\mathbf{x}_i$ was sourced.
- K indicates the total number of datasets being distinguished.
- N denotes the total number of samples of in the train/validation/test set.
- $\mathbf{1}[\cdot]$ is the indicator function, which returns 1 if the condition inside is true and 0 otherwise.

A resulting accuracy significantly higher than the chance level $(1/K)$ indicates the presence of quantifiable dataset bias.

2.5.2 Cross-Dataset Generalization Methods

Cross–dataset generalization (CDG) measures how strongly a model's performance deteriorates when it is transferred from its *source* dataset to an ostensibly related *target* dataset. Torralba and Efros [2011] first quantified this phenomenon across six classical image corpora by reporting the *percentage performance drop*

$$\text{Drop} = \frac{\text{AP}_{\text{self}} - \text{AP}_{\text{target}}}{\text{AP}_{\text{self}}}. \tag{2.5.2}$$

- $\mathrm{AP}_{\mathrm{self}}$ — average precision (AP) obtained when training and testing on the same (*source*) dataset.
- $\mathrm{AP}_{\mathrm{target}}$ — AP achieved when the model trained on the source dataset is evaluated on a distinct *target* dataset.
- Drop — relative decline in AP; larger values denote stronger dataset bias.

The same study introduced the *sample-value coefficient* α, implicitly defined by

$$\mathrm{AP}_j^j(n) = \mathrm{AP}_i^j\left(\frac{n}{\alpha}\right), \tag{2.5.3}$$

which estimates how many additional source dataset images are needed to match target dataset accuracy, namely the relative worth of images from dataset i in terms of dataset j's accuracy.

- i— index of the *source* dataset.
- j— index of the *target* dataset.
- n— number of positive training examples drawn from the source dataset in the comparison.
- α — sample-value coefficient; $\alpha > 1$ implies that each source sample is less informative for the target domain than a native target sample.
- $\mathrm{AP}_j^j(n)$ — AP on dataset j when the model is trained on dataset j with n positives.
- $\mathrm{AP}_i^j(\frac{n}{\alpha})$ — AP on dataset j when the model is trained on dataset i with n/α positives.

Later, Tommasi et al. [2017] revisited CDG, contrasting BoW–SIFT with CNN features on twelve datasets and introducing a continuous *cross-dataset (CD) index*

$$\mathrm{CD} = \left(1 + e^{-\frac{\mathrm{AP}_{\mathrm{self}} - \mathrm{AP}_{\mathrm{target}}}{100}}\right)^{-1}, \tag{2.5.4}$$

which maps the raw performance gap into $(0, 1)$ to facilitate cross-experiment comparison.

- $\mathrm{AP}_{\mathrm{self}}$, $\mathrm{AP}_{\mathrm{target}}$ — defined as above.
- 100 — scaling constant chosen empirically to spread values within the sigmoid's sensitive region.
- CD — bounded score; CD > 0.5 signals pronounced dataset bias, while values near 0.5 denote weaker bias.

2.5.3 Shannon Entropy Methods to Quantify Dataset Bias

A variety of approaches leverage Shannon entropy and related information-theoretic measures to assess and mitigate bias in visual datasets. The following sections summarize four representative methods, detailing their key formulas and the meaning of each variable.

2.5.3.1 Conditional Entropy for Category-Level Bias Quantification

Dataset bias may manifest when certain object or scene categories occur almost exclusively with a particular label. Panda et al. [2018] compute the conditional Shannon entropy of the label distribution given each category:

$$H(Y \mid X = c) \;=\; -\sum_{y \in \mathcal{Y}} p(y \mid X = c)\, \log p(y \mid X = c) \tag{2.5.5}$$

- X : random variable representing the detected category (e.g. an object or scene label).
- c : a specific category value (e.g. "beach", "car").
- Y : random variable denoting the target label (e.g. emotion positive/negative).
- $\mathcal{Y}$: set of possible labels (often binary, e.g. $\{e_p, e_n\}$).
- $p(y \mid X = c)$: empirical probability of label y given category c, estimated by counting labeled samples.

Low (or zero) values of $H(Y \mid X = c)$ indicate that category c almost deterministically predicts Y, revealing a strong dataset leakage from X to Y.

Quantification Example: In the Deep Emotion dataset the object category "*balloon*" appears exclusively in the negative set for *sadness*. Hence $p(e_p \mid X = \text{balloon}) = 0$ and $H(Y \mid X = \text{balloon}) = 0$, signalling maximal bias. For instance, 30 % of object categories for *sadness* exhibit zero entropy, corroborating severe imbalance.

2.5.3.2 Mutual Information for Quantifying Feature–Bias Leakage

Dataset bias at the feature level can be quantified by the mutual information between the learned representation and a ***known*** bias attribute Kim et al. [2019]. Mutual information is defined as

$$I\big(b(X)\,;\, f(X)\big) \;=\; H\big(b(X)\big) \;-\; H\big(b(X) \mid f(X)\big), \tag{2.5.6}$$

with the conditional entropy

$$H\big(b(X) \mid f(X)\big) \;=\; -\sum_{b\in\mathscr{B}} p\big(b \mid f(X)\big)\,\log p\big(b \mid f(X)\big). \tag{2.5.7}$$

- $b(X)$ — bias attribute of sample X (e.g. colour red/green).
- $f(X)$ — feature vector extracted by the network.
- $\mathscr{B}$ — finite set of bias values (e.g. $\{\text{red}, \text{green}\}$).
- $p(b \mid f(X))$ — probability assigned by an estimator to bias b given features $f(X)$.
- $H(\cdot)$, $I(\cdot\,;\,\cdot)$ — Shannon entropy and mutual information.

Quantification Example (Colored MNIST):

1. *Bias prior entropy:* In the training data colours are balanced, so

   $$p(\text{red}) = 0.5,\; p(\text{green}) = 0.5 \;\Rightarrow\; H\big(b(X)\big) = -2 \times 0.5 \log_2(0.5) = 1.0 \text{ bit/sample}.$$

2. *Conditional entropy of bias given features:* A baseline feature extractor allows 95% accuracy in predicting colour:

   $$\begin{aligned} & p(\text{red} \mid f) = 0.95, \quad p(\text{green} \mid f) = 0.05 \\ \Longrightarrow \quad & H\big(b(X) \mid f(X)\big) = -0.95 \log_2(0.95) - 0.05 \log_2(0.05) \\ & \approx 0.286 \text{ bits/sample}. \end{aligned} \tag{2.5.8}$$

3. *Mutual information (bias strength):*

   $$I\big(b(X)\;;\; f(X)\big) = 1.0 - 0.286 = 0.714 \text{ bits/sample}.$$

A value of 0.714 bits/sample indicates substantial leakage of the colour attribute into the learned features, directly quantifying specific types of dataset bias.

2.5.3.3 Shannon Diversity Index for Attribute-Level Imbalance

Shannon entropy provides a global measure of how evenly an attribute's discrete values are represented, thereby quantifying dataset bias Merler et al. [2019]. For an attribute with S categories the H *diversity index* and its normalised counterpart, E *evenness*, are

$$H \;=\; -\sum_{i=1}^{S} p_i \,\log p_i, \qquad E \;=\; \frac{H}{\log S}, \tag{2.5.9}$$

where natural logarithms are used in the original paper; base-2 logs would express H in bits.

- S — number of discrete bins for the attribute (e.g. six Fitzpatrick skin-tone groups).
- p_i — empirical probability (frequency divided by sample count) for category i.
- H — Shannon diversity; maximal when all $p_i = 1/S$.
- $E \in [0, 1]$ — evenness; equals 1 only under perfect balance.

E

Quantification Example (Fitzpatrick skin tone): Consider a face dataset with the category counts

$$[\,400{,}000,\ 300{,}000,\ 150{,}000,\ 100{,}000,\ 40{,}000,\ 10{,}000\,]$$

out of $N = 1\,000{,}000$ images. Then

$$p = \big[0.40,\ 0.30,\ 0.15,\ 0.10,\ 0.04,\ 0.01\big], \quad H = -\sum_{i=1}^{6} p_i \ln p_i = 1.42 \text{ nats/sample},$$

$$E = \frac{1.42}{\ln 6} = 0.79.$$

For comparison, a perfectly balanced set ($p_i = 1/6$) would yield $H_{\max} = \ln 6 = 1.79$ nats/sample and $E = 1.00$. The observed evenness of 0.79 therefore exposes a moderate but non-trivial bias favouring lighter tones, which may translate into performance disparities in downstream face-analysis tasks.

2.5.3.4 Functional Entropy for Modality-Bias Quantification

Information imbalance across input modalities can be quantified through the *functional entropy* of a prediction function under small stochastic perturbations Gat et al. [2021]. Let $x \in \mathbb{R}^d$ be a multi-modal input decomposed into M blocks $\{x^{(m)}\}_{m=1}^{M}$ (e.g. image and text). For a square-integrable function $f : \mathbb{R}^d \to \mathbb{R}_{\geq 0}$ and a product Gaussian measure $\mu = \mathcal{N}(x, \sigma^2 I)$, functional entropy is

$$\mathrm{Ent}_\mu(f) \;=\; \int f(z)\,\log f(z)\,d\mu(z) \;-\; \Big(\int f(z)\,d\mu(z)\Big)\log\Big(\int f(z)\,d\mu(z)\Big). \tag{2.5.10}$$

- z — perturbed sample drawn from μ ($z = x + \varepsilon,\ \varepsilon \sim \mathcal{N}(0, \sigma^2 I)$).

- $f(z)$ — non-negative score; in practice the cross-entropy between the soft-max outputs at x and z as in Gat et al. [2021].
- μ — perturbation distribution; variance σ^2 can be set per modality.
- $\mathrm{Ent}_\mu(f)$ — zero iff f is constant on the support of μ (complete insensitivity to perturbations).

Quantification example (Text–Image VQA). Assume a VQA classifier is evaluated on one sample $x = (x_t, x_i)$ with text question x_t and image x_i. Three perturbations $\{z_t^{(k)}\}_{k=1}^{3}$ of the text are drawn while the image is fixed; the resulting cross-entropy values are constant

$$f_t(z^{(1)}) = f_t(z^{(2)}) = f_t(z^{(3)}) = 0.10.$$

Hence $\int f_t \log f_t \, d\mu = 0.10 \log 0.10$, $\int f_t \, d\mu = 0.10$, and $\mathrm{Ent}_\mu(f_t) = 0$, revealing that predictions do not change with text perturbations—text is over-relied upon.

For the image modality, perturbing only x_i yields cross-entropy values

$$f_i(z^{(1)}) = 0.05, \quad f_i(z^{(2)}) = 0.15, \quad f_i(z^{(3)}) = 0.25.$$

A Monte-Carlo estimate gives

$$\mathrm{Ent}_\mu(f_i) \approx \frac{1}{3}\sum_{k=1}^{3} f_i(z^{(k)}) \ln f_i(z^{(k)}) \; - \; \Big(\frac{1}{3}\sum_{k=1}^{3} f_i(z^{(k)})\Big) \ln\Big(\frac{1}{3}\sum_{k=1}^{3} f_i(z^{(k)})\Big) = 0.024 \text{ nats.}$$

Because $\mathrm{Ent}_\mu(f_i) > 0$ while $\mathrm{Ent}_\mu(f_t) = 0$, the metric quantitatively confirms a dataset-induced bias toward text.

2.6 Dataset Bias Mitigation Methods

Mitigation strategies for dataset bias in this work are organized into two main categories: *dataset-centered methods* and *model-centered methods*, each targeting bias from a different perspective in the learning pipeline:

- **Dataset-centered methods** modify, augment, or expand the training data in order to reduce the influence of dataset-specific shortcuts and produce a more balanced and representative input distribution. Typical techniques include bias-aware resampling, constrained distribution learning, targeted synthetic data generation, and large-scale automated sample collection.

- **Model-centered methods** alter the learning process or model architecture to reduce bias sensitivity, even when trained on imperfect data. Approaches in this category include explanation distillation, adversarial unlearning, protected-attribute suppression, and domain confusion.

2.6.1 Data-Centered Methods to Mitigate Dataset Bias

2.6.1.1 Bias-aware Resampling and Re-weighting

A foundational approach to bias mitigation is dataset resampling, which alters the data distribution to reduce the influence of biased samples. Li and Vasconcelos [2019] address "representation bias," where a model learns to exploit spurious correlations specific to a certain feature representation rather than the underlying task . They introduce the REPAIR (REPresentation bIas Removal) algorithm, which formulates bias minimiza

The bias of a dataset $\mathscr{D}$ with respect to a representation ϕ is defined as

$$\mathscr{B}(\mathscr{D}, \phi) = 1 - \frac{\mathscr{R}^*(\mathscr{D}, \phi)}{H(Y)},$$

where

- $\mathscr{D} = \{(x_i, y_i)\}_{i=1}^n$: original dataset of inputs x_i and labels y_i,
- ϕ: fixed feature representation whose induced shortcuts are to be suppressed,
- $\mathscr{R}^*(\mathscr{D}, \phi)$: minimum empirical risk achievable on $\mathscr{D}$ using representation ϕ,
- $H(Y) = -\sum_y p_y \log p_y$: Shannon entropy of the class-label distribution, with p_y the empirical marginal probability of label y.

REPAIR introduces example-level weights $w_i \in (0, 1)$ that down-weight samples easily explained by the biased representation, and a surrogate classifier with parameters θ operating on $\phi(x_i)$. The bias mitigation objective is a minimax optimization:

$$(w^*, \theta^*) = \min_w \max_\theta \mathscr{V}(w, \theta), \qquad \mathscr{V}(w, \theta) = 1 - \frac{\sum_i w_i \log P(y_i \mid x_i; \theta)}{\sum_i w_i \log p'_{y_i}}.$$

In this formulation,

- w_i: learnable weight for example i, controlling its influence in the reweighted dataset,
- θ: parameters of the surrogate (bias estimator) classifier applied on $\phi(x_i)$,

- $P(y_i \mid x_i; \theta)$: predicted posterior probability of label y_i by the classifier with parameters θ,
- $p'_{y_i} = \frac{\sum_{j:y_j=y_i} w_j}{\sum_j w_j}$: reweighted class prior for label y_i,
- $\mathscr{V}(w, \theta)$: objective measuring the relative ease with which the current representation can explain labels under the weighting; minimizing over w suppresses easy (biased) examples, while maximizing over θ tightens the surrogate risk estimate.

The optimization is carried out via alternating stochastic gradient updates of w and θ; the resulting reweighted dataset encourages downstream models to rely less on the biased representation and to learn more generalizable features.

2.6.1.2 Distribution Learning under Fairness Constraints

Moving beyond direct resampling, a more principled modification involves learning a new data distribution that satisfies explicit fairness constraints while remaining close to the empirical distribution. Celis et al. [2020] propose a framework based on the principle of maximum entropy, selecting the distribution that is maximally non-committal subject to predefined fairness constraints.

Let q denote a prior empirical distribution and $\mathscr{C}$ the set of distributions satisfying target constraints such as representation rate or statistical rate. The objective is:

$$p^* = \arg\min_{p \in \mathscr{C}} \mathrm{KL}(p \,\|\, q).$$

- p^*: the learned debiased distribution.
- p: a candidate distribution over the domain.
- $\mathscr{C}$: the constraint set encoding fairness desiderata (e.g., lower bounds on representation rates across protected groups, statistical parity constraints).
- $\mathrm{KL}(p \,\|\, q)$: Kullback–Leibler divergence measuring closeness of p to the empirical prior q.
- q: the original empirical (prior) distribution estimated from data.

By duality, the problem is reduced to solving

$$\inf_{\lambda \in \mathbb{R}^d} h_{\theta,q}(\lambda) := \log \left(\sum_{\alpha \in \Omega} q(\alpha) e^{\langle \alpha - \theta, \lambda \rangle} \right),$$

where

- $\lambda \in \mathbb{R}^d$: dual variables corresponding to the imposed constraints.
- $h_{\theta,q}(\lambda)$: convex dual objective whose minimizer yields the natural parameters of the maximum-entropy distribution.
- Ω: discrete domain over which the distribution is defined (e.g., all possible attribute combinations).
- $q(\alpha)$: prior mass assigned to outcome α under the empirical distribution.
- θ: vector encoding the target marginal expectations or fairness targets (e.g., desired representation rates).
- $\langle \alpha - \theta, \lambda \rangle$: inner product capturing deviation from target constraints weighted by dual variables.

The resulting maximum-entropy distribution admits an exponential-family form and can be efficiently computed; sampling from p^* yields pseudo-data that, when used to train downstream classifiers, produce models with improved fairness metrics while incurring minimal degradation in utility [Celis et al., 2020].

2.6.1.3 Synthetic Data Generation

Synthetic data generation is used either to augment existing datasets or to construct new datasets that are inherently less biased. Two representative strategies are considered.

Targeted Augmentation via Sim2Real Jaipuria et al. [2020] study *noise factor distribution bias* in autonomous driving perception datasets, which arises from insufficient diversity along task-specific environmental variables such as weather, lighting, lane marker types, occlusions, and scene configurations. Their approach is a two-stage targeted augmentation pipeline: first, gaming-engine simulators produce semantically controlled variations corresponding to these noise factors; second, an unsupervised sim-to-real translation model is applied to the simulated images to improve photorealism, yielding synthetic examples that bridge the domain gap.

The augmented training set is constructed by mixing original real data with the translated synthetic data:

$$D = \alpha D_{\text{real}} \cup (1 - \alpha) D_{\text{syn}},$$

- D: the resulting training set used for model fitting.

- $\alpha \in [0, 1]$: the mixing coefficient controlling the proportion of real data; it is tuned to dilute existing selection and capture biases while retaining task relevance.
- D_{real}: the original real-world dataset samples.
- D_{syn}: the simulated data after unsupervised sim-to-real translation (e.g., via generative models) that enhance realism while preserving the controlled diversity of noise factors.
- $\cup$: denotes the combination (with the total training size kept constant), effectively replacing a fraction $1 - \alpha$ of real examples with synthetic ones.

Empirical evaluation on multiple perception tasks demonstrates that this targeted synthetic augmentation significantly improves cross-dataset generalization, while incurring negligible or no degradation on in-domain performance. In particular, replacing portions of biased real data with sim2real-translated examples diffuses strong dataset signatures (as illustrated in Name That Dataset style analyses) and leads to more robust downstream model.

Full Dataset Generation Jiang et al. [2024] propose lbGen, which synthesizes a low-biased annotated dataset by fine-tuning a diffusion model with a composite semantic alignment objective augmented by a quality preservation term. The overall loss is

$$\mathscr{L}_{\text{sem}} = \lambda_g \mathscr{L}_{\text{global}} + \lambda_l \mathscr{L}_{\text{local}} + \lambda_q \mathscr{L}_{\text{quality}},$$

- $\mathscr{L}_{\text{global}}$: encourages the aggregate generated distribution to match a target low-bias semantic distribution, preventing collapse into dominant biased modes;
- $\mathscr{L}_{\text{local}}$: enforces per-instance semantic fidelity by aligning each generated image with its textual class description in embedding space;
- $\mathscr{L}_{\text{quality}}$: maintains perceptual realism, ensuring that the debiasing process does not degrade visual quality.
- λ_g: weight for the global alignment term, governing the strength of matching the aggregate generated distribution to a target low-bias semantic distribution and preventing collapse into dominant biased modes;
- λ_l: weight for the local alignment term, determining the importance of per-instance semantic fidelity by aligning each generated image with its textual class description in embedding space;

- λ_q: weight for the quality preservation term, balancing the debiasing objective against maintaining perceptual realism so that visual quality does not degrade.

The generated dataset is used for backbone pre-training, yielding representations with reduced embedded bias and improved downstream generalization.

2.6.1.4 Automated Real-World Sample Collection

Sevetlidis et al. [2022] present a pipeline that mitigates dataset bias by *expanding* the empirical support with previously unseen Web images while filtering noise at scale. Starting from a class label list $\mathcal{Q} = \{q_1, \ldots, q_K\}$, the system issues parallel queries to multiple search engines, retrieving a raw set $\mathscr{R}$ of $N_{\text{raw}} \approx 8.9 \times 10^5$ candidates. Two sequential rejection modules then refine the data:

- **Irrelevant–content filter**: three pre-trained CNNs (InceptionV3, MobileNet, ResNet-50) act as binary voters; an image x is kept only if the majority predicts *relevant*. Approximately $N_{\text{irr}} = 1.12 \times 10^5$ images (12.6%) are discarded.
- **Duplicate–removal module**: locality-sensitive hashes (average, perceptual, difference) combined with a BK-tree and Hamming distance detect near-duplicates. This step eliminates $N_{\text{dup}} = 1.29 \times 10^5$ images (14.6%).

The remaining $N_{\text{final}} = 6.45 \times 10^5$ images ($\approx$ 5 190 per class) constitute an expanded dataset D_{auto} that (i) reduces intra-class bias, (ii) lowers model performance variance across benchmark splits, and (iii) is produced in $\sim$4 days thanks to containerised, parallel execution. Cross-evaluation with Food-101 variants shows that models trained on D_{auto} generalise more uniformly, indicating that broad real-world sampling can effectively dilute dataset-specific signatures.

2.6.2 Model-Centered Methods to Mitigate Dataset Bias

2.6.2.1 Explanation Distillation for Mitigating Shortcut Learning

Explanation distillation constrains a model to adopt the *rationale* of an (assumed) less-biased teacher by aligning explanation maps, thereby reducing reliance on spurious shortcuts in the training data Bassi et al. [2024]. Let $g_T(x, k)$ and $g_S(x, k)$ be the teacher's and student's explanation maps for class k. The core alignment loss is

$$d(g_T, g_S) = \frac{\|g_T - g_S\|_1}{\sqrt{\|g_T\|_1 \, \|g_S\|_1}}, \tag{2.6.1}$$

- $g_T(x, k)$: explanation map from the teacher for input x and class k,
- $g_S(x, k)$: corresponding explanation map from the student,
- $\|\cdot\|_1$: element-wise L_1 norm.

To capture multi-scale consistency and suppress scale-specific biased cues, the loss is averaged over pooled resolutions:

$$L_{\text{expl}} = \frac{1}{M} \sum_{m=0}^{M-1} d(\text{AvgPool}(g_T, 2^m),\ \text{AvgPool}(g_S, 2^m)) \,, \tag{2.6.2}$$

- M: number of scales,
- $\text{AvgPool}(\cdot, 2^m)$: spatial average pooling by factor 2^m.

The overall objective combines task supervision with explanation alignment, with the explanation loss applied to internal layers and standard output-level supervision confined to the final layer (DL-DL scheme) to avoid competing gradient signals:

$$L = L_{\text{task}} + \beta\, L_{\text{expl}}, \tag{2.6.3}$$

- L_{task}: primary prediction loss (e.g., cross-entropy or soft-label distillation at output),
- β: trade-off weight.

This mechanism mitigates dataset bias by forcing the student to base predictions on the teacher's explanation structure rather than on superficial correlations present in biased training examples; multi-scale alignment ensures that both fine and coarse evidential patterns reflect the intended (less-biased) reasoning. The DL-DL partitioning further prevents the student from reverting to shortcut learning via conflicting objectives. The approach presumes access to a teacher with relatively unbiased explanations and stability of the explanation operator; if the teacher's explanations are themselves biased or noisy, the benefit is limited.

2.6.2.2 Adversarial Unlearning of Known Bias via Mutual Information Minimization

When a dataset contains a known and annotatable bias b that correlates with the label only in training (not in the target distribution), models tend to exploit this spurious

shortcut. Adversarial training (Ganin et al. [2016])suppresses the encoding of such bias in the internal representation by treating the bias predictor as an adversary(Kim et al. [2019]): the feature extractor is trained to perform the main task while simultaneously making bias recovery difficult. The training objective is:

$$\min_{\theta,\phi} \mathscr{L}_{\text{task}}\big(y, g_\phi(f_\theta(x))\big) - \lambda\, \mathscr{L}_{\text{bias}}\big(b, h_\psi(f_\theta(x))\big), \quad \min_{\psi} \mathscr{L}_{\text{bias}}\big(b, h_\psi(f_\theta(x))\big), \tag{2.6.4}$$

- $f_\theta(x)$: feature representation produced by the extractor with parameters θ,
- $g_\phi(\cdot)$: main task classifier with parameters ϕ,
- $h_\psi(\cdot)$: bias classifier (adversary) with parameters ψ,
- y: true label for the primary task,
- b: known bias attribute (e.g., color, demographic group),
- $\mathscr{L}_{\text{task}}$: loss for the main task (typically cross-entropy between $g_\phi(f_\theta(x))$ and y),
- $\mathscr{L}_{\text{bias}}$: loss for predicting b from $f_\theta(x)$,
- $\lambda > 0$: weight controlling the strength of bias removal.

In practice, the negative term for $\mathscr{L}_{\text{bias}}$ in the feature extractor's update is implemented via a gradient reversal mechanism, so that h_ψ is trained to minimize $\mathscr{L}_{\text{bias}}$ while f_θ is trained to *maximize* it (i.e., to fool the bias predictor), effectively minimizing the mutual information $I(b; f_\theta(X))$. This yields representations informative for y but uninformative for b, diminishing the model's ability to rely on the spurious correlation. Such disentanglement improves generalization when the bias-label association does not hold at test time. What is note worthy is that the approach requires prior knowledge of the bias attribute and its annotation. Training stability depends on balancing the adversarial dynamics; improper weighting (λ) or overly expressive bias predictors can lead to collapse or excessive degradation of task-relevant features when the bias and target are tightly entangled.

2.6.2.3 Adversarial Fairness via Protected-Attribute Suppression

Adversarial fairness methods mitigate dataset bias by explicitly removing information about a protected attribute from the model's outputs, enforcing independence between the predictor and the sensitive variable under formal fairness criteria such as demographic parity or equality of odds Zhang et al. [2018]. A primary predictor $f(X)$ produces $\hat{Y}$, while an adversary attempts to recover the protected attribute Z from $\hat{Y}$ (and optionally

Y for conditional criteria). The predictor's update is modified to both avoid inadvertently helping the adversary and to actively suppress Z-related signals:

$$\nabla_W L_P \leftarrow \nabla_W L_P - \text{proj}_{\nabla_W L_A} \nabla_W L_P - \alpha \nabla_W L_A, \tag{2.6.5}$$

- L_P: primary task loss (e.g., classification loss for $\hat{Y}$),
- L_A: adversary loss for predicting protected attribute Z,
- ∇_W: gradient with respect to predictor parameters W,
- $\text{proj}_{\nabla_W L_A} \nabla_W L_P$: projection of $\nabla_W L_P$ onto $\nabla_W L_A$, removing components that would assist the adversary,
- $\alpha > 0$: weight controlling the strength of active suppression of Z-information.

By adjusting the adversary's inputs (e.g., including Y when enforcing equality of odds), different fairness constraints are instantiated. The combined effect of the projection and explicit adversarial term ensures that the predictor's outputs retain task-relevant information while becoming statistically less informative about Z, thereby reducing bias induced by correlations between Z and the label in the training data (Zhang et al. [2018]). Also, the method assumes access to the protected attribute Z during training. Training stability depends on balancing the adversary and predictor; poor capacity choices or hyperparameter settings (e.g., α) can lead to under- or over-suppression, degrading task performance or failing to remove bias. Additionally, different fairness definitions may conflict, requiring explicit trade-offs.

2.6.2.4 Domain Confusion via Maximum Mean Discrepancy

Domain confusion mitigates dataset bias arising from distributional shift by learning representations that are simultaneously discriminative for the main task and invariant across source and target domains (Tzeng et al. [2014]). The method measures and minimizes the discrepancy between domain feature distributions using Maximum Mean Discrepancy (MMD). Given source samples X_S and target samples X_T, with feature mapping $\phi(\cdot)$, the empirical MMD is

$$\text{MMD}(X_S, X_T) = \left\| \frac{1}{|X_S|} \sum_{x_s \in X_S} \phi(x_s) - \frac{1}{|X_T|} \sum_{x_t \in X_T} \phi(x_t) \right\|, \tag{2.6.6}$$

- X_S, X_T: source and target domain samples,
- $\phi(x)$: learned representation of input x,

- $|\cdot|$: cardinality of the sample set,
- $\|\cdot\|$: norm in feature space (often RKHS-induced).

The joint objective combines the primary classification loss L_C (on labeled source data) with the squared MMD to enforce domain invariance:

$$L = L_C + \lambda \, \mathrm{MMD}^2(X_S, X_T), \tag{2.6.7}$$

- L_C: supervised task loss (e.g., cross-entropy on source labels),
- $\lambda > 0$: trade-off hyperparameter balancing discrimination and invariance.

An adaptation layer is inserted into the deep network, and empirical estimates of MMD guide the selection of its placement and dimensionality so that the representation optimally aligns domains without sacrificing semantic separability. By minimizing the distributional gap between domains in representation space, the model is discouraged from relying on domain-specific artifacts—i.e., dataset bias—when making predictions, improving transfer to shifted target data (Tzeng et al. [2014]).This approach presumes access to (unlabeled or sparsely labeled) target-domain samples during training to compute MMD. Its effectiveness depends on the choice of kernel or implicit feature embedding used in estimating MMD; inappropriate choices can yield weak alignment. Hyperparameter tuning (e.g., λ, adaptation layer depth/width) remains necessary, and extreme domain shifts with complex structural differences may require more expressive alignment mechanisms beyond first-order moment matching.

2.7 Research Gaps and Challenges

Despite substantial progress in understanding dataset bias, several critical research gaps persist in the literature, presenting significant challenges for future research.

2.7.1 Fragmented Evaluation Protocols

Most empirical investigations adopt bespoke train–test splits, bias metrics, and architectural backbones, which significantly hinders cross-paper comparability. “Name That Dataset” accuracy, cross-dataset generalization, and entropy-based indices target different bias facets, yet seldom co-occur in a single study [Tommasi et al., 2017, Torralba and Efros, 2011, Zeng et al., 2024]. The absence of a unified benchmark suite obstructs reproducibility and complicates meta-analysis across mitigation strategies.

2.7.2 Benefit-Harm Paradox (Dual Nature)

A fundamental tension exists between two empirically observed but theoretically ununified phenomena: (1) dataset bias encodes transferable semantic patterns beneficial for downstream tasks [Liu and He, 2025, Zeng et al., 2024], yet (2) simultaneously causes systematic O.O.D. generalization failures [Geirhos et al., 2020, Liu et al., 2021]. Current quantification methods lack granularity to disentangle *which* kind of biased factors contribute to beneficial transfer versus harmful shortcut learning. No existing framework predicts when biased features transits between being advantageous and detrimental [Fabbrizzi et al., 2022].

2.7.3 Incomplete Bias Mechanistic Understanding

Critical gaps persist in explaining how bias propagates through learning systems:

- **Architectural mediation**: The V-shaped bias-amplification curve [Hall et al., 2022] and differential attention mechanisms in transformers [Talebpour et al., 2025] lack comprehensive theoretical links between architectural properties (capacity, attention geometry) and bias propagation dynamics.
- **Multimodal entanglement**: Vision-language models exhibit complex cross-modal biases where textual descriptions reinforce visual stereotypes [Zeng et al., 2024], yet no mitigation frameworks address this coupled distortion.
- **Temporal dynamics**: Current approaches [Bassi et al., 2024, Jiang et al., 2024, Kim et al., 2019, Zhang et al., 2018] treat bias as static, ignoring its evolution in continuously updated data streams—a critical limitation for real-world streaming applications.

2.7.4 Synthetic Data Limitations

While synthetic augmentation shows promise for debiasing [Jaipuria et al., 2020, Jiang et al., 2024], two fundamental constraints undermine its efficacy:

- **Inherent bias inheritance**: Generative models trained on biased datasets inevitably encode and amplify existing distortions [Zeng et al., 2024], as truly unbiased dataset remain nonexistent [Fabbrizzi et al., 2022, Mehrabi et al., 2021].
- **Evaluation myopia**: Quality assessments [Jaipuria et al., 2020] focus predominantly on in-distribution accuracy, obscuring performance on O.O.D. distribution.

2.7.5 Partial Mitigation and Objective Conflicts

Current debiasing approaches exhibit three critical shortcomings:

- **Uni-dimensional focus**: Methods typically target single bias attributes (e.g., color or gender) [Merler et al., 2019, Panda et al., 2018] while neglecting interacting biases, often amplifying untreated distortions during mitigation.
- **Conflicting objectives**: Tensions between fairness, robustness, and accuracy remain undercharacterized, with immature multi-objective optimization frameworks [Gat et al., 2021].
- **Overcorrection risks**: Aggressive debiasing [Kim et al., 2019] may discard transferable semantic signals alongside harmful shortcuts, degrading task-relevant feature quality.

2.8 Summary of Challenges

These gaps coalesce into four primary research challenges:

1. *Metric Unification*: Establishing unified benchmarks for bias quantification and mitigation evaluation.
2. *Benefit-Harm Paradox*: Developing theoretical frameworks to predict when bias features aid versus harm generalization.
3. *Generative Integrity*: Ensuring synthetic data pipelines avoid harmful bias inheritance while maintaining semantic fidelity.
4. *Holistic Debiasing*: Creating multi-dimensional debiasing frameworks that balance competing objectives.

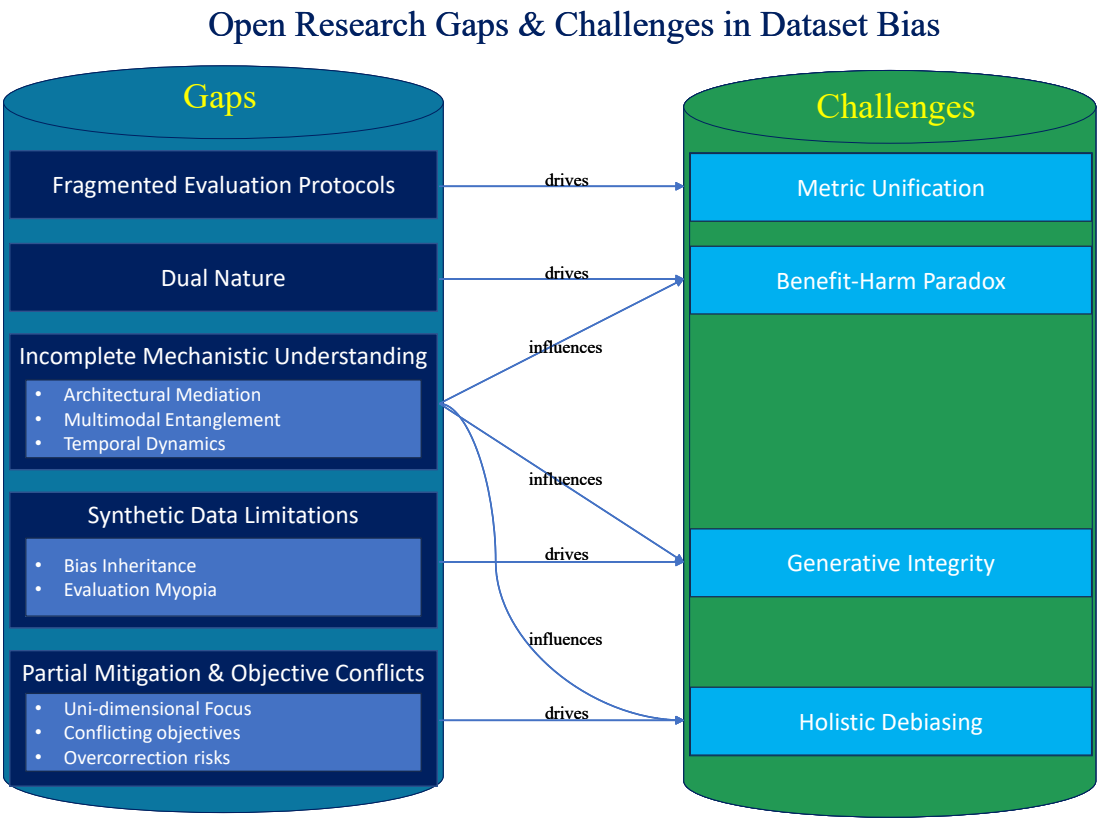


FIGURE 2.3: Open Research Gaps & Challenges in Dataset Bias

2.9 Situating the Present Study within the Literature

The foregoing survey has traced a trajectory from the first demonstrations of dataset–specific signatures in shallow models [Torralba and Efros, 2011] through early debiasing attempts in discriminative frameworks [Fang et al., 2013, Khosla et al., 2012], to recent evidence that modern architectures amplify rather than eliminate bias [Liu and He, 2025, Tommasi et al., 2017, Zeng et al., 2024]. Against this background, the present dissertation contributes four complementary advances that address persisting blind spots in the field.

The present work

1. offers the reproduction of the classical dataset-bias experiment on modern DNNs;
2. quantifies the relative dominance and conditional salience of non-semantic information versus semantic information contained in dataset bias;
3. reveals architecture-specific bias propagation pathways via cross-layer linear probing [Alain and Bengio, 2016]; and

4. introduces two improved loss functions that improve semantic classification performance while lowering dataset bias strength.

These advances situate the dissertation at the nexus of empirical quantification, mechanistic understanding, and practical mitigation, thereby extending a decade of research on dataset bias into a unified and actionable framework.

Chapter 3

Methodology

3.1 Sectional Introduction

The Methodology chapter outlines the structured approach adopted to investigate and mitigate dataset bias in deep neural networks. Centered on the core research question of analyzing and mitigating dataset bias, this chapter briefly introduces the overall research framework comprising three principal stages: (1) reproducing the "Name that Dataset" baseline with modern deep neural networks, (2) conducting extended investigations designed to further dissect the nature and localization of dataset bias, and (3) developing and evaluating bias mitigation techniques.

To support rigorous experimentation, the construction of carefully designed merged datasets (03–06) from TinyImageNet and CIFAR-100 is first elaborated, encompassing semantic label alignment, controlled resolution standardization, and new image split methods.

Subsequently, the ResNet and Vision Transformer architectures serving as the analytical substrates are introduced.

Finally, three specialized training protocols (A, B, C) formalizing procedures for ResNet semantic classifier training, Vision Transformer semantic classifier training, and linear probe training for data bias analysis are presented.

Together, these elements establish a coherent, step by step pathway from dataset bias investigation to mitigation, laying the empirical foundation for the analyses that follow.

3.2 Overall Research Framework

The research is structured into three principal stages:

- Baseline Reproduction: Name the Dataset with Modern DNN
- Extended Experiments to Investigate Dataset Bias Further
- Dataset Bias Mitigation and Evaluation

3.2.1 Baseline Reproduction: Name the Dataset with Modern DNN

The "Name that dataset" experiment of Torralba & Efros [1] is re-implemented using modern deep neural network architectures. Unlike the original study, which employed SVM classifiers on hand-crafted feature descriptors, this reproduction utilizes architectures such as ResNet to assess the capacity of modern DNNs to identify the source dataset.

To this end, the baseline reproduction proceeds in two consecutive steps (Fig.3.1):

- **Name the Label**
  In this preliminary stage, images from TinyImageNet and CIFAR-100 that share a common set of *semantically overlapping object categories* are merged into a single pool and labeled by class. Here, "semantically overlapping" means that the categories represent the same underlying concept, even if their label names differ (e.g., `grass` in one dataset vs. `meadow` in another, or `ladybug` vs. `beetle`). Categories without such semantic overlap are excluded from consideration. A model is then trained to predict the probability distribution over these shared labels.
- **Name the Dataset**
  In this stage, the model obtained from the "Name the Label" task has all its layers frozen except for the final fully connected layer. The original fully connected layer is removed and replaced with a new one whose output size matches the number of source datasets (two: TinyImageNet and CIFAR-100). **This setup—where we freeze the entire backbone and train only a single new linear classifier—is exactly what's known as a linear probe [Alain and Bengio, 2016].** Training is conducted exclusively on this new fully connected layer so that the network learns to predict the probability of each image originating from each dataset, thus revealing how much dataset-specific information is contained in the learned representations.

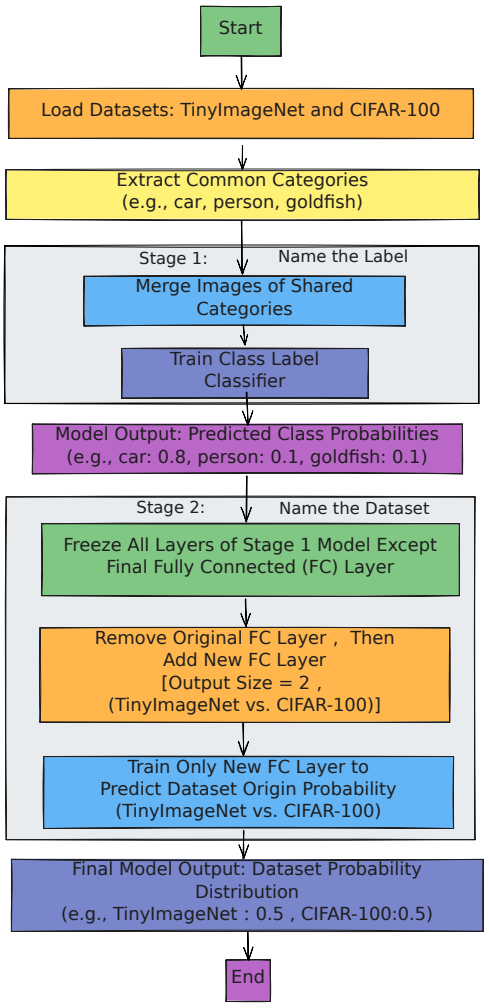


FIGURE 3.1: Name the Label and Name the Dataset Workflow

3.2.2 Extended Experiments to Investigate Dataset Bias Further

Building upon the DNN-based reproduction of the "Name the Dataset" experiment, three additional studies were designed to disentangle the contributions of semantic versus non-semantic information, to localize dataset bias within network hierarchies, and to evaluate dataset bias robustness when the merged dataset scale is enlarged.

1. **Non-Semantic Test**

   This experiment evaluates whether the classifier relies primarily on useful, high-level semantic information (e.g., object shapes and structures) or on spurious, non-semantic information (e.g., compression artifacts, sensor noise) to discriminate between TinyImageNet and CIFAR-100. Comparative performance on these manipulated inputs reveals the relative importance of each information type for dataset identification.

2. **Linear Probes [Alain and Bengio, 2016] at Different Layers**
   A series of linear probes are attached at multiple depths of each pretrained backbone—after each residual stage in ResNet and after selected transformer blocks in ViT. Each selected position, where a linear probe is inserted, is trained with two separate probes: one probe is trained to predict the object class labels (Name the Label) and the other is trained to predict the dataset source (Name the Dataset). By charting accuracy as a function of network depth, this approach pinpoints the layer(s) at which dataset biases become most salient, and it contrasts the proportion of semantic versus non-semantic information contained in dataset bias across the network.

3. **Two-Dataset Merge**
   Unlike the shared-category baseline, this variant merges the full TinyImageNet and CIFAR-100 datasets without requiring any label overlap. The combined pool is then used for both the Name the Label and Name the Dataset tasks. This setup amplifies dataset bias by adding more images for the model to learn, making it easier for the model to distinguish datasets.

3.2.3 Data Bias Mitigation Methods

Inspired by Domain-Adversarial Neural Networks (DANN) [Ganin et al., 2016], two modified training methods are proposed to mitigate dataset bias. When applied to the Name the Label (semantic classification) task, the methods achieved approximately a $1 \sim 2$ percentage-point absolute increase in Name the Label test accuracy relative to the unmitigated baseline for a ResNet-18 trained on a dataset containing 170,000 images.

The primary idea of the modified training methods is adding a new **bias strength**[1] term in the loss funtion to be minimize, and the new loss function can be expressed as:

$$L_{\text{total}}(\theta) = \underbrace{L(\theta)}_{\text{Task Loss}} + \lambda \cdot \underbrace{\mathscr{B}(\theta)}_{\text{Bias Strength}}$$

where $\mathscr{B}(\theta)$ approximates the dataset bias strength, whereas λ is a parameter controlling the relative importance of mitigating the dataset bias compared with minimizing the task loss.

[1] Here, the "bias strength" is approximated by the *Name-the-Dataset* Accuracy.

3.3 Datasets Construction

Datasets in computer vision are composed of images grouped into discrete categories, commonly referred to as *classes* (or *labels*). Each *class* (or *label*) represents a distinct object category or semantic concept, and every image in the dataset is annotated with exactly one of these classes (or *labels*). Formally, if the set of all classes (or *labels*) are denoted by

$$\mathscr{C} = \{c_1, c_2, \ldots, c_K\},$$

then each sample in the dataset can be written as a pair $(\mathbf{x}_i, y_i)$, where $\mathbf{x}_i$ is the image and $y_i \in \mathscr{C}$ is its class label. In this work, "class" and "label" are used interchangeably to refer to these categorical annotations.

Besides, all the dataset construction details are summarized in Appendix. A

3.3.1 Dataset Introduction and Selection Rationale

The research utilizes two widely-recognized image datasets: TinyImageNet and CIFAR-100. Their selection is based on their distinct characteristics, which facilitate efficient experimentation and enable targeted investigations into dataset bias.

- **TinyImageNet** [Li et al., 2015] is a subset of the larger ImageNet dataset, designed to be more manageable for research purposes. It comprises 200 distinct object lables (classes), with 500 training images per class and 50 validation images per class. **Although it holds a test set, but in the test set there are only images without annotations**. All images are downsampled to a resolution of 64x64 pixels. This smaller scale, while still offering a diverse range of semantic categories, allows for faster model training and iteration compared to the full ImageNet dataset.
- **CIFAR-100** [Krizhevsky and Hinton, 2009] is another established image dataset, featuring 100 object classes. Similar to TinyImageNet, it includes 500 training images per class and 100 testing images per class, and **it do not have a validation set**. The images in CIFAR-100 are of a lower resolution, specifically 32×32 pixels. It also provides *coarse labels*, which group the 100 fine-grained classes into 20 superclasses, though the primary focus of this research is on the individual (fine-grained) classes.

The choice of TinyImageNet and CIFAR-100 for this research is strategic due to several key factors:

- **Manageable Scale and Experimental Speed**: Both datasets are relatively small in size compared to larger alternatives. This enables rapid experimentation and reduces the computational resources and time required for training and evaluating deep neural networks, making the iterative research process more efficient.
- **Semantic Overlap Lables**: Crucially, there is a degree of semantic overlap between the object categories present in TinyImageNet and CIFAR-100. This overlap is essential for the "Name the Label" task in our baseline reproduction, where images sharing common classes are merged. This allows us to investigate how models differentiate between datasets when high-level semantic information is similar.

This Table 3.1 highlights their scale, resolution, and label-structure differences, guiding our choice for efficient, semantically controlled experiments.

Feature	**TinyImageNet**	**CIFAR-100**
Number of classes	200	100
Training images per class	500	500
Validation / test images per class	50 / 50 (test set no annotation)	0 / 100 (no validation set)
Image resolution	64×64	32×32
Used label structure	synset	100 fine-grained labels
Label overlap	27 labels semantically overlapping with CIFAR-100	27 labels semantically overlapping with TinyImageNet
Common uses	Small-scale ImageNet research, fine-tuning	Rapid prototyping of classification models, benchmarking

TABLE 3.1: Datasets Comparison of TinyImageNet & CIFAR-100

3.3.2 Construction Workflow for Merged Datasets(Datasets 03 & 04)

The construction (Fig.3.2) of the "Merged New Dataset of Semantically Overlapping Labels" (Datasets 03 and 04) follows a rigorous, multi-stage pipeline designed to create a unified dataset from CIFAR-100 and TinyImageNet. This methodology carefully addresses semantic alignment, resolution standardization, and robust partitioning to facilitate comprehensive studies on dataset bias and cross-dataset generalization.

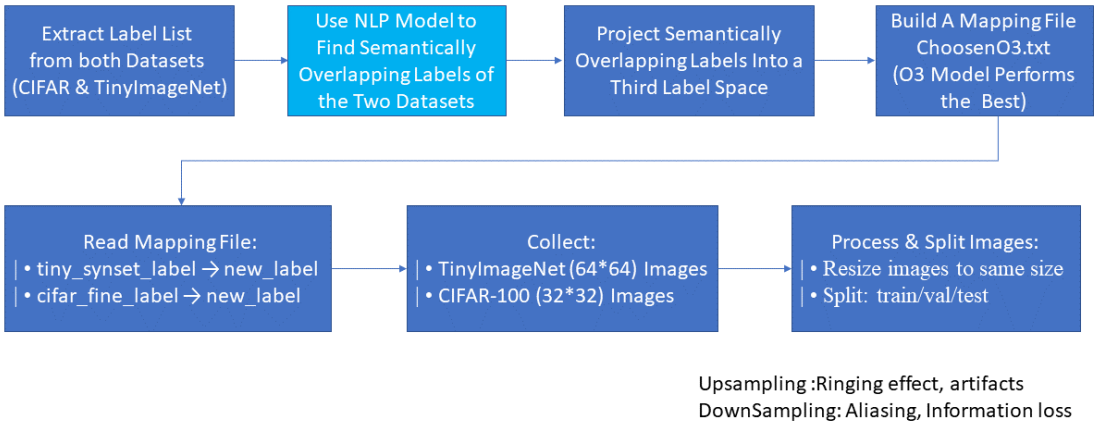


FIGURE 3.2: Overall Workflow for Dataset 03 & 04 Construction

3.3.2.1 Label Extraction & Semantic Matching

The initial phase involves establishing conceptual correspondence between the class (label) vocabularies of the source datasets, which can be implemented by two steps:

1. **Label Acquisition**: Labels were first extracted from CIFAR-100 (specifically its "fine" labels) and TinyImageNet (using its WordNet synset names).

2. **NLP-Driven Semantic Proximity Quantification**: A Natural Language Processing (NLP) model was subsequently employed to quantify the semantic proximity between every possible pair of labels from the two datasets. This approach generated candidate mappings based on conceptual relatedness rather than strict lexical equivalence. As an example, CIFAR-100's "aquarium fish" and TinyImageNet's "goldfish" are jointly merged and projected onto a single unified label within the projected label space, acknowledging their close semantic proximity despite lexical variation. A representative subset of these semantic alignments is provided in Figure 3.3.

3.3.2.2 Label Space Projection

Following the comprehensive NLP semantic matching process, the identified semantically overlapping labels were mapped into a unified new label space. This critical transformation involved the creation of a consolidated taxonomy designed to bridge the distinct label vocabularies of the TinyImageNet and CIFAR-100 datasets.

	A	B	C	D	E	F
1	new_label_id	new_label	tiny_synset_id	tiny_synset_name	cifar_fine	cifar_fine_name
2	0	goldfish	n01443537	goldfish, Carassius auratus	1	aquarium_fish
3	1	brown bear	n02132136	brown bear, bruin, Ursus arctos	3	bear
4	2	bee	n02206856	bee	6	bee
5	3	lady beetle	n02165456	ladybug, ladybeetle, lady beetle, ladybird, ladybird beetle	7	beetle
6	4	pop bottle	n03983396	pop bottle, soda bottle	9	bottle
7	5	suspension bridge	n04366367	suspension bridge	12	bridge
8	6	school bus	n04146614	school bus	13	bus
9	7	monarch butterfly	n02279792	monarch, monarch butterfly, milkweed butterfly, Danaus plexippus	14	butterfly
10	8	Arabian camel	n02437312	Arabian camel, dromedary, Camelus dromedarius	15	camel
11	9	ox	n02403003	ox	19	cattle
12	10	rocking chair	n04099969	rocking chair, rocker	20	chair
13	11	chimpanzee	n02481823	chimpanzee, chimp, Pan troglodytes	21	chimpanzee
14	12	cockroach	n02233338	cockroach, roach	24	cockroach
15	13	African elephant	n02504458	African elephant, Loxodonta africana	31	elephant
16	14	computer keyboard	n03085013	computer keyboard, keypad	39	keyboard
17	15	lawn mower	n03649909	lawn mower, mower	41	lawn_mower
18	16	lion	n02129165	lion, king of beasts, Panthera leo	43	lion
19	17	American lobster	n01983481	American lobster, Northern lobster, Maine lobster, Homarus americanus	45	lobster
20	18	mushroom	n07734744	mushroom	51	mushroom
21	19	orange	n07747607	orange	53	orange
22	20	plate	n07579787	plate	61	plate
23	21	snail	n01944390	snail	77	snail
24	22	dining table	n03201208	dining table, board	84	table
25	23	pay-phone	n03902125	pay-phone, pay-station	86	telephone
26	24	tractor	n04465501	tractor	89	tractor
27	25	bullet train	n02917067	bullet train, bullet	90	train
28	26	bell pepper	n07720875	bell pepper	83	sweet_pepper

FIGURE 3.3: Selected 27 Semantically Overlapping Labels from Tiny ImageNet & CIFAR-100

Crucially, this mapping intentionally incorporated classes exhibiting partial semantic overlap—meaning these labels are conceptually related rather than strictly identical (e.g., 'ox' and 'cattle'). This deliberate methodological choice facilitates a more nuanced analysis of model performance under varying degrees of semantic relatedness.

A representative subset of this projection is presented in the Table 3.2 below, exemplifying instances where original labels with partial semantic overlap have been effectively merged into the unified label space. This precise alignment constitutes a pivotal aspect of our research design.

New label id	New label name	Tiny synset	Tiny synset name	CIFAR id	CIFAR fine name
5	suspension bridge	n04366367	suspension bridge	12	bridge
6	school bus	n04146614	school bus	13	bus
9	ox	n02403003	ox	19	cattle
23	pay-phone	n03902125	pay-phone, pay-station	86	telephone

Note: Mappings only require semantic relatedness rather than exact equivalence (e.g., "ox" → "cattle").

TABLE 3.2: Partial semantic overlap label pairs

3.3.2.3 Data Collection & Preprocessing

This stage involved the acquisition of raw image data and its subsequent transformation into a standardized format suitable for deep learning model input.

1. **Image Aggregation**: All semantically overlapping samples from CIFAR-100 (original resolution: 32×32 pixels) and TinyImageNet (original resolution: 64×64 pixels), as identified by our mapping file, were systematically aggregated.

2. **Resolution Standardization**: To ensure compatibility with models(e.g., ResNet, ViT) and to create a homogeneous dataset, all collected images underwent resolution standardization. This process also marks the only distinction between Dataset03 and Dataset04, as detailed below (Tab.3.3):

Preprocessing	**Dataset03**	**Dataset04**
TinyImageNet (64×64)	(64×64)→(256×256) (Bicubic)	(64×64)→(32×32)→(256×256) (Bicubic)
CIFAR-100 (32×32)	(32×32)→(256×256) (Bicubic)	(32×32)→(256×256) (Bicubic)

TABLE 3.3: Preprocessing steps for Dataset03 and Dataset04

   - For *Dataset03*, all images were directly resized to a uniform target size of 256×256 pixels using **bicubic interpolation** [Keys, 2003]. This method was selected for its balance in preserving image quality during upsampling.
   - For *Dataset04*, a more complex preprocessing pipeline was implemented specifically for TinyImageNet images to investigate the impact of an initial "resolution parity" step:
     - TinyImageNet (64×64) images were first downsampled to 32×32 pixels (matching CIFAR-100's native resolution) using **bicubic interpolation**.
     - Following this, both these downsampled TinyImageNet images and the original CIFAR-100 (32×32) images were then upsampled to 256×256 pixels using **bicubic interpolation**.

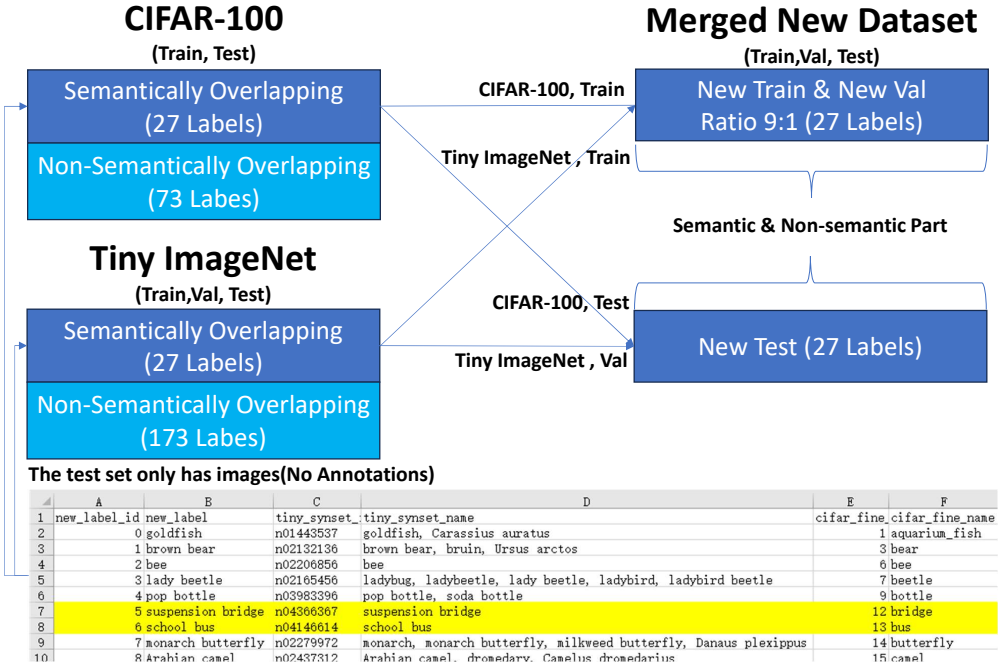
3. **Inherent Interpolation Biases**: It is critical to acknowledge that all interpolation methods, whether upsampling or downsampling, inherently introduce distinct types of artifacts and biases into the image data. Upsampling, for instance, risks introducing visual distortions such as ringing effects (false edges or halos near sharp transitions) and other general interpolation artifacts. Conversely, downsampling can lead to aliasing (jagged or stair-stepped lines) and irretrievable information loss. Despite these known biases, resizing is a necessary step for ensuring model compatibility and creating a unified input format.

4. **Storage**: To strike a balance between image quality and manageable storage requirements, every resized image was saved as a **JPEG with a quality setting of 95**. This specific setting was chosen to minimize compression artifacts, which, if left uncontrolled, are capable of inducing new dataset biases, while still maintaining a practical overall dataset size.

3.3.2.4 Dataset Partitioning

The final step (Fig.3.4) involves organizing the processed images into standard training, validation, and testing sets. This partitioning strategy ensures a balanced representation of all unified labels across subsets, critical for fair model evaluation and generalization studies.

- **Test Set Split**: Specifically, the new test set was constructed from TinyImageNet's original validation split and CIFAR-100's original test split, corresponding to the semantically overlapping classes.
- **Train & Validation Set Split**: The combined training images from TinyImageNet and CIFAR-100 (for the overlapping classes) were then randomly partitioned into 90% for training and 10% for validation, with a fixed random seed (= 42) to ensure reproducibility.



	A	B	C	D	E	F
1	new_label_id	new_label	tiny_synset_	tiny_synset_name	cifar_fine_	cifar_fine_name
2	0	goldfish	n01443537	goldfish, Carassius auratus	1	aquarium_fish
3	1	brown bear	n02132136	brown bear, bruin, Ursus arctos	3	bear
4	2	bee	n02206856	bee	6	bee
5	3	lady beetle	n02165456	ladybug, ladybeetle, lady beetle, ladybird, ladybird beetle	7	beetle
6	4	pop bottle	n03983396	pop bottle, soda bottle	9	bottle
7	5	suspension bridge	n04366367	suspension bridge	12	bridge
8	6	school bus	n04146614	school bus	13	bus
9	7	monarch butterfly	n02279972	monarch, monarch butterfly, milkweed butterfly, Danaus plexippus	14	butterfly
10	8	Arabian camel	n02437312	Arabian camel, dromedary, Camelus dromedarius	15	camel

FIGURE 3.4: Train/Validation/Test Splits for the Merged Datasets (03 & 04)

This comprehensive pipeline yields a coherent, multi-source dataset specifically designed to enable cross-dataset biases studies. And the details of number of images per-split are listed in Table 3.4.

Dataset / Split	**Overlapping Classes**	**Images per class**	**Source subset**	**Image count**
CIFAR-100 (original train)	27	500	train	13,500
TinyImageNet (original train)	27	500	train	13,500
Train total (90%)	—	—	merged 27-class train	24,300
Validation total (10%)	—	—	merged 27-class train	2,700
CIFAR-100 (original test)	27	100	test	2,700
TinyImageNet (original val → test)	27	50	val (treated as test)	1,400
Test total	—	—	merged 27-class test/ val	4,050
Overall total	—	—	all splits	31,050

Note: all image counts assume exactly 27 overlapping classes.

TABLE 3.4: Number of Images of Train/Validation/Test Splits for the Merged Datasets (03 & 04)

3.3.2.5 Construction of Non-Semantic Test Set in Datasets 03 & 04

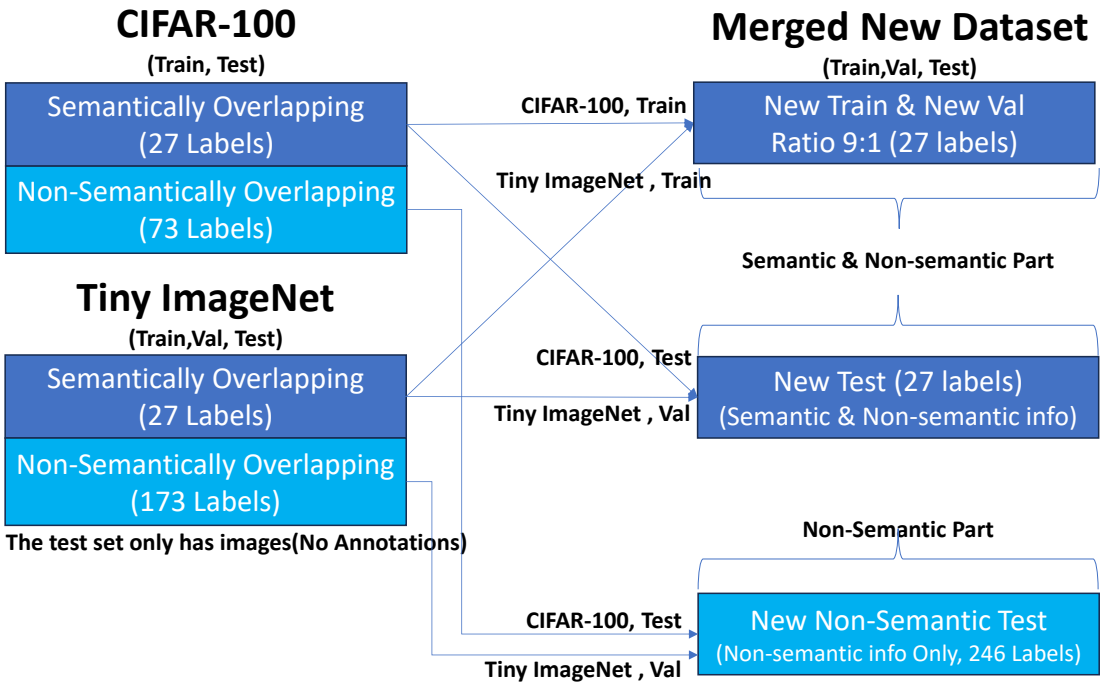


FIGURE 3.5: Train/Validation/Test/Non-Semantic Test Splits for the Merged Datasets (03 & 04)

Although Datasets 03 and 04 already include a balanced train/val/test split for the 27 semantically-overlapping labels, an additional test-only benchmark (Fig.3.5) was created to diagnose how strongly a model trained on those 27 labels relies on non-semantic information to name the dataset.

Specifically, for the two chosen datasets CIFAR-100 and TinyImageNet, there are 27 semantically-overlapping labels (Fig.3.3) and the rest 246 labels are non-semantically overlapping labels (Tab.3.5, Fig.3.5). **The corresponding images of those non-semantically overlapping labels in TinyImageNet's validation set and CIFAR-100's test set are gathered together to form a new non-semantic test set.**

	CIFAR-100	TinyImageNet	Total
Original fine / synset labels	100	200	300
Semantically-overlapping labels	27	27	27
Non-overlapping labels (used here)	**73**	**173**	**246**

TABLE 3.5: Illustration of Non-overlapping Labels Used in Non-Semantic Test Set

Because the preprocessing pipelines differ slightly between Dataset 03 and Dataset 04, **two parallel versions of the non-semantic test set were produced—one for**

each dataset variant—so that every model is evaluated on data treated *exactly* like its training images.

1. Number of Source Images

The Table 3.6 below shows the number of source images from each dataset.

Dataset split	Used Labels	Images per class	Source subset	Image count
CIFAR-100	73	100	test	7 300
TinyImageNet	173	50	val	8 650
Total	246	–	–	15 950

TABLE 3.6: Number of Images of the Non-semantic Test Set (03-NS & 04-NS)

2. Pre-processing & Saving Format (identical to each parent dataset 03/04)

This (Tab.3.7) guarantees that every non-semantic test image has undergone the *same* chain of interpolations and JPEG compression as its corresponding Dataset 03 or Dataset 04 training images, preserving any biasing artifacts (e.g., ringing, aliasing).

Stage	**03 Non-semantic**	**04 Non-semantic**
TinyImageNet images	$64 \times 64 \rightarrow 256 \times 256$ (bicubic upsample)	$64 \times 64 \rightarrow 32 \times 32$ (bicubic downsample) $\rightarrow 256 \times 256$ (bicubic upsample)
CIFAR-100 images	$32 \times 32 \rightarrow 256 \times 256$ (bicubic upsample)	$32 \times 32 \rightarrow 256 \times 256$ (bicubic upsample)
File format	JPEG, quality = 95	JPEG, quality = 95

TABLE 3.7: Pre-processing steps for the non-semantic test set

3. Intended Usage

- **Training:** Train a model solely on the 27 semantically-Overlapping labels in Dataset 03 or Dataset 04.
- **Evaluation:**
  - Semantic accuracy: Measure standard top-1/5 accuracy to name the dataset on the in-distribution test set (27 labels).
  - *Non-semantic ("name-the-dataset") accuracy: Evaluate the same model to name the dataset on the 15 950-image non-semantic test set (243 Labels).*

- **Anticipated Result Interpretation:**
  - High accuracy on the non-semantic test (243 Lables) set implies the model can still predict the source dataset (CIFAR-100 vs. TinyImageNet) in the absence of semantic information, indicating dependence on non-semantic inforation (e.g., resolution- or compression-induced artifacts).
  - Conversely, near-chance performance suggests minimal reliance on such non-semantic information.

The **Non-Semantic Test Set** provides a rigorous, dataset-controlled insight which uncovers how much non-semantic information is learned by the model. By mirroring the exact preprocessing of datasets 03 and 04, it enables precise quantification of how much generalisation is driven by genuine semantic understanding versus dataset bias introduced during image resizing, JPEG compression or other sampling processes.

3.3.3 Construction of Fully Integrated Datasets (Datasets 05 & 06)

To evaluate model performance across the full complement of CIFAR-100 and Tiny-ImageNet labels, two fully integrated datasets—Dataset 05 and Dataset 06—were constructed:

- **Extension beyond Datasets 03/04:** Both Dataset 05 and Dataset 06 expand the label space from the 27 shared labels used in Datasets 03/04 to all 273 labels (Fig.3.6, 27 shared + 73 CIFAR-100 exclusives + 173 TinyImageNet exclusives).
- **Analogy to the Dataset 03/04 distinction:** Just as Dataset 04 differs from Dataset 03 by inserting a 64×64 →32×32 down-sampling step for TinyImageNet before up-sampling (whereas Dataset 03 omits it), Dataset 06 similarly applies that same initial down-sampling to TinyImageNet images prior to bicubic up-sampling to 256×256. Dataset 05, in parallel with Dataset 03, omits this intermediate down-sampling.
- **Other procedures:** All other procedures—data aggregation, stratified 90 %/10 % train/validation split with fixed seed, bicubic up-sampling of CIFAR-100, JPEG storage at quality 95—remain identical across both benchmarks.

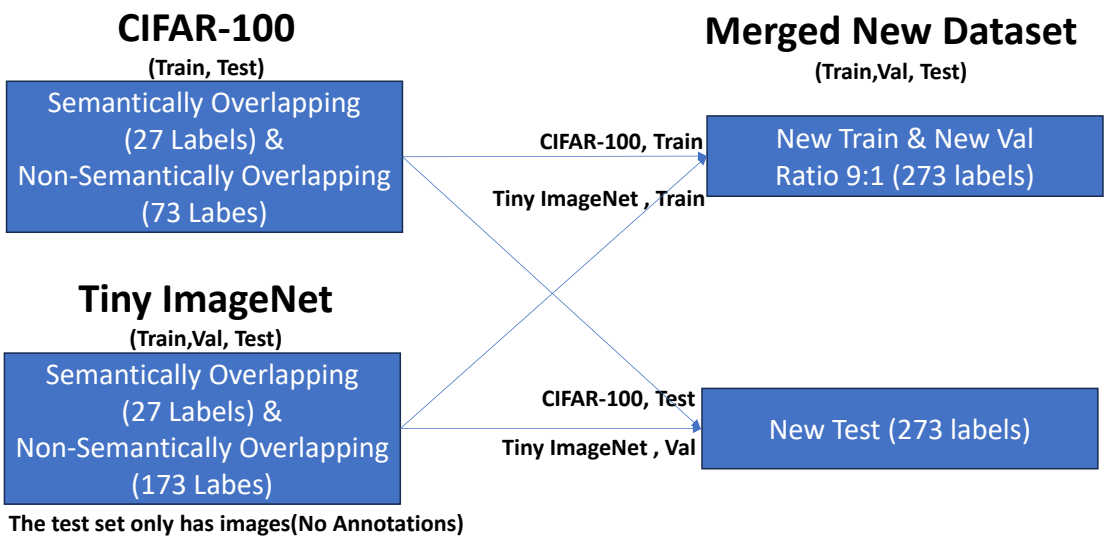


FIGURE 3.6: Train/Validation/Test Splits for the Merged Datasets(05 & 06)

3.3.3.1 Class Partitioning

The table (Tab.3.8) shows the number of semantically overlapping and non-semantically overlappin labels from each dataset.

Category	CIFAR-100	TinyImageNet	Total Used
Overlapping	27	27	27
Non-overlapping	73	173	246
Combined	100	200	**273**

TABLE 3.8: Class Partitioning for Fully Integrated Datasets

3.3.3.2 Train/Validation/Test Splits

As shown in Table 3.9, the test set retains all original test images—100 per CIFAR-100 class and 50 per TinyImageNet synset—for final evaluation, while the combined CIFAR-100 and TinyImageNet training data are split 90 %/10 % into training and validation subsets using a fixed random seed to ensure reproducibility.

3.3.3.3 Pre-processing & Saving Pipeline

The Table 3.10 below shows that the pre-processing pipeline of dataset 05/06 is identical to dataset 03/04

Dataset / Split	**Labels Included**	**Images per Label**	**Source Subset**	**Image Count**
CIFAR-100 (original train)	100	500	train	50 000
TinyImageNet (original train)	200	500	train	100 000
Train total (90 %)	–	–	merged 273-class train	135 000
Validation total (10 %)	–	–	merged 273-class train	15 000
CIFAR-100 (original test)	100	100	test	10 000
TinyImageNet (original val → test)	200	50	val (treated as test)	10 000
Test total	–	–	merged 273-class test	20 000
Overall total (all splits)	–	–	–	170 000

Table 3.9: Train/Validation/Test Splits for Fully Integrated Datasets (05 & 06)

Source	**Dataset 05**	**Dataset 06**
TinyImageNet images	$64 \times 64 \rightarrow 256 \times 256$ (bicubic upsample)	$64 \times 64 \rightarrow 32 \times 32$ (bicubic downsample) $\rightarrow 256 \times 256$ (bicubic upsample)
CIFAR-100 images	$32 \times 32 \rightarrow 256 \times 256$ (bicubic upsample)	$32 \times 32 \rightarrow 256 \times 256$ (bicubic upsample)
File format	JPEG, quality = 95	JPEG, quality = 95

Table 3.10: Pre-processing Pipeline for Datasets 05 & 06

By holding all elements of preprocessing, split proportions, and storage format constant—and varying only the full-vocabulary label set and the presence/absence of TinyImageNet down-sampling—Datasets 05 and 06 enable controlled comparison of model behavior under two standard interpolation regimes across the full 273-classification task.

3.3.4 Metadata File Record

To facilitate experimental reproducibility and sample tracking, a unified metadata file, `metadata.csv`, was generated for all splits of Datasets 03–06. This file records, for each image, a unique identifier, the original file path, the consolidated new label ID and

name, the source dataset's synset (label)) / fine ID and name, the dataset ID, and the split designation (train/validation/test/Non-semantic test). Employed as input to the `pytorch` data loader, it enables automated filtering by split and mapping of labels.

A representative snippet of the CSV header and one example row is shown in Table 3.11 below:

Field Name	Description
`image_id`	Unique identifier for each sample image
`filepath`	Relative or absolute path to the image file
`new_label_id`	Integer ID assigned to the consolidated label
`new_label`	Name of the consolidated label
`tiny_synset_id`	Original WordNet synset ID from TinyImageNet
`tiny_synset_name`	Original synset name from TinyImageNet
`cifar_fine_id`	Original "fine" label ID from CIFAR-100
`cifar_fine_name`	Original "fine" label name from CIFAR-100
`dataset_id`	Source dataset identifier (e.g., 0 - Tiny, 1 - CIFAR)
`split`	Data split designation (train/validation/test/non-semantic test)

TABLE 3.11: Metadata File Field Descriptions

3.4 Introduction of Used Model Architectures

3.4.1 ResNet Series Introduction

The ResNet family [He et al., 2016] consists of deep convolutional networks enhanced by residual connections, which help mitigate vanishing gradients and enable very deep models (e.g. ResNet-18, -34, -50, -101). In each residual block, the output is computed as

$$y = \mathcal{F}\big(x, \{W_i\}\big) + x, \tag{3.4.1}$$

where

- x is the input feature map of spatial size $H \times W$ with C channels.
- $\{W_i\}$ denotes the learnable parameters within this block (e.g. convolutional kernels, batch-norm weights and biases).
- $\mathcal{F}\big(x, \{W_i\}\big)$ is the residual function, typically a sequence of

  $$\text{Conv} \;\rightarrow\; \text{BatchNorm} \;\rightarrow\; \text{ReLU},$$

  often repeated twice or more.

- y is the output feature map, carrying both the original information and the learned residual.

This identity-mapping design preserves gradient flow and encourages feature reuse, leading to stronger accuracy on image classification tasks.

3.4.2 Vision Transformer (ViT) Series Introduction

The Vision Transformer [Dosovitskiy et al., 2020] splits an image into N fixed-size patches, projects each patch into a D-dimensional embedding, prepends a learnable `[CLS]` token, and adds positional embeddings. The resulting sequence is processed by L identical Transformer-encoder blocks, each combining multi-head self-attention and a feed-forward sublayer with residual connections and layer-normalization:

$$X^{(l)} = \underbrace{\mathrm{MSA}\big(\mathrm{LN}\big(X^{(l-1)}\big)\big) + X^{(l-1)}}_{\text{self-attention residual}} + \underbrace{\mathrm{MLP}\big(\mathrm{LN}\big(X^{(l-1)}\big)\big) + X^{(l-1)}}_{\text{feed-forward residual}}, \tag{3.4.2}$$

where

- N: number of image patches.
- D: embedding dimension for each patch (and the `[CLS]` token).
- `[CLS]`: a learnable classification token, serving as the aggregate representation.
- L: total number of Transformer encoder layers.
- $X^{(l-1)}$: output of layer $l-1$, input to layer l.
- $\mathrm{LN}(\cdot)$: Layer Normalization, applied to each token embedding.
- $\mathrm{MSA}(\cdot)$: multi-head self-attention, mapping $(N+1)\times D$ to $(N+1)\times D$.
- $\mathrm{MLP}(\cdot)$: two-layer feed-forward network (e.g. $D \to D_{\mathrm{ff}} \to D$) with a nonlinearity.
- $X^{(l)}$: output of layer l.

After L layers, the final-state `[CLS]` embedding $X^{(L)}_{[\mathrm{CLS}]}$ is passed to a linear classifier to predict the image's label.

3.5 Training Protocols

3.5.1 General Workflow for All Training Protocols

Figure 3.7 illustrates the overall training loop employed across all our experimental protocols. Each epoch begins by placing the model into training mode, where stochastic layers (e.g., dropout) and moving-average statistics (e.g., batch-normalization) are updated. The training set is then processed batch-by-batch: for each mini-batch, the network performs a forward pass to generate predictions, computes the corresponding loss against ground-truth targets, and then carries out a backward pass to propagate gradients. These gradients are used by the chosen optimizer (such as SGD or Adam) to update the model parameters immediately, before proceeding to the next batch. Once all batches in the current epoch have been processed, the model switches into validation mode in order to assess its performance on the held-out validation set without further weight updates.

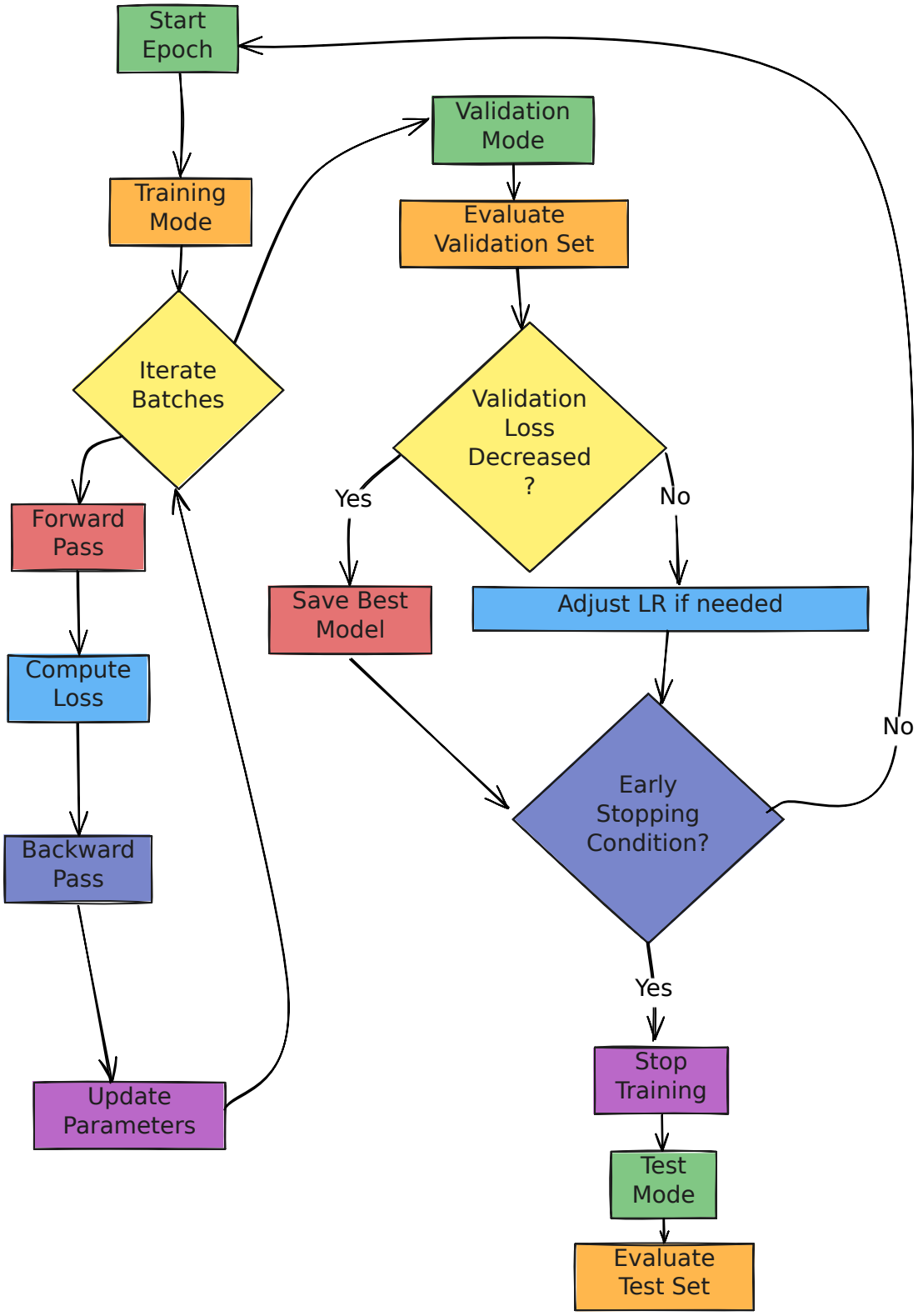


FIGURE 3.7: General Workflow for All Training Protocols

In validation mode, a single forward sweep over the validation data yields a scalar validation loss, which is compared against the best-recorded value so far. If the loss has decreased, the current model weights are checkpointed as the new best model; if not, a counter of unimproved epochs is incremented, and, when this counter reaches a predefined threshold (for example, three epochs), the learning rate is reduced according to a predetermined schedule (e.g., by a factor of ten). After saving or adjusting the learning rate, the procedure checks an early-stopping criterion—typically based on the number of successive non-improving epochs or a maximum epoch cap. If this criterion is met, training halts, the model reenters test mode, and a final performance evaluation is conducted on the test set. Otherwise, the cycle repeats with the next epoch, thereby ensuring that training proceeds only as long as genuine improvements are observed on the validation data.

3.5.2 Protocol A: ResNet Semantic Classifier Training

The objective of protocol A is to train ResNet-18/34/50/101 from scratch to build semantic classifiers on the target datasets (Datasets 03–06) and uniformly compare the impact of different network depths and preprocessing schemes on model performance.

To support these goals, the following environment and computational resources are specified:

1. **Environment & Resources**
    - Based on the `PyTorch` framework.
    - Prefer GPU (CUDA); otherwise CPU. When using GPU, enable `pin_memory=True` and set `num_workers=8` for data loading.
    - Batch size fixed at 128 samples per batch.
2. **Data Preprocessing & Augmentation**
    - **Loading**: Load each split (train/validation/test) via a unified CSV metadata file; shuffle only the training set, leave validation and test in original order.
    - **Training phase**:
        - Randomly resize and crop to 224×224.
        - Random horizontal flip.
        - Convert to tensor.
        - Normalize using ImageNet mean $[0.485, 0.456, 0.406]$ and std $[0.229, 0.224, 0.225]$.
    - **Validation/Test phase**:

- Center-crop to 224×224.
- Convert to tensor.
- Apply the same normalization.

3. **Model Architecture & Optimization Settings Without Dataset Bias Mitigation**

   - **Architecture**
     - ResNet-18, ResNet-34, ResNet-50 and ResNet-101 are trained from scratch.
     - The original final fully-connected layer is replaced by a linear layer of dimension equal to the number of classes (27 for Datasets 03/04; 273 for Datasets 05/06).
   - **Loss & Optimizer**
     - *Loss function*: Name the Label loss $\mathscr{L}_{\text{y}} = \text{CrossEntropy}(\hat{y}, y)$ [2][He et al., 2016]
     - *Optimizer*: Adam [Kingma and Ba, 2014] with initial learning rate 1×10^{-3}.
     - *Learning-rate scheduler*: `ReduceLROnPlateau` [Paszke et al., 2019] monitoring validation loss (patience = 3; factor = 0.1).
     - *Epochs*: up to 60.
     - *Early stopping* [Prechelt, 1998]: halt if validation loss fails to decrease by at least 1×10^{-4} for 10 consecutive epochs.

4. **Model Architecture & Optimization Settings With Dataset Bias Mitigation (*Before Reading This Part, Read Chapter 6 First*)**

   - **Architecture**
     - A domain-adversarial variant (Fig.6.1) [Ganin et al., 2016] of ResNet-18, ResNet-34, ResNet-50 or ResNet-101 is employed.
     - The backbone comprises all layers of the chosen ResNet up to (but excluding) its final *fully-connected* layer (the global pooling layer is kept).
     - Two parallel heads are appended to the backbone:
       * A *Name the Label head*, projecting the pooled feature embedding to the $C = 27/273$ semantic label classes.
       * A *Name the Dataset head*, projecting the same embedding (after Gradient Reversal) to the $D = 2$ dataset domains.

[2] y is the one-hot true label vector of an image, while $\hat{y}$ is a model's label prediction vector of an image after softmax

– Gradient Reversal Layer (GRL) [Ganin et al., 2016]: included only in the One-hot Minimax Dataset Bias Mitigation method to invert gradients flowing from the Name the Dataset head into the feature extractor (backbone); omitted in the Uniform Double Mini method.

- **Loss & Optimizer**
  - *Loss: One-hot Minimax Dataset Bias Mitigation Method*
    - *Name the Label loss*: $\mathscr{L}_{\mathrm{y}} = \mathrm{CrossEntropy}(\hat{y}, y)$.[3]
    - *One-hot Minimax Name the Dataset loss*: $\mathscr{L}_{\mathrm{d1}} = \mathrm{CrossEntropy}(\hat{d}, d)$.[4]
    - *One-hot Minimax Backbone loss*: $\mathscr{L}_{\mathrm{minimax}} = \mathscr{L}_{\mathrm{y}} - \lambda\,\mathscr{L}_{\mathrm{d1}}$.
  - *Loss: Uniform Double Mini Dataset Bias Mitigation Method*
    - *Name the Label loss*: $\mathscr{L}_{\mathrm{y}} = \mathrm{CrossEntropy}(\hat{y}, y)$.[5]
    - *Uniform Double Mini Name the Dataset loss*: $\mathscr{L}_{\mathrm{d2}} = \mathrm{CrossEntropy}(\hat{d}, u)$.[6]
    - *Uniform Double Mini Backbone loss*: $\mathscr{L}_{\mathrm{UDM}} = \mathscr{L}_{\mathrm{y}} + \lambda\,\mathscr{L}_{\mathrm{d2}}$.
  - *Optimizer*: Adam [Kingma and Ba, 2014] with three parameter groups:
    - Backbone parameters, $\mathrm{lr_b} = 1 \times 10^{-3}$.
    - Classification-head parameters, $\mathrm{lr_c} = 1 \times 10^{-3}$.
    - Domain-head parameters, $\mathrm{lr_d} = 1 \times 10^{-3}$.
  - *Learning-rate scheduler*: `ReduceLROnPlateau` [Paszke et al., 2019] monitoring validation *Name the Label* loss, with factor = 0.1 and patience = 3.
  - *GRL coefficient schedule* [Ganin et al., 2016]:

$$\lambda(p) \;=\; \lambda_{\max}\left(2/(1+e^{-\gamma p}) - 1\right), \quad p = \frac{e-1}{E},$$

    where e is the current epoch, E the total epochs, $\lambda_{\max} = 0.36$, and $\gamma = 2.0$.
  - *Epochs*: up to 60.
  - *Early stopping* [Prechelt, 1998]: terminate if validation Name the Label loss fails to improve by at least 1×10^{-4} for 10 consecutive epochs; the model with the lowest validation Name the Label is retained.

5. **Training & Evaluation Workflow**

- **Training loop**: For each epoch:

[3] y is the one-hot true label vector; $\hat{y}$ is the model's predicted label distribution after softmax.
[4] d is the one-hot true dataset ID vector; $\hat{d}$ is the model's predicted dataset distribution after softmax.
[5] Definitions as above.
[6] $u = \left[\frac{1}{K}, \ldots, \frac{1}{K}\right]^{\top} \in \mathbb{R}^K$ is the uniform distribution over K source datasets; $\hat{d}$ is the model's predicted dataset distribution after softmax.

  - Perform a training pass (forward → backward → update).
  - Evaluate on the validation set.
  - Record training/validation loss and Top-1/Top-5 accuracy.
  - Display metrics via a progress bar.
- **Model saving**:
  - Save the *best model* whenever validation loss reaches a new minimum and reset the early-stop counter.
  - If no improvement for 10 consecutive epochs, stop early.
  - Save a checkpoint every 10 epochs.
- **Final test**:
  - Load the best model after training.
  - Evaluate on the test set.
  - Report test loss.
  - Report Top-1 and Top-5 accuracy.
  - Specify the epoch and value of the best validation loss.

3.5.3 Protocol B: ViT Semantic Classifier Training

The objective of protocol B is to train a Vision Transformer (ViT-B/32) from scratch on the target datasets (Datasets 03–06) to build semantic classifiers and provide a transformer-based baseline directly comparable to the ResNet results obtained under Protocol A.

To support these goals, the following environment and computational resources are specified:

1. **Environment & Resources**
   - Based on the `PyTorch` framework.
   - Prefer GPU (CUDA); otherwise CPU. When using GPU, enable `pin_memory=True` and set `num_workers=8` for data loading.
   - Batch size fixed at 128 samples per batch.
2. **Data Preprocessing & Augmentation (Same as Protocol A)**
   - **Loading**: Load each split (train/validation/test) via a unified CSV metadata file; shuffle only the training set, leave validation and test in original order.
   - **Training phase**:

  - Randomly resize and crop to 224×224.
  - Random horizontal flip.
  - Convert to tensor.
  - Normalize using ImageNet mean $[0.485, 0.456, 0.406]$ and std $[0.229, 0.224, 0.225]$.
- **Validation/Test phase**:
  - Center-crop to 224×224.
  - Convert to tensor.
  - Apply the same normalization.

3. **Model Architecture & Optimization Settings Without Dataset Bias Mitigation**

- **Architecture**
  - ViT-B/32 is trained *from scratch*.
  - The default classification head is replaced by a linear layer whose output dimension equals the number of semantic classes ($C = 27$ for Datasets 03/04; $C = 273$ for Datasets 05/06).
- **Loss & Optimizer**
  - *Loss function*: *Name the Label loss* $\mathscr{L}_{\text{y}} = \text{CrossEntropy}(\hat{y}, y)$[7] [Goodfellow et al., 2016].
  - *Optimizer*: AdamW [Loshchilov and Hutter, 2017] with learning rate 4×10^{-5} and weight decay 0.05.
  - *Learning-rate schedule*.

    Warm-up: `LinearLR` from factor 0.01 to 1.0 during the first 5 epochs.

    Main: `CosineAnnealingLR` [Loshchilov and Hutter, 2016] with $T_{\max} = 55$ and $\eta_{\min} = 1 \times 10^{-6}$.

    Composition: `SequentialLR` switches from warm-up to cosine after epoch 5.
  - *Epochs*: up to 60.
  - *Early stopping*: halt if validation loss fails to improve by at least 1×10^{-4} for 10 consecutive epochs.

4. **Model Architecture & Optimization Settings With Dataset Bias Mitigation (*Before Reading This Part, Read Chapter 6 First*)**

- **Architecture**
  - A domain-adversarial variant (Fig.6.1) of ViT-B/32 is employed.

[7] y denotes the one-hot ground-truth label, and $\hat{y}$ the predicted label distribution after softmax.

  - The backbone comprises ViT-B/32 up to (and including) the global-pooling layer; its original classification head is removed
  - Two parallel heads are appended to the backbone:
    * A *Name the Label head*, projecting the pooled feature embedding to the $C = 27/273$ semantic label classes.
    * A *Name the Dataset head*, projecting the same embedding (after Gradient Reversal) to the $D = 2$ dataset domains.
  - Gradient Reversal Layer (GRL) [Ganin et al., 2016]: included only in the One-hot Minimax Dataset Bias Mitigation method to invert gradients flowing from the Name the Dataset head into the feature extractor (backbone); omitted in the Uniform Double Mini method.

- **Loss & Optimizer**
  - *Loss: One-hot Minimax Dataset Bias Mitigation Method*
    * *Name the Label loss*: $\mathscr{L}_{\mathrm{y}} = \mathrm{CrossEntropy}(\hat{y}, y)$.[8]
    * *One-hot Minimax Name the Dataset loss*: $\mathscr{L}_{\mathrm{d1}} = \mathrm{CrossEntropy}(\hat{d}, d)$.[9]
    * *One-hot Minimax Backbone loss*: $\mathscr{L}_{\mathrm{minimax}} = \mathscr{L}_{\mathrm{y}} - \lambda \mathscr{L}_{\mathrm{d1}}$.
  - *Loss: Uniform Double Mini Dataset Bias Mitigation Method*
    * *Name the Label loss*: $\mathscr{L}_{\mathrm{y}} = \mathrm{CrossEntropy}(\hat{y}, y)$.[10]
    * *Uniform Double Mini Name the Dataset loss*: $\mathscr{L}_{\mathrm{d2}} = \mathrm{CrossEntropy}(\hat{d}, u)$.[11]
    * *Uniform Double Mini Backbone loss*: $\mathscr{L}_{\mathrm{UDM}} = \mathscr{L}_{\mathrm{y}} + \lambda \mathscr{L}_{\mathrm{d2}}$.
  - *Optimizer*: Adam with three parameter groups:
    * Backbone parameters, $\mathrm{lr_b} = 1 \times 10^{-3}$.
    * Classification-head parameters, $\mathrm{lr_c} = 1 \times 10^{-3}$.
    * Domain-head parameters, $\mathrm{lr_d} = 1 \times 10^{-3}$.
  - *Learning-rate schedule.*

    Warm-up: `LinearLR` from factor 0.01 to 1.0 during the first 5 epochs.

    Main: `CosineAnnealingLR` [Loshchilov and Hutter, 2016] with $T_{\max} = 55$ and $\eta_{\min} = 1 \times 10^{-6}$.

    Composition: `SequentialLR` switches from warm-up to cosine after epoch 5.
  - *Epochs*: up to 60.
  - *GRL coefficient schedule* [Ganin et al., 2016]:

$$\lambda(p) \;=\; \lambda_{\max}\left(2/(1+e^{-\gamma p}) - 1\right), \quad p = \frac{e-1}{E},$$

[8] y is the one-hot true label vector; $\hat{y}$ is the model's predicted label distribution after softmax.

[9] d is the one-hot true dataset ID vector; $\hat{d}$ is the model's predicted dataset distribution after softmax.

[10] Definitions as above.

[11] $u = \left[\frac{1}{K}, \ldots, \frac{1}{K}\right]^{\top} \in \mathbb{R}^K$ is the uniform distribution over K source datasets; $\hat{d}$ is the model's predicted dataset distribution after softmax.

where e is the current epoch, E the total epochs, $\lambda_{\max} = 0.36$, and $\gamma = 2.0$.

- *Early stopping* [Prechelt, 1998]: monitor the *Name the Label loss* on the validation set; terminate if no improvement of at least 1×10^{-4} for 10 consecutive epochs.

5. **Training & Evaluation Workflow**

- **Training loop**: For each epoch:
  - Perform a training pass (forward → backward → update).
  - Evaluate on the validation set.
  - Record training/validation loss and Top-1/Top-5 accuracy.
  - Display metrics via a progress bar.
- **Model saving**:
  - Save `best_vit_b32.pth` whenever validation loss reaches a new minimum and reset the early-stop counter.
  - If no improvement for 10 consecutive epochs, terminate training early.
  - Save periodic checkpoint `vit_b32_epoch_{E}.pth` every 10 epochs.
- **Final test**:
  - Load the best model after training.
  - Evaluate on the test set.
  - Report test loss.
  - Report Top-1 and Top-5 accuracy.
  - Specify the epoch and value of the best validation loss.

3.5.4 Protocol C: Linear Probe Training

In order to accurately evaluate the semantic information and dataset-specific characteristics encoded at different network depths, the backbone of a model is frozen and a single-layer fully connected classifier is attached to the output of each specified layer for training—an approach known as a linear probe [Alain and Bengio, 2016]. Comparing each probe's performance on the "Name the Label" (27- or 273-way classification) and "Name the Dataset" (2-way classification) tasks reveals how semantic and dataset biases are distributed throughout different layers of the network.

To support these goals, the following environment and computational resources are specified:

1. **Environment & Resources**

   - Based on the `PyTorch` framework.
   - Prefer GPU (CUDA); otherwise CPU. When using GPU, enable `pin_memory=True` and set `num_workers=8` for data loading.
   - Batch size fixed at 128 samples per batch.

2. **Data Preprocessing & Augmentation (Same as Protocol A)**

   - **Loading**: Load each split (train/validation/test) via a unified CSV metadata file; shuffle only the training set, leave validation and test in original order.
   - **Training phase**:
     - Randomly resize and crop to 224×224.
     - Random horizontal flip.
     - Convert to tensor.
     - Normalize using ImageNet mean $[0.485, 0.456, 0.406]$ and std $[0.229, 0.224, 0.225]$.
   - **Validation/Test phase**:
     - Center-crop to 224×224.
     - Convert to tensor.
     - Apply the same normalization.

3. **Model & Probe Initialization**

   - **Backbone**:
     - Use a ResNet-18/34/50/101 or ViT-B/32 model trained under Protocol A or B.
     - Freeze all backbone parameters (`requires_grad=False`) and set to `model.eval()` to fix batch-norm statistics.
   - **Probe Construction**:
     - **Tap selection**: choose several intermediate points in the backbone (e.g., output of `layer1.0`, `layer4.1` for ResNet; the output of transformer block ℓ for ViT).
     - **Hook registration**: attach a forward-hook at each tap so that its output tensor $\mathbf{h}$ is captured during the network's forward pass.
     - **Feature extraction**
       - **Convolutional outputs (e.g., ResNet)**: The captured tensor $\mathbf{h}$ has shape (B, C, H, W):

$$\mathbf{h} \in \mathbb{R}^{B \times C \times H \times W},$$

where B is the batch size, C the number of channels, and H, W the spatial height and width. Average each channel over its $H \times W$ grid:

$$f_{b,i} = \frac{1}{HW} \sum_{u=1}^{H} \sum_{v=1}^{W} h_{b,i,u,v}, \quad i = 1, \ldots, C,$$

yielding $\mathbf{f} \in \mathbb{R}^{B \times C}$.

* **Transformer outputs (e.g., ViT)**: The captured tensor $\mathbf{h}$ has shape (B, T, D):

$$\mathbf{h} \in \mathbb{R}^{B \times T \times D},$$

where T is the number of tokens (patch embeddings plus the [CLS] token) and D the embedding dimension. Extract the [CLS] token:

$$f_{b,*} = h_{b,0,*},$$

yielding $\mathbf{f} \in \mathbb{R}^{B \times D}$.

– **Linear head**

Feed the resulting (B, d) tensor into a fully connected layer

$$\mathbb{R}^d \longrightarrow \mathbb{R}^K,$$

where $d = C$ for ResNet or $d = D$ for ViT, and $K = 27/273$ ("Name the Label") or $K = 2$ ("Name the Dataset"). Only this layer's parameters are trained. The Table 3.12 summarizes how the output shape of feature extraction changes in order to fit the input shape requirements of linear probes.

Backbone & Tap	**Raw output shape**	**Processed feature shape**	d
ResNet-50 @ layer3[1]	$(B, 1024, 14, 14)$	$(B, 1024)$	1024
ViT-B/32 @ Block 6 [CLS]	$(B, 197, 768)$	$(B, 768)$	768

TABLE 3.12: Linear Probe Input Feature Shape Summary

4. **Loss & Optimizer**

- *Loss function*: Cross Entropy [He et al., 2016]
- *Optimizer*: Adam [Kingma and Ba, 2014] with initial learning rate 1×10^{-3}.
- *Learning-rate scheduler*: `ReduceLROnPlateau` [Paszke et al., 2019] monitoring validation loss (patience = 2; factor = 0.1).

- *Epochs*: up to 30.
- *Early stopping* [Prechelt, 1998]: halt if validation loss fails to decrease by at least 1×10^{-6} for 5 consecutive epochs.

5. **Training & Evaluation Workflow**

   - **Training loop**: For each epoch:
     - Perform a training pass (forward $\rightarrow$ backward $\rightarrow$ update) updating only the probe and keep the backbone frozen.
     - Evaluate on the validation set.
     - Record training/validation loss and Top-1/5 accuracy.
     - Display metrics via a progress bar.
   - **Probe saving**:
     - Save weights whenever validation loss reaches a new minimum and reset the early-stop counter.
     - If validation loss does not improve by $\geq 1 \times 10^{-6}$ for 5 consecutive epochs, stop early.
   - **Testing**:
     - Load the best weights after training.
     - Evaluate on the test set.
     - Report test loss.
     - Report Top-1/5 accuracy and confusion matrix.
     - Specify the epoch and value of the best validation loss.

Chapter 4

Experiments to Investigate Dataset Bias

4.1 Sectional Introduction

This chapter presents a series of experiments designed to investigate dataset biases in modern deep neural networks. By replicating and extending Torralba & Efros's Name That Dataset (2011) study, these experiments explore how semantic and non-semantic information influences model performance, the distribution of dataset biases across network layers, and the impact of label diversity on dataset identification. The experiments employ ResNet and Vision Transformer (ViT) architectures, with linear probing techniques to dissect the learned representations. The following experiments are included:

- **Experiment 1: Name the Dataset with DNN** – Validates dataset biases in modern deep neural networks by training a ResNet-18 to classify 27 semantically overlapping labels and linear probing its ability to Name the Dataset.
- **Experiment 2: Non-semantic Test** – Quantifies the reliance on non-semantic information to Name the Dataset by evaluating the pretrained ResNet-18 from Experiment 1 on a test set of 246 non-semantically overlapping labels.
- **Experiment 3: Cross Layer Linear Probing** – Investigates the distribution of semantic and non-semantic information across ResNet and ViT layers by attaching linear probes at multiple depths to both Name the label Name the Dataset.

- **Experiment 4: Two Full Datasets** – Compares the impact of pretraining ResNet-18 on 27 (semantically overlapping only) versus 273 (semantically overlapping + non-semantically overlapping) labels to assess whether increased label diversity enhances the model's ability to Name the Dataset.

4.2 Experiment 1: Name the Dataset with DNN

4.2.1 Experiment 1: Objective

This experiment validates the existence of dataset biases in modern deep neural networks by replicating Torralba & Efros's "Name That Dataset" (2011) study, substituting their original feature extractors + SVM with a ResNet-18 classifier.

4.2.2 Experiment 1: Workflow

The overall workflow for reproducing "Name the Dataset" with a ResNet-18 backbone is illustrated in the Figure 4.1 below:

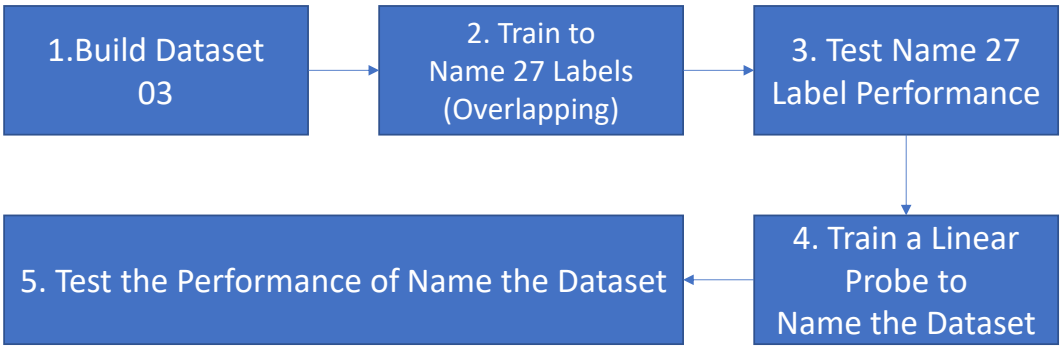


FIGURE 4.1: Experiment 1: Train to Name the Dataset Workflow

1. **Construct Dataset 03**
   Build the semantically-overlapping Dataset 03 from TinyImageNet and CIFAR-100 exactly as defined in Section 3.3.2. Crucially, the mapping (selection) of semantically overlapping labels intentionally incorporates labels exhibiting partial semantic overlap (Tab.3.2; e.g., "ox" , "cattle"), allowing nuanced analysis of semantic relatedness effects.

2. **Train to Name 27 Labels (Overlapping)**
   Use Protocol A: ResNet Semantic Classifier Training (Section 3.5.2) to train a ResNet-18 (or variant) from scratch on the Dataset 03 (27 merged labels) to Name the 27 Labels.

3. **Evaluate Label Classification Performance**
   After training finishes (or early-stops), evaluate the best checkpoint on the 27-label test split and record Top-1/Top-5 semantics accuracy.

4. **Train a Linear Probe to Name the Dataset**
   Following Protocol C: Linear Probe Training (Section 3.5.4), freeze all backbone layers of the trained model in step 2. Replace the final fully-connected layer with a new linear classifier (2-way output: CIFAR vs. Tiny) and train only this linear probe (new linear classifier) to name the dataset.

5. **Evaluate Dataset Naming Performance**
   Test the Name the Dataset performance on the same 27-label test split to measure how well the source-dataset of an image sample can be predicted.

4.2.3 Experiment 1: Results, Analysis & Discussion

4.2.3.1 Name 27 Label Results

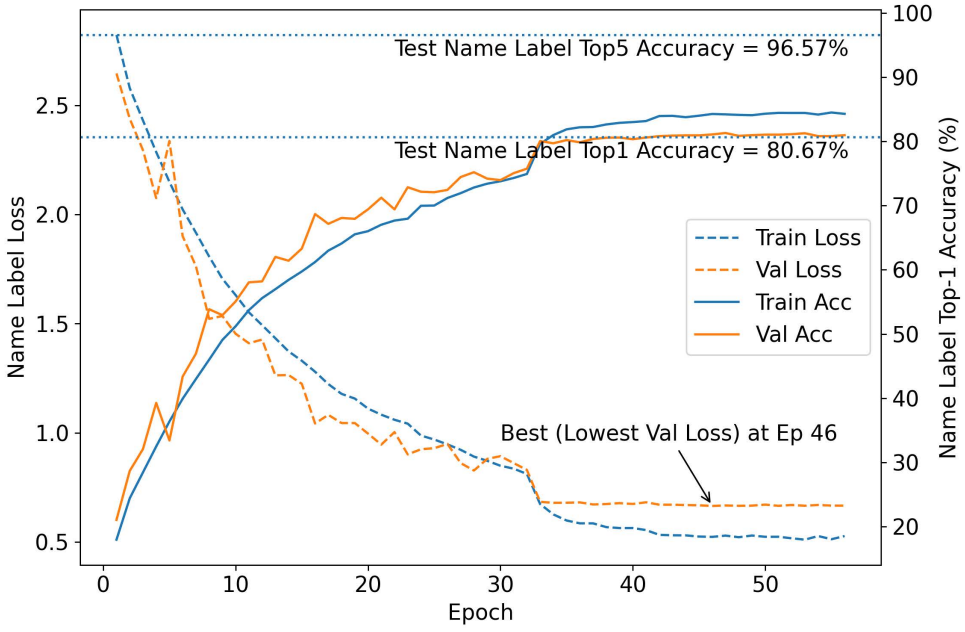


FIGURE 4.2: Train & Test Log of a ResNet-18 trained with Protocol A (Section 3.5.2) on Dataset 03 to Name 27 Label (1803L)

4.2.3.2 Name 27 Label Analysis

The training log in Figure 4.2 shows two straight horizontal line and four curves: Test Top1/5(blue dotted) , training loss (blue dashed), validation loss (orange dashed), training Top-1 accuracy (blue solid) and validation Top-1 accuracy (orange solid) over 55 epochs.

- **Test Performance:** Horizontal reference lines mark test Top-1 (80.67%) and Top-5 (96.57%) accuracies.
- **Loss Curves:** The training loss decreases rapidly from about 2.8 to around 1.0 within the first $\sim$ 30 epochs, then gradually declines to a minimum of approximately 0.52 at epoch 46. The validation loss follows a similar trend, with a small uptick to around 1.55 at epoch 5 before falling steadily to its lowest point (approximately 0.77) at epoch 46, indicating that the best checkpoint (early stopping) occurs around this epoch.
- **Accuracy Curves:** Training Top-1 accuracy climbs from roughly 22% to about 85%, while validation Top-1 accuracy rises from approximately 10% to around 82%. The gap between training and validation accuracy is largest (around 15–20 percentage points) during epochs 0–10, then narrows to about 3–5 percentage points by epochs 20–55, demonstrating improved generalization.

4.2.3.3 Name 27 Label Discussion

- **Effective learning and early stopping:** The model learns rapidly in the first 30 epochs and reaches its best validation performance at epoch 46, indicating that training beyond this point yields diminishing returns.
- **Strong generalization:** The high held-out test Top-1 (80.67%) and Top-5 (96.57%) accuracies, suggesting consistent generalization.

4.2.3.4 Name the Dataset Train & Test Log Results

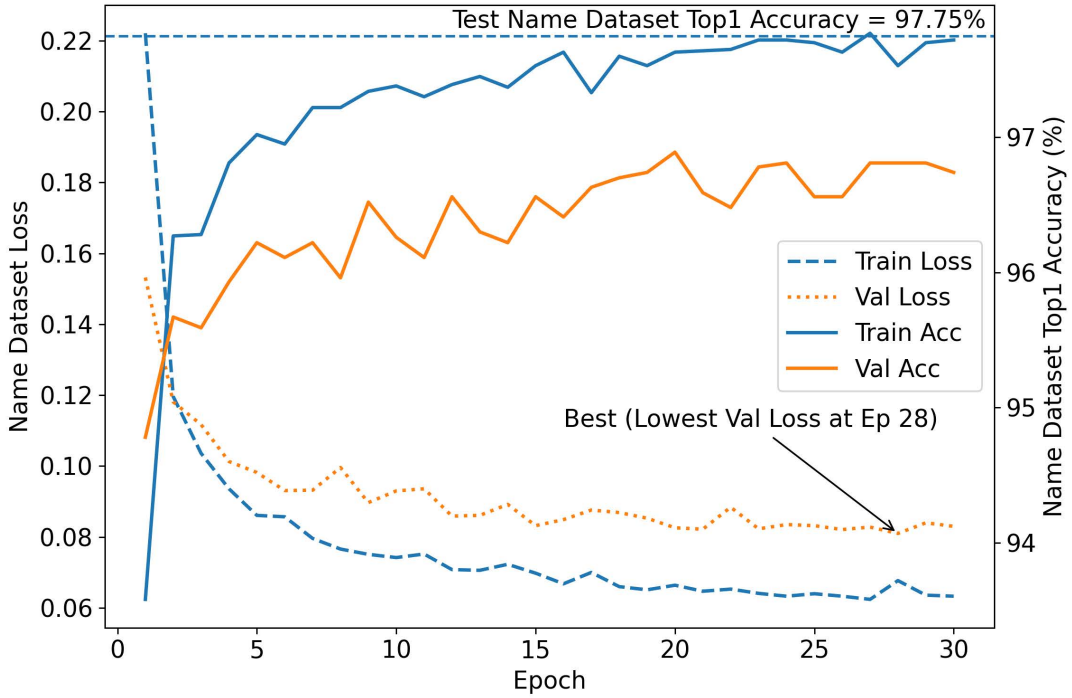


FIGURE 4.3: Train & Test Log of the ResNet-18 Pre-trained to Name 27 Labels and Fine-tuned with Protocol C (Section 3.5.4) on Dataset 03 to Name the Label (1803D)

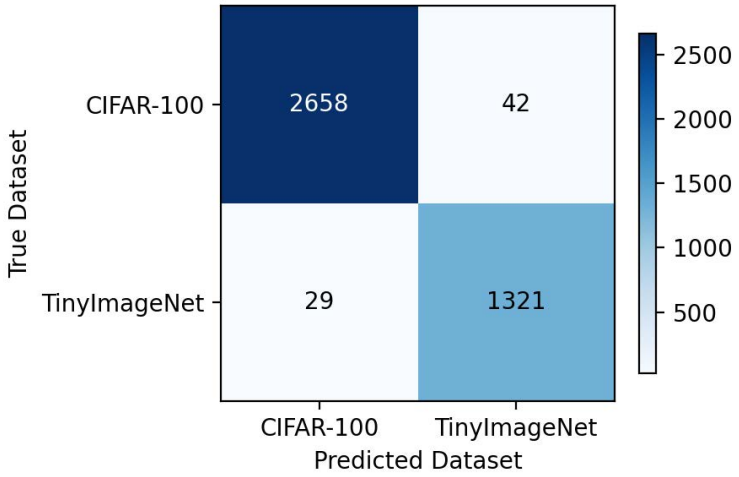
4.2.3.5 Name the Dataset Train & Test Log Analysis

- **Rapid Convergence:** Unlike the Name the Label task in Figure 4.2 whose validation Top-1 accuracy climbs gradually epoch by epoch, the Name the Dataset task in Figure 4.3 achieves over 90% validation Top-1 accuracy after **just one epoch**.
- **Loss Curves:** The training loss (blue dashed) drops sharply from about 0.22 to $\sim$ 0.10 by epoch 3, then slowly decreases to a minimum of $\approx$ 0.06 by epoch 30. Validation loss (orange dotted) also falls rapidly to $\approx$ 0.09 at epoch 3, then oscillates around 0.08–0.10.
- **Accuracy Curves:** Training Top-1 accuracy (blue solid) climbs from $\sim$ 93.5% at epoch 1 to steadily approach to less than 98% by 30. Validation Top-1 accuracy (orange solid) jumps to over 90% in epoch 1, then fluctuates in the 94%–97% range.
- **Test Performance:** A horizontal line marks test Top-1 accuracy at 97.75%, confirming that the Name the Dataset model generalizes exceedingly well to held-out samples.

4.2.3.6 Name the Dataset Train & Test Log Discussion

The Name the Dataset model's rapid attainment of high accuracy implies that the ResNet-18 backbone, trained on the semantic label task, has learned representations that readily separate the two source datasets (CIFAR vs. TinyImageNet). Early epoch performance above 90% demonstrates minimal additional learning is required, and the final test Top-1 of 97.75% underscores the backbone's robustness in capturing dataset-specific signatures.

4.2.3.7 Name the Dataset Multi-dimensional Test Results



Note: Each cell shows the number of support images predicted to belong to the dataset on the x-axis, given their true origin on the y-axis.

FIGURE 4.4: Test Confusion Matrix of the ResNet-18 Pre-trained to Name 27 Labels and Fine-tuned with Protocol C (Section 3.5.4) on Dataset 03 to Name the Dataset (1803D)

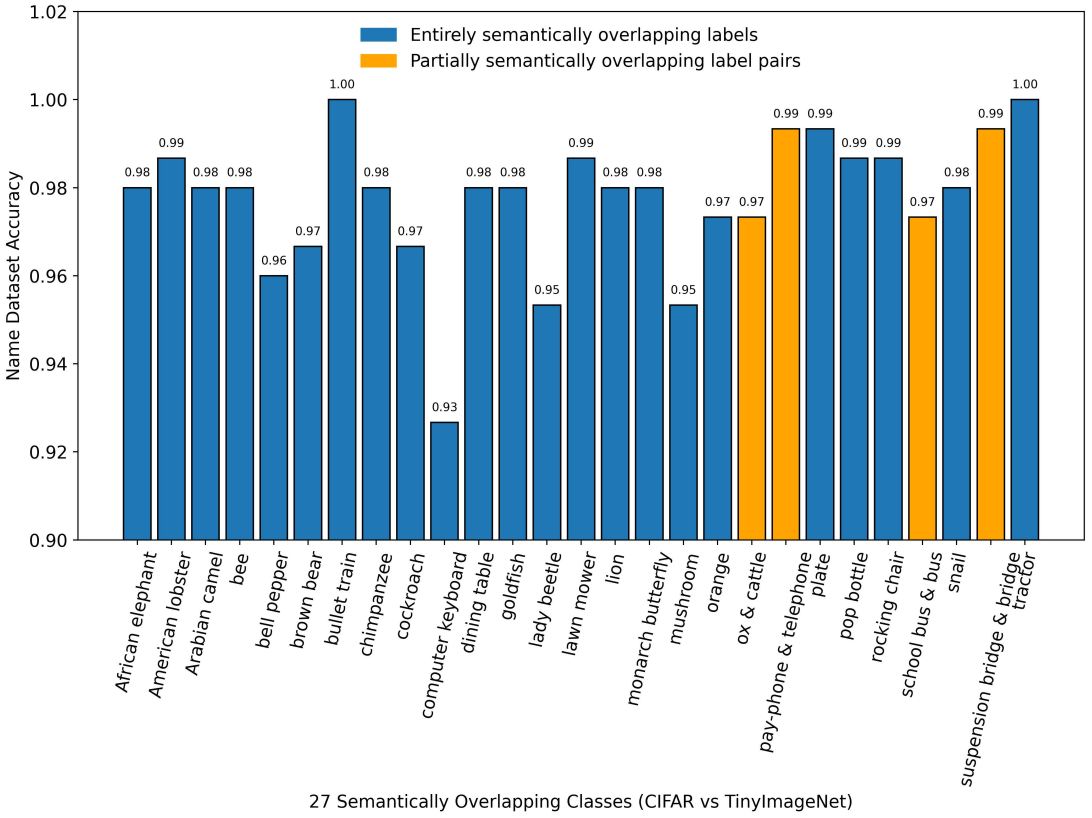


FIGURE 4.5: Test Per-label Name Dataset Accuracy of a ResNet-18 Fine-tuned with Protocol C (Section 3.5.4) on Dataset 03 to Name the Dataset (1803D)

4.2.3.8 Name the Dataset Multi-dimensional Test Analysis

- **Overall Performance:** CIFAR-100 samples are misclassified at a rate of 1.56% (42/2700), while TinyImageNet samples are misclassified at 2.15% (29/1400).
- **Partial vs. Full Overlap:** The 4 *partially* semantically overlapping label pairs exhibit nearly identical accuracy to the 23 *entire* overlapping labels.
- **Label-Level Distribution:**
  - **Lowest Accuracy Labels:** *computer keyboard* achieves only 93% accuracy, and the sets {*goldfish*, *mushroom*, *orange*} averages 95%.
  - **Accuracy Gap:** The mean accuracy difference between the four "partially overlapping" labels (97.8%) and the 23 "fully overlapping" labels (98.0%) is only 0.2 percentage points !

4.2.3.9 Name the Dataset Multi-dimensional Test Discussion

Dataset biases persist in modern datasets and deep neural networks:
Entire backbone frozen; the original fully-connected layer is replaced with a new two-way linear classifier, and only this linear probe is trained to predict the source dataset. Even so, it still distinguishes CIFAR-100 from TinyImageNet at over 97% accuracy. This shows that features learned for the semantic classification (Name the Label) task continue to carry dataset biases—such as textural, sampling, or compression artifacts—that indicate the source dataset.

Semantic overlap has minimal impact on dataset identification:
Labels with complete semantic overlap and those with only partial semantic overlap (e.g., "ox" vs. "cattle," "school bus" vs. "bus," "pay-phone" vs. "telephone") all exhibit comparable accuracy—contrary to the expectation that the model would exploit subtle visual distinctions (e.g., between a school bus and a regular bus) to achieve higher Name the Dataset accuracy. *This insight motivates the next experiment:* ***the Non-Semantic Test****.*

4.3 Experiment 2: Non-Semantic Test

4.3.1 Introduction on Semantic and Non-Semantic Information

For computer vision, *semantic information* denotes high-level, task-specific features that encapsulate an image's conceptual content—such as object shape and spatial layouts—which enable models to correctly identify and differentiate classes of an image based on their inherent meaning; In contrast, *non-semantic information* comprises incidental or artifactual cues bearing no direct connection to object identity—for example, sensor noise signatures, JPEG compression artifacts, dataset-specific colour histograms, vignette or ring-effect patterns, and uniform background textures [Hosseini and Poovendran, 2018, Wang et al., 2023]. While semantic features support a model's ability to generalise across diverse visual contexts, non-semantic signals can inadvertently serve as spurious shortcuts [Ge et al., 2021, Joshi et al., 2019] allowing classifiers to exploit dataset bias and boost in-domain performance without true semantic understanding. Consequently, separating these two types of information is critical for evaluating model robustness and ensuring genuinely interpretable learning.

Table 4.1 summarizes the key distinctions between semantic and non-semantic information in vision models.

Category	**Semantic Information**	**Non-Semantic Information**
Definition	High-level, meaningful features ("what it is").	Incidental or artifactual signals unrelated to object identity.
Network level	Ideally, dominant in deeper layers (but emerges from the first layers onward).	Ideally, dominant in early layers (yet residuals can persist to high layers).
Role	Enables true label discrimination and cross-domain generalisation.	Serves as a low-level feature basis—but may be exploited as "shortcuts," harming generalisation and interpretability.
Examples	Object shape and silhouette Texture patterns (fur, fabric) Spatial arrangement and context	Raw RGB pixel distributions Sensor noise patterns JPEG compression artefacts Vignette or ring-effect signatures
Impact if relied on	Robust recognition across varied scenes.	Spurious in-domain gains without genuine semantic understanding.

TABLE 4.1: Comparison of Semantic vs. Non-Semantic Information

4.3.2 Experiment 2: Objective

To quantify the extent to which the pretrained Name the Dataset probes relies exclusively on non-semantic information—i.e. interpolation artifacts, JPEG compression signatures, sensor noise—by evaluating the pretrained Name the Dataset linear probe in experiment 1 (Fig.4.1) trained on Dataset 03's 27 overlapping classes against a held-out Test set of non-semantic overlapping labels (73 CIFAR-only + 173 Tiny-only).

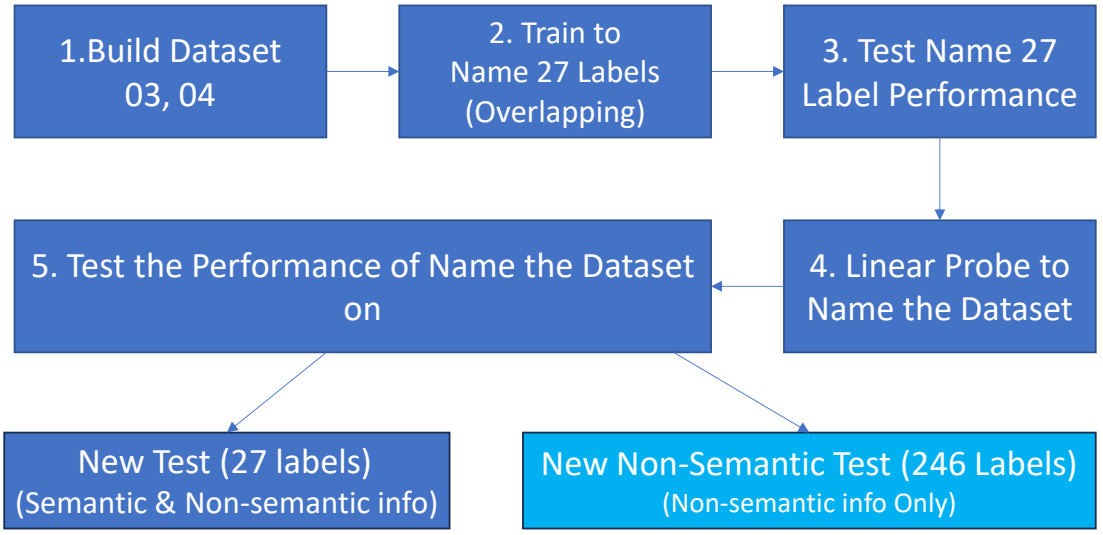
4.3.3 Experimental 2: Workflow

The overall workflow for evaluating the pretrained Name the Dataset linear probe on two test sets is illustrated in Figure 4.6 below; Dark blue standing for works done in Experiment 1 and light blue new procedure in Experiment 2:

1. **Construct the Two Test Sets**

(a) **New Test (27 labels; Semantic + Non-semantic info):**
Select CIFAR-100 test images and TinyImageNet validation images for the 27 semantically-overlapping classes (as in Fig.3.3, Fig.3.4).

(b) **New Non-Semantic Test (246 labels; Non-semantic info only):**
Select CIFAR-100 test images from the 73 non-overlapping classes and TinyImageNet validation images from the remaining 173 synsets (as in Section 3.3.2.5, Fig.3.5).



Note: All the models' backbones are trained on 27 semantically overlapping labels of Dataset 04, but in the Non-Semantic test of the Name the Dataset task, the models have to Name the Dataset source of an image from the other 246 Non-Semantically Overlapping Labels

FIGURE 4.6: Non-Semantic test Workflow

2. **Evaluate with the Pretrained Linear Probe in Experiment 1**
Load the frozen ResNet-18 backbone and its 2-way dataset-naming probe (trained in Experiment 1, Section 4.2), then **run inference—without any retraining**—on both test sets.

4.3.4 Preliminaries for Non-Semantic Test Results

Prior to presenting the final results, there are two points to be stated clearly.

1. **Dataset 04 Used Here:**
The non-semantic test was applied to Dataset 03 (Tab. A.1); however, per-label

Name the Dataset accuracies of the pre-trained linear probe clustered tightly above 90 %, and the resulting scatter plot was rather crowded and not very pretty. To address this limitation, the experiment was repeated on Dataset 04 (Tab. A.1), which incorporates an additional downsampling step (Tab.4.4) for TinyImageNet, this pre-processing step of downsampling making per-label Name the Dataset accuracies of the model no longer concentrated around 90% but more spread out.

Preprocessing	**Dataset 03**	**Dataset 04**
TinyImageNet (64×64)	64×64 → 256×256 (Bicubic)	64×64 → 32×32 (Bicubic) → 256×256 (Bicubic)
CIFAR-100 (32×32)	32×32 → 256×256 (Bicubic)	32×32 → 256×256 (Bicubic)

TABLE 4.2: Preprocessing Comparison for Dataset 03 vs. Dataset 04.

2. **Linear Probe Training & Evaluation.**
   The model (backbone + linea probe) was trained exclusively on the 27 semantically overlapping classes. For evaluation, the model is tested on:

   - *New Test* (27 labels; Semantic information + Non-semantic information): *The model (backbone + linea probe) may exploit both learned semantic information of the 27 labels and dataset-wide non-semantic information.*
   - *New Non-Semantic Test* (246 labels; Non-semantic information only): *Since the model is train on 27 semantically overlapping labels, it has no access to 246 non-semantically overlapping labels' semantics and must rely purely on dataset-wide non-semantic information to infer dataset origin of an input image from 246 non-semantically overlapping labels.*

4.3.5 Experiment 2: Results

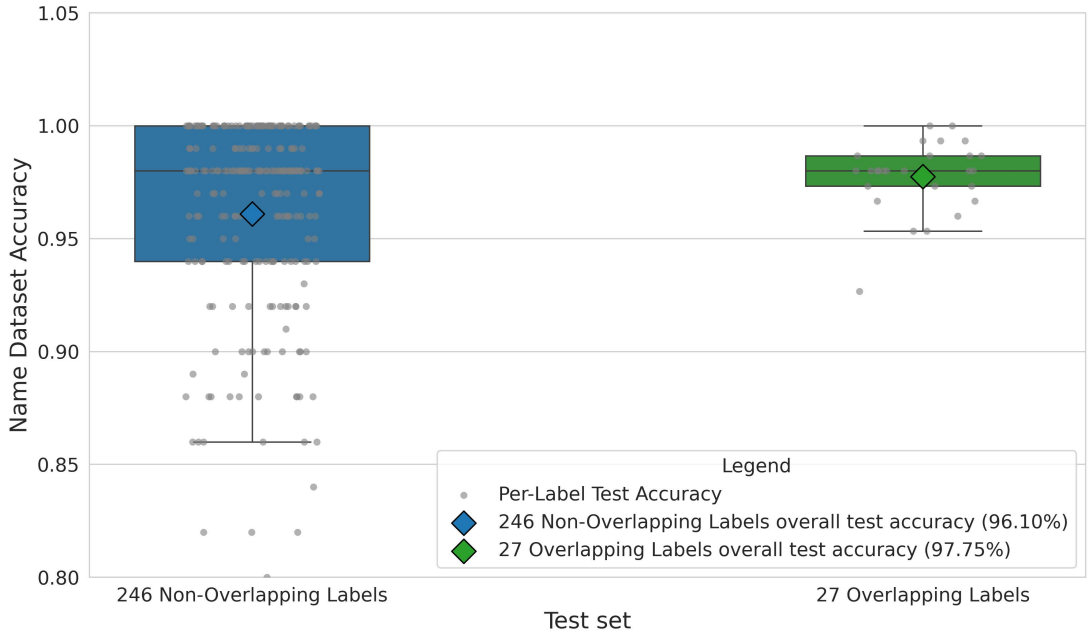


FIGURE 4.7: Non-Semantic Test Per-Label Name Dataset Accuracy of a ResNet-18 Pre-Trained with Protocol A (Section 3.5.2) to Name Label, Linear Probed with Protocol C (Section 3.5.4) to Name Dataset on Dataset 03

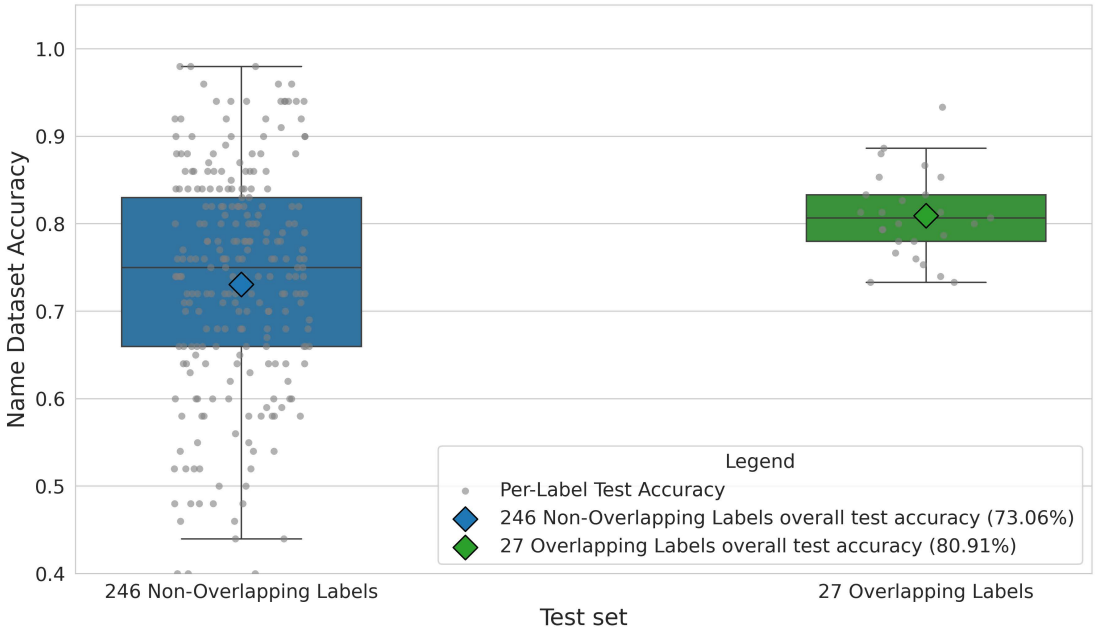


FIGURE 4.8: Non-Semantic Test Per-Label Name Dataset Accuracy of a ResNet-18 Pre-Trained with Protocol A (Section 3.5.2) to Name Label, Linear Probed with Protocol C (Section 3.5.4) to Name Dataset on Dataset 04

4.3.6 Experiment 2: Analysis

The per-label Name the Dataset accuracy distributions in Figure 4.7 and Figure 4.8 illustrate the clustering and spread of per-label Name the Dataset accuracies for both the 246-class non-semantic split and the 27-class semantic + non-semantic split across Datasets 03 and 04.

- **Figure 4.7 (ResNet18 Trained on Dataset 03):**
  - 246-class non-semantically overlapping split: accuracies tightly clustered between 0.80–1.00; median ≈ 0.96 (except outliers).
  - 27-class semantically overlapping split: accuracies between 0.95–1.00 (median ≈ 0.98).
- **Figure 4.8 (ResNet18 Trained on Dataset 04):**
  - 246-class non-semantic split: broader spread $\approx$ 0.4–1.0 (median ≈ 0.73) except outliers).
  - 27-class overlapping split: spread $\approx$ 0.75–0.90 (median ≈ 0.81) except outliers).
- **Non-semantic Performance:** ResNet18 Trained on Dataset 03 achieves 96.10% on entirely unseen classes; ResNet18 Trained on Dataset 04 drops to 73.06% on the same split (–23.04 pp).
- **Semantic Uplift:** Adding the 27 overlapping labels yields only +1.65 pp for ResNet18 Trained on Dataset 03 (96.10 $\rightarrow$ 97.75%), whereas for ResNet18 Trained on Dataset 04 semantics contribute +7.85 pp (73.06 $\rightarrow$ 80.91%).
- **Distributional Differences:**
  The interquartile range (IQR) is defined as the difference between the third quartile (Q_3) and the first quartile (Q_1) of a dataset. It measures the spread of the middle 50% of the data and is more robust to outliers than the total range. For example, consider the dataset $\{1, 2, 3, 4, 5\}$. Here, $Q_1 = 2$ (the median of the lower half) and $Q_3 = 4$ (the median of the upper half), so

  $$\text{IQR} = Q_3 - Q_1 = 4 - 2 = 2,$$

  indicating that the central half of the values spans an interval of length 2.
  - Non-semantic test (246 labels):
    * 1803D IQR ≈ 0.06 (accuracies between $\approx$ 0.94–1.00).

    - 1804D IQR ≈ 0.23 (accuracies between ≈ 0.66–0.83).
  - Semantic + non-semantic test (27 labels):
    - 1803D IQR ≈ 0.01 (accuracies between ≈ 0.965–0.975).
    - 1804D IQR ≈ 0.05 (accuracies between ≈ 0.78–0.83).

4.3.7 Experiment 2: Discussion

- **Dataset bias contains both semantic and non-semantic components; however, the findings indicate that, in this case, non-semantic information is its principal drivers:**
  On Dataset 03 the model scored 96.10% on the 246-class non-semantic split versus 97.75% on the 27-class semantic+non-semantic split (+1.65 pp), and on Dataset 04 it scored 73.06% versus 80.91% (+7.85 pp). These modest semantic gains confirm that non-semantic information are the principal drivers of the model's decisions under conventional preprocessing.
- **The preprocessing pipeline affects the exploitability of semantic and non-semantic information:**
  Non-semantic information are sufficient to separate CIFAR-100 from TinyImageNet, yielding 96% accuracy for ResNet18 trained on Dataset 03; adding a TinyImageNet $\rightarrow 32 \times 32$ downsampling step in Dataset 04 weakens these shortcuts, cuts the Name the Dataset accuracy by 23 pp, and correspondingly increases the relative contribution of semantic information from +1.65 pp to +7.85 pp.

4.4 Experiment 3: Cross Layer Linear Probing

4.4.1 Experiment 3: Objective

The goal of this experiment is to investigate how dataset bias and useful generalizable features are distributed across different layers of deep neural networks, a series of linear probes [Alain and Bengio, 2016] are attached at multiple depths of a backbone (Fig.4.9)—after each residual block in ResNet and after each transformer block in Vision Transformer. Each selected position where a linear probe is inserted is trained with two separate probes: one probe is trained to predict the object labels ('Name the Label') and the other is trained to predict the dataset source ('Name the Dataset'). By charting accuracy as a function of probe index (network depth), this approach pinpoints the layer(s) at which dataset biases become most salient, and it contrasts the emergence of

generalizable features versus dataset bias across the network, providing insights into how deep neural networks evolve during learning.

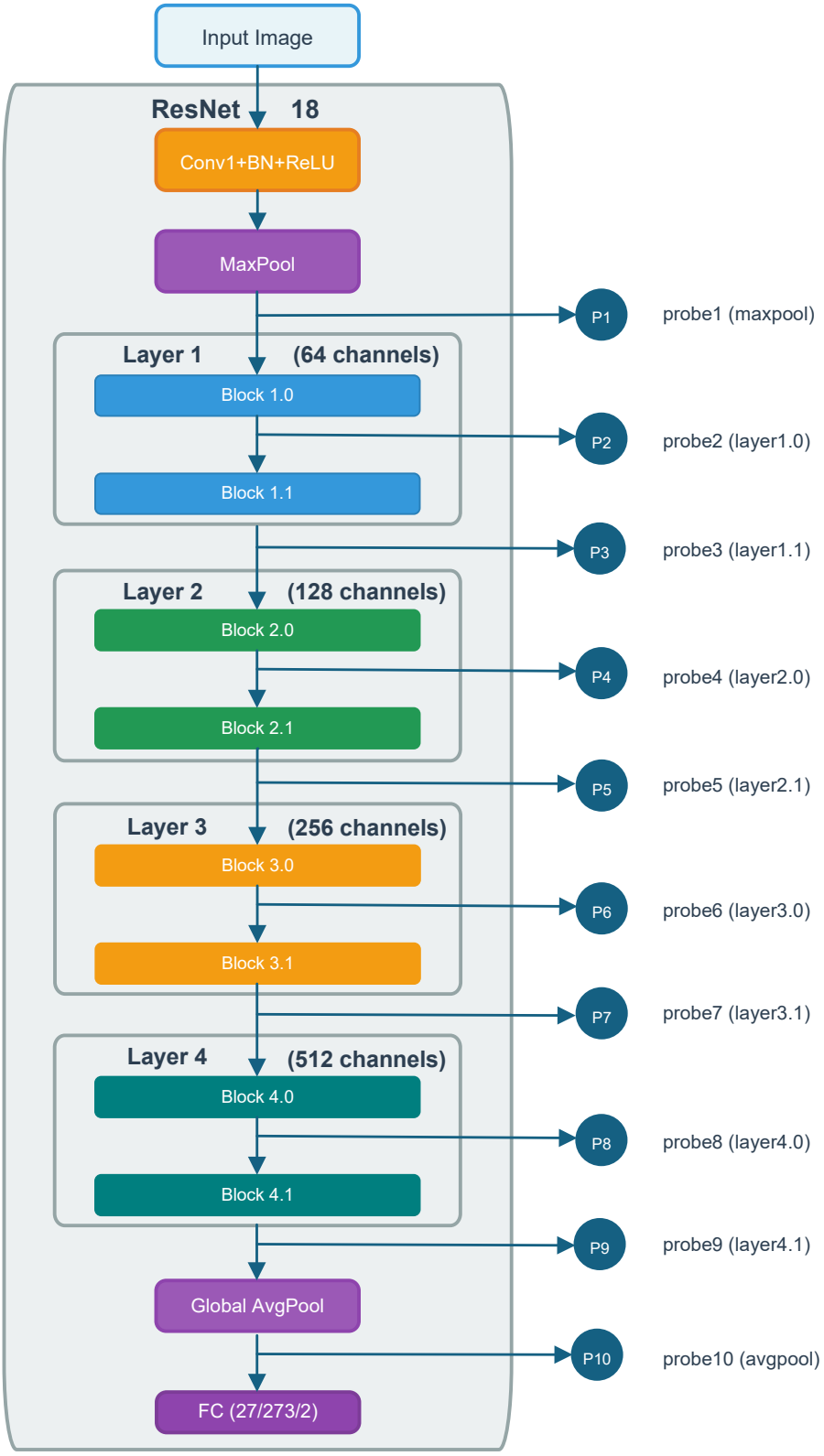


FIGURE 4.9: Illustration for Cross Layer Linear Probing Using Resnet18 as an Example

4.4.2 Experiment 3: Workflow

The experimental workflow for Cross Layer Linear Probing is outlined as follows:

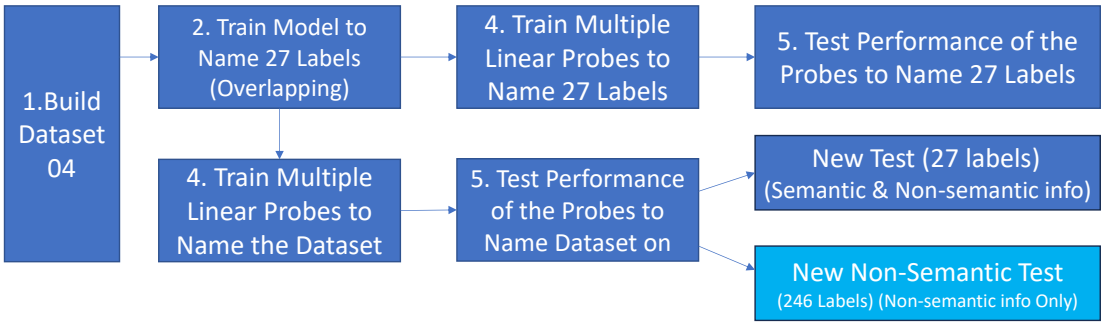


FIGURE 4.10: Cross Layer Linear Probing Workflow

1. **Dataset Construction:**
   Construct Dataset 04 (Tab.A.1) for the overlapping labels between TinyImageNet and CIFAR-100, enabling the model to learn both semantic information of the 27 semantically overlapping labels and dataset-wide non-semantic information.

2. **Train to Name 27 Labels (Overlapping):**
   The same as Experiment 1, use Protocol A (Section 3.5.2) to train ResNet series and use Protocol B (Section 3.5.3) to train Vision Transformer from scratch on the Dataset 04 (27 merged labels) to Name the 27 Labels.

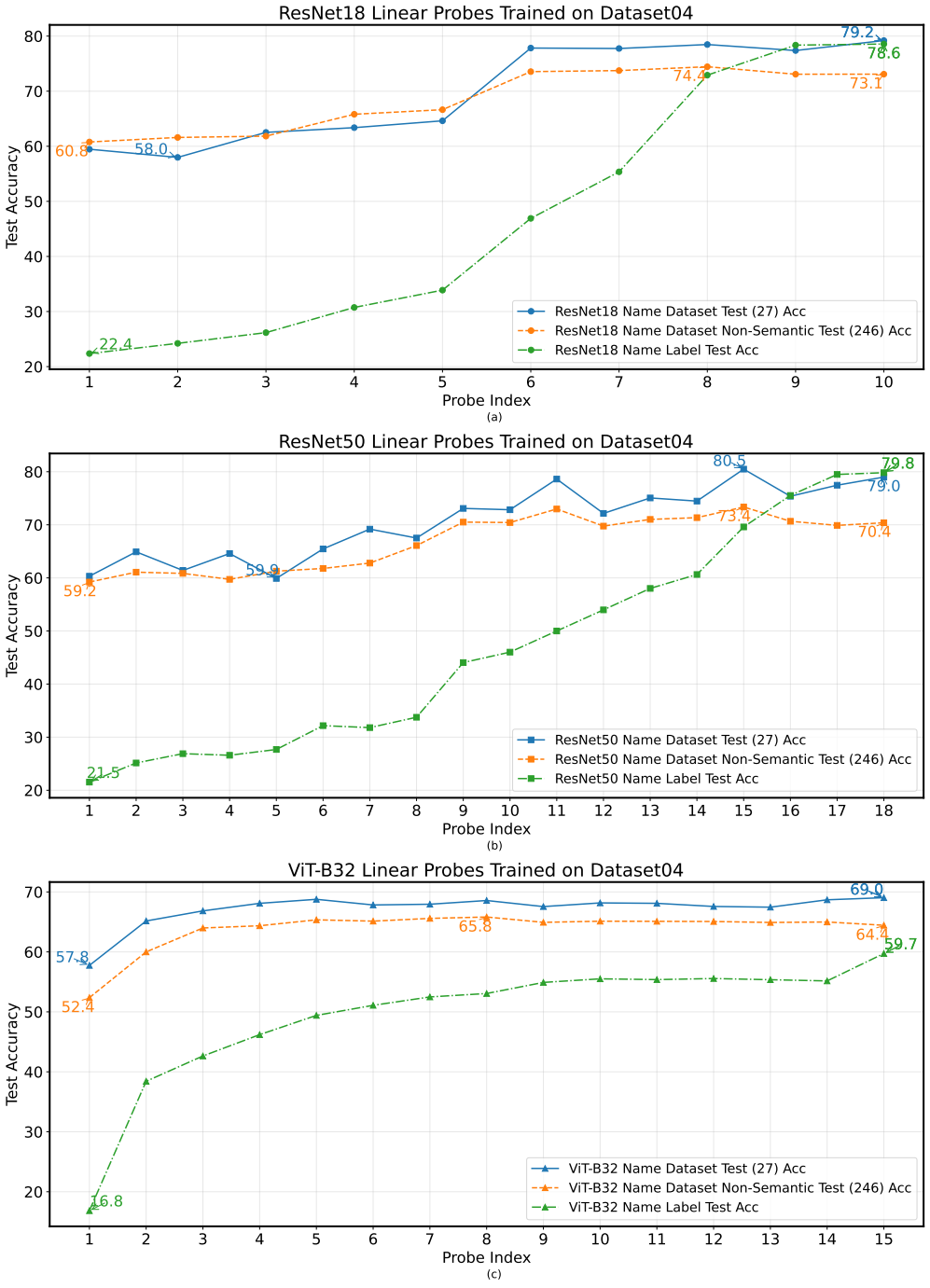
3. **Linear Probe Setup:**
   Insert linear probes at each designated positions (Tab.4.3) of the backbone networks—after every residual block in ResNet and after every transformer block in ViT—and configure two separate linear probes per insertion: one head to predict the object label ("Name the Label") and another to predict the source dataset ("Name the Dataset").

4. **Training and Evaluation:** Train each linear probe independently according to Protocol C (Section 3.5.4); evaluate probe Name 27 Label accuracy on the 27-class overlapping test set. And evaluate probe Name the Dataset accuracy on the 27-label overlapping test set and on the 246-class non-semantic test set to isolate how well each layer encodes non-semantic information when object semantics are unavailable.

Probe ID	**ResNet-18 Location**	**ResNet-50 Location**	**ViT-B/32 Location**
P1	conv1 → bn1 → relu → maxpool Output	conv1 → bn1 → relu → maxpool Output	Patch Embedding (conv proj., flatten + Linear) Output
P2	BasicBlock1 (layer1) Output	Bottleneck (layer1.0) Output	Transformer Block 0 Output
P3	BasicBlock2 (layer1.1) Output	Bottleneck (layer1.1) Output	Transformer Block 1 Output
P4	BasicBlock1 (layer1.2) Output	Bottleneck (layer1.2) Output	Transformer Block 2 Output
P5	BasicBlock2 (layer2.0) Output	Bottleneck (layer2.0) Output	Transformer Block 3 Output
P6	BasicBlock1 (layer2.1) Output	Bottleneck (layer2.1) Output	Transformer Block 4 Output
P7	BasicBlock2 (layer2.2) Output	Bottleneck (layer2.2) Output	Transformer Block 5 Output
P8	BasicBlock1 (layer2.3) Output	Bottleneck (layer2.3) Output	Transformer Block 6 Output
P9	BasicBlock2 (layer3.0) Output	Bottleneck (layer3.0) Output	Transformer Block 7 Output
P10	Global Average Pooling Output	Bottleneck (layer3.1) Output	Transformer Block 8 Output
P11	N/A	Bottleneck (layer3.2) Output	Transformer Block 9 Output
P12	N/A	Bottleneck (layer3.3) Output	Transformer Block 10 Output
P13	N/A	Bottleneck (layer3.4) Output	Transformer Block 11 Output
P14	N/A	Bottleneck (layer3.5) Output	LayerNorm (after last Transformer Block) Output
P15	N/A	Bottleneck (layer4.0) Output	Extract CLS Token Output
P16	N/A	Bottleneck (layer4.1) Output	N/A
P17	N/A	Bottleneck (layer4.2) Output	N/A
P18	N/A	Global Average Pooling Output	N/A

TABLE 4.3: Probing Locations for ResNet-18, ResNet-50 and ViT-B/32

4.4.3 Experiment 3: Results



Note: All the models' backbones are trained on 27 semantically overlapping labels of Dataset 04, but in the Non-Semantic test of the Name the Dataset task, the models have to Name the Dataset source of an image from the other 246 Non-Semantically Overlapping Labels

FIGURE 4.11: Cross Layer Linear Probing Name Dataset/ Label Test Accuracies of ResNet-18 (a)/50(b) & ViT-b-32(c) Pre-Trained with Protocol A (Section 3.5.2) and then Fine-tuned with Protocol C (Section 3.5.4) on Dataset 04

4.4.4 Experiment 3: Analysis

The results (Fig.4.11) from the cross-layer linear probing reveal distinct patterns in how ResNet and Vision Transformer (ViT) models process semantic (object label) and non-semantic (dataset source) information across their layers.

4.4.4.1 ResNet-18 & ResNet-50 (CNNs)

1. **Semantic Feature Learning (*Name the Label; Green Lines*):**
   For both ResNet models, the accuracy of predicting the object label demonstrates a strong positive correlation with network depth. The accuracy starts low in the initial layers (around 22% for both) and increases steadily, with the most significant gains occurring in the mid-to-late layers. For ResNet-18, accuracy jumps from 55% at Probe 7 to 78% at Probe 9. Similarly, for ResNet-50, accuracy rises sharply from 60% at Probe 14 to nearly 80% at Probe 17.

2. **Dataset Bias (*Name the Dataset; Blue, Orange Lines*):**
   In contrast, the ability to identify the source dataset is already high in the earliest layers (around 60% at P1 for both). (This is basically because there are only two datasets to name in total, even though the model simply guess by chance there would be an accuracy of 50%.) The Name the Dateset test (blue) and Non-Semantic test (orange) accuracies shows a strong positive correlation. And both of them fluctuates through the network, tends to approaching their peaks much earlier than the Name the Label accuracy.

4.4.4.2 ViT-B/32 (Transformer)

1. **Semantic Feature Learning (*Name the Label; Green Lines*):**
   The ViT model shows a different dynamic. After a low initial accuracy in the patch embedding layer (P1: 16.8%), there is a sharp increase after the first transformer block (P2: 38.4%). Following this jump, the accuracy continues to rise, but more gradually and steadily through the remaining blocks, reaching a peak of 59.7% at the final probe.

2. **Dataset Bias (*Name the Dataset; Blue, Orange Lines*):**
   Similar to ResNet, the Name the Dateset test (blue) and Non- Semantic test (orange) accuracies shows a strong positive correlation. Specifically the Name the Dataset test accuracy starts at 57.8% at P1—well above the 50% chance level—rises quickly to 68.3% by P5, and then slightly fluctuates around 67–69% through P15, peaking at 69.0%. The Name the Dataset Non-Semantic test accuracy begins at

52.3% at P1, climbs to approximately 65% by P5, and thereafter fluctuates modestly around 64–65% across all subsequent transformer blocks.

4.4.5 Experiment 3: Discussion

- **The Same as Experiment 2 Interpretation (Section 4.3.7), dataset bias is primarily encoded in Non-Semantic information:**
  In both ResNet and ViT architectures, the probes in the early layers were significantly more effective at Naming the Dataset than Naming the Label. This is because superficial statistics (e.g., resolution, compression artifacts, color profiles) that distinguish datasets are readily available in the initial layers.

- **Fluctuations in the Name the Dataset accuracy curves on both the test and non-semantic test sets are driven by two factors:**

  1. *The increase arises as the model uncovers mappings between semantic or non-semantic information (but primarily non-semantic features) and dataset bias, thereby boosting its ability to recognize which dataset a sample comes from.*
  2. *The decrease occurs because, as the network depth increases, non-semantic information is progressively transformed into higher-level semantic information, which disrupts the original mappings between non-semantic information and dataset bias.*

  To conclude, the fluctuations in the Name the Dataset accuracy curves are, in essence, the external manifestations of the model's integration and transformation of the infortion of an input image.

- **It seems that deeper ResNets can better accomplish the transformation from non-semantic to semantic information and are less sensitive to dataset bias:**
  It is observed that the two Name the Dataset accuracies of the final probe in ResNet50 (79.0%; 70.3%) is lower than that of ResNet18 (79.2%; 73.1%).*Due to computing power and time constraints, this paper has not yet verified whether this is also true for ViT and other model architectures.*

- **The ResNet blocks may act as distinct feature-processing stages, whereas transformer layers refine representations more continuously:**

  1. **ResNets exhibit hierarchical refinement.**
     ResNets show fairly discrete "phase changes" at certain residual blocks, where both Name the Dataset test accuracy and Name the Label test accuracy

exhibit step-wise jumps. They learn generic, low-level features rich in early layers, then progressively abstract these into complex, semantic features. This "purifies" the representation and some non-semantic information seems to be gradually integrated and transformed to be semantic information as the network layers gets deeper, but every layer, even the last one, still retains both semantic and non-semantic information simultaneously.

2. **ViTs maintain parallel representations of semantic and non-semantic information.**
   Vision Transformers distribute the learning more evenly, with smoother gains in both tasks across successive transformer blocks.The initial patch embedding captures dataset bias, and global self-attention builds semantic understanding without discarding the original non-semantic information. As a result, every layer retains both semantic and non-semantic information simultaneously, and the Name the Dataset curves of ViT did not undergo a drastic fluctuation like ResNet, but rather fluctuate more gently.

4.5 Experiment 4: Two Full Datasets

4.5.1 Experiment 4: Objective

The primary aim of this experiment is to evaluate whether pretraining a ResNet-18 backbone using Protocol A (Section 3.5.2) on a richly annotated, 273-label dataset (Dataset 06; Section 3.3.3, Tab.A.1) yields superior accuracy when subsequently linear probing, using Protocol C (Section 3.5.4), to Name the Dataset, as compared to a baseline model pretrained, using Protocol A (Section 3.5.2), on only the 27 semantically overlapping labels (Dataset 04; Section 3.3.2, Tab.A.1).

Concretely, it is compared:

- **Dataset04 (27 training labels):** A baseline setup using only the 27 labels shared between CIFAR-100 and TinyImageNet.
- **Dataset06 (273 training labels):** An expanded setup in which all 273 labels (27 overlapping + 73 CIFAR-only + 173 Tiny-only) are used during pretraining and validation.

By holding all factors constant except the breadth of semantic labels during pretraining and linear probing, this investigation isolates the impact of label diversity (or the number of input samples) on Name the Dataset performance.

Besides, in the evaluation stage, the Name the Dataset confusion matrix (Fig.4.4) in Experiment 1 (ResNet18 pretraining with Protocol A and linear probing with Protocol C on Dataset03) is also introduced, with the object to compare Name the Dataset performance of a model with different pre-processing pipelines (Dataset 03/04) and with different breadth (Dataset 04/06) of semantic labels (or the number of input samples).

4.5.2 Experiment 4: Workflow

The end-to-end procedure for comparing pretraining on 27 vs. 273 labels before linear probing "Name the Dataset" is shown in Figure 4.12. It parallels the steps in Experiment 1 but now uses two distinct pretraining datasets.

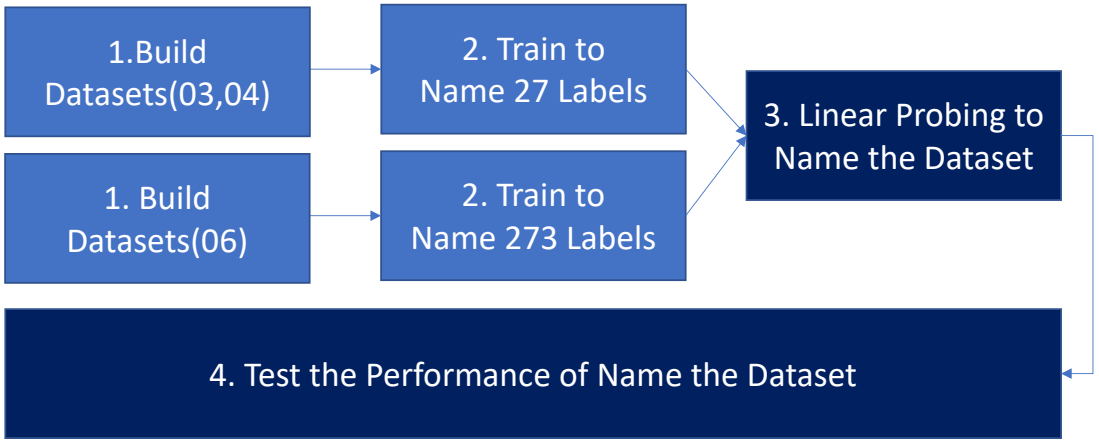


FIGURE 4.12: Two Full Datasets Workflow

1. **Construct Pretraining Datasets**
   - *Dataset 04 (27 labels):* Assemble the semantically overlapping label set as in Section 3.3.2, merging TinyImageNet & CIFAR-100 into 27 shared classes.
   - *Dataset 06 (273 labels):* Assemble the full union of 27 shared + 73 CIFAR-only + 173 TinyImageNet-only labels as in Section 3.3.3.
2. **Train Semantic Classifiers (Protocol A)**
   - *Train on 27 labels:* From scratch, train a ResNet-18 on Dataset 04 to classify into the 27 merged classes (Protocol A; Section 3.5.2).
   - *Train on 273 labels:* Likewise, from scratch, train a separate ResNet-18 on Dataset 06 to classify into all 273 classes (Protocol A; Section 3.5.2).
3. **Linear Probe to Name the Dataset (Protocol C)** For each pretrained backbone (27-label and 273-label), freeze all convolutional layers and replace the final

fully-connected layer with a 2-way linear classifier (CIFAR vs. Tiny). Train only this linear probe (2-way linear classifier) on the training split to Name the Dataset of each image (Protocol C; Section 3.5.4).

4. **Evaluate Dataset Naming Performance** Test each linear probe on its corresponding 27-label or 273-label test split. Record Top-1 Test accuracy for Name the Dataset and compare whether the richer 273-label pretraining yields Name the Dataset Performance.

4.5.3 Experiment 4: Cross Datasets Confusion Matrices Results

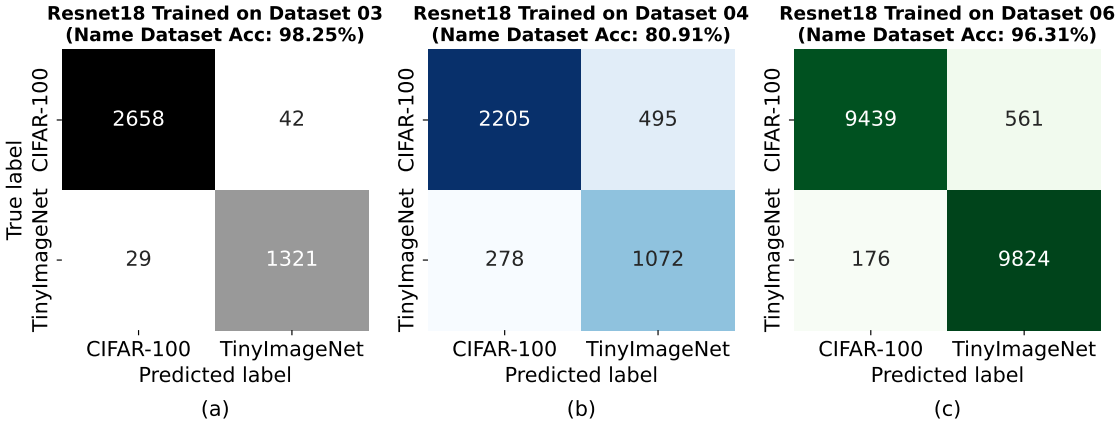


FIGURE 4.13: Comparison of Name Dataset Confusion Matrices of ResNet18 Pretrained (Protocal A) and Linear Probed (Protocal C) on Dataset 03/04/06 Respectively

Preprocessing	**Dataset 03**	**Dataset 04/06**
TinyImageNet (64×64)	64×64 → 256×256 (Bicubic)	64×64 → 32×32 (Bicubic) → 256×256 (Bicubic)
CIFAR-100 (32×32)	32×32 → 256×256 (Bicubic)	32×32 → 256×256 (Bicubic)

TABLE 4.4: Preprocessing Comparison for Dataset 03 vs. Dataset 04/06.

4.5.4 Experiment 4: Cross Datasets Confusion Matrices Analysis

Each matrix (Fig.4.13) reveals the model's performance in distinguishing between CIFAR-100 and TinyImageNet images after pretraining and linear probing.

- **Model on Dataset 03 (Control):**
  - Overall accuracy: 98.25%

  - CIFAR-100: 2658/2687 correct, 29 misclassified as TinyImageNet.
  - TinyImageNet: 1321/1363 correct, 42 misclassified as CIFAR-100.

- **Model on Dataset 04 (Baseline):**
  - Overall accuracy: 80.91%
  - CIFAR-100: 2205 correct, 278 misclassified.
  - TinyImageNet: 1072 correct, 495 misclassified as CIFAR-100.

- **Model on Dataset 06 (Experimental):**
  - Overall accuracy: 96.31%
  - CIFAR-100: 9439 correct, 176 misclassified.
  - TinyImageNet: 9824 correct, 561 misclassified as CIFAR-100.

In summary, the most striking differences are *the sharp drop in Name the Dataset accuracy from Dataset 03 to Dataset 04 and the subsequent significant rebound from Dataset 04 to Dataset 06.*

4.5.5 Experiment 4: Cross Datasets Confusion Matrices Discussion

As the number of training samples increases, the gain of alleviating dataset bias by simply improving preprocessing methods will gradually weaken.

- **The Impact of Preprocessing Artifacts (Dataset 03 vs. Dataset 04):** The sharp performance decline from 98.25% on Dataset 03 to 80.91% on Dataset 04 is caused by the change in preprocessing. In Dataset 03, TinyImageNet was upsampled by a factor of 4; CIFAR was upsampled by a factor of 8, whereas in Dataset 04 both CIFAR and TinyImageNet were upsampled by a factor of 8, making the upsampling artifacts harder to leverage for Name the Dataset.

- **The Impact of Semantic Diversity (Dataset 04 vs. Dataset 06):** The significant accuracy rebound from 80.91% on Dataset 04 to 96.31% on Dataset 06 directly shows the effect of richer training samples. Although both models used the same "fairer" preprocessing that lacked obvious artifacts to Name the Dataset, the model trained on 273 labels Dataset 06 learned a far more robust and generalizable set of features to Name the Dataset *with richer training samples.*

4.5.6 Experiment 4: Dataset 06 Name the Dataset Results

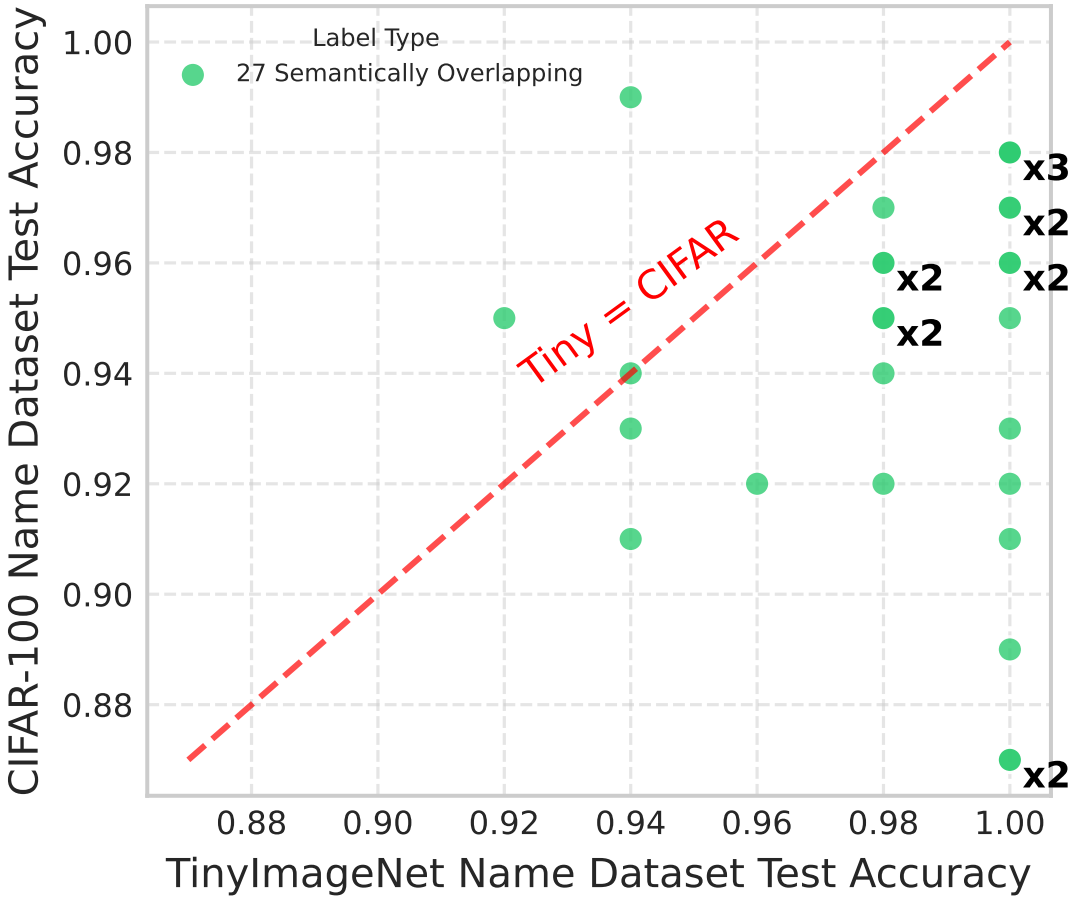


FIGURE 4.14: Two Full Datasets *TinyImageNet vs. CIFAR-100 Name Dataset Test Accuracies for 27 Semantically Overlapping Labels*: Scatter plot with duplicate points annotated. ResNet-18 pre-trained with Protocol A (Section 3.5.2) and linear-probed with Protocol C (Section 3.5.4) on Dataset 06.

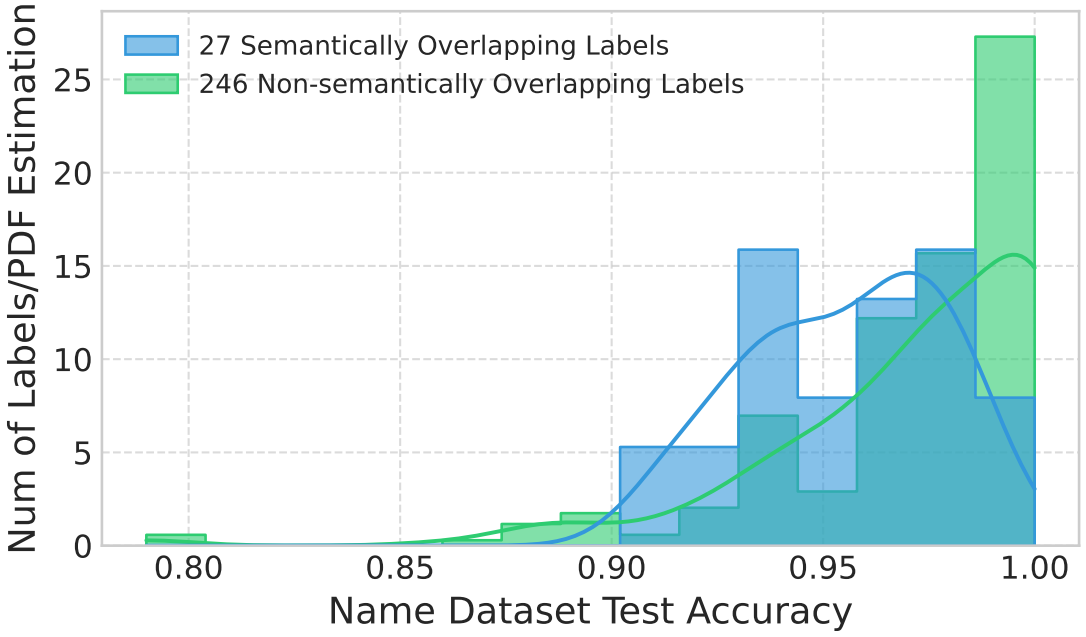


FIGURE 4.15: Two Full Datasets Per-Label Name Dataset Test Accuracies Histogram & Estimated Probability Density Distribution Using Kernel Density Estimation(KDE) of the ResNet-18 Pre-Trained with Protocol A (Section 3.5.2) and then Linear Probed with Protocol C on Dataset 06 (Section 3.5.4)

Label Type	Count	Mean Acc	Min Acc	Max Acc	Median Acc
Non-semantically Overlapping	246	0.9713	0.7900	1.0000	0.9800
Semantically Overlapping	27	0.9551	0.9133	0.9867	0.9600

TABLE 4.5: Non-Semantically vs. Semantically Overlapping Labels Per-Label Name the Dataset Test Accuracy Statistics for Figure 4.15

4.5.7 Experiment 4: Dataset 06 Name the Dataset Analysis

The per-label Name the Dataset performance on Dataset 06 is summarized in Figure 4.14 (TinyImageNet vs. CIFAR-100 scatter of the 27 semantically overlapping labels) and Figure 4.15 (histogram & KDE of all 273 labels), with key statistics in Table 4.5.

- **Scatter of overlapping labels (Fig.,4.14):** Most of the 27 semantically overlapping labels lie in the upper-right quadrant, indicating high accuracy on both TinyImageNet and CIFAR-100, but **for most of the 27 semantically overlapping labels Name the Dataset test accuracy for TinyImageNet is higher than that of CIFAR-100.** Duplicate markers (e.g. "×2", "×3") show that several classes achieved identical performance across domains.

- **Accuracy distribution (Fig.,4.15):** The KDE curves reveal that the 246 non-semantically overlapping labels (green) cluster more tightly toward perfect accuracy, peaking near 99–100%, whereas the 27 overlapping labels (blue) form a slightly broader peak around 95–97%.
- **Summary statistics (Table 4.5):** The Name the Dataset Mean Accuracy and Median Accuracy for 246 Non-semantically Overlapping Labels are higher than that for 27 Semantically Overlapping Labels.

4.5.8 Experiment 4: Dataset 06 Name the Dataset Discussion

- **Different Preprocessing methods will introduce different Non-Semantic information (such as interpolation artifacts), which may be integrated into the dataset bias, thereby exploited by the model to Name the Dataset.**

  In Figure 4.14, systematic advantage for TinyImageNet is observed, as the majority of points fall below the parity line, indicating that per-label accuracy is often higher on TinyImageNet than on CIFAR-100; this finding aligns with the expectation that downsampling of TinyImageNet during preprocessing (Tab.4.4) produces distinguishable artifacts, while CIFAR does not go through such downsampling process. A small subset of labels (Fig.4.14) deviates markedly from this parity line, one class achieving 100% on TinyImageNet but only 87% on CIFAR-100—suggesting strong dataset biases in image style or annotation for TinyImageNet.

- **It is much easier for the model to Name the Dataset for 243 Non-semantically Overlapping Labels than 27 Semantically Overlapping Labels.**

  1. **Distinct "flag" effect of non-overlapping labels**:
     The 246 non-semantically overlapping labels achieve higher mean and median accuracies because—during training of the 273-way classifier on dataset 06—they are already well separated in the high-dimensional feature space before the final fully-connected layer. When fine-tuning that last dataset-ID layer, the model can exploit each of these 243 Non-semantically Overlapping labels as a unique "flag" for its single associated dataset, pushing their per-label accuracy toward 100%.
  2. **Shared labels are much harder to name the dataset**
     By contrast, the 27 semantically overlapping labels occur in multiple datasets and thus lack a one-to-one dataset correspondence. They cannot serve as

unique dataset identifiers, leading to slightly lower average accuracy and greater variability in the Name-the-Dataset task.

Chapter 5

Maths Modeling for Dataset Bias Generation Process

5.1 Sectional Introduction

In this chapter, a unified mathematical framework for understanding and quantifying dataset bias in Computer Vison is developed. It is beginned by formulating the hypothesis that every collected image dataset represents an incomplete, biased sample of the real visual world, and that models trained on such datasets necessarily internalize these biases. To make this intuition precise, the observed variables—semantic labels, dataset origins, and raw image samples—are defined and distinguished from the latent factors of interest: generalizable features and dataset bias shortcuts. A probabilistic generative process is then described in which generalizable features and dataset bias shortcuts jointly give rise to each image, illustrating how this entanglement leads models to latch onto spurious, dataset bias shortcuts. The resulting generalization failures on out-of-distribution data are analyzed, and an idealized regularizer that would suppress reliance on dataset bias shortcuts is proposed—highlighting the conceptual challenges involved in implementing such a regularizer in practice. This formalism lays the groundwork for the detailed definitions, theoretical results, and empirical analyses that follow.

5.2 Hypothesis: Inescapable Dataset Bias and Its Inevitable Internalization

1. **Dataset Bias: Where Does It Come From?**
   In Computer Vision, every dataset is inevitably an incomplete sampled version of the entire real visual world.

   (a) The choices made during sampling—sensor type, lighting, camera angle, post-processing, etc.—impose distributional discrepancies between the sampled images and the real vision world.

   (b) These sampling-induced distributional discrepancies form the dataset's inherent bias, which manifests as distributional discrepancies:

   - *Between each dataset and the real world*, because a dataset can only capture a sampled subset of the visual world, not its entirety.
   - *Between different datasets themselves* produced with different sampling protocols.

2. **Dataset Bias Acquisition: How does a model learn it?**
   A model trained on a biased dataset will inevitably learn dataset bias that that boost performance on the training set but impair the model's generalization performance on out-of-distribution test sets.

   (a) The training dataset represents the model's entire visual world [Torralba and Efros, 2011].

   (b) Without any external reference, the model lacks prior knowledge to distinguish:

   - *Generalizable features* — patterns valid across datasets and the real visual world.
   - *Dataset bias shortcuts*[1]— patterns predictive only within the training data.

5.3 Key Variables Defined in Dataset Bias Analysis

To formalize our understanding, the variables are defined at first, distinguishing between those can be directly observed during training and those that are latent (hidden).

[1]The term "dataset-bias shortcuts" is used to denote useless *Non-Semantic Information* (e.g.,sensor type, lighting, artifacts, etc.) that exist only in the training set but fail to generalize to other datasets or real-world scenarios. Remember that there are also some useful *Semantic Information* contained in dataset bias (Section 4.3.7).

5.3.1 Observed Variables

- **Semantic Label ($\boldsymbol{y} \in \boldsymbol{Y}$)**: This is the true classification label for the image (e.g., "dog," "car") denoted as **one-hot vector** format, which the model is trained to predict. Its real semantic meaning is consistent across all datasets.

- **Dataset Source ($\boldsymbol{d} \in \boldsymbol{D}$):** This is an identifier for the data's origin (e.g., CIFAR, Tiny ImageNet) denoted as **one-hot vector** format, indicating different choices made during sampling—sensor type, lighting, camera angle, post-processing, etc.—for different data source.

- **Image Sample ($\boldsymbol{x} \in \boldsymbol{X}$):** This is the input data seen by the model (e.g. an image from CIFAR). Usually write $X = \mathbb{R}^D$, where D is the dimension of the observed features (e.g. the number of pixels times the number of channels). Crucially, $\boldsymbol{x}$ is the signal generated by the mixture of latent features $\boldsymbol{g}$ and $\boldsymbol{b}$, which can be expressed as:

$$\boldsymbol{x} = \phi(\boldsymbol{g}, \boldsymbol{b}),$$

where

$$\boldsymbol{g} \in \mathbb{R}^{\infty}, \quad \boldsymbol{b} \in \mathbb{R}^{\infty}$$

are both infinite-dimensional column vectors, and $\phi : \mathbb{R}^{\infty} \times \mathbb{R}^{\infty} \to \mathbb{R}^D$ is the (unknown) generative mixing function.

5.3.2 Latent Variables

- Generalizable Features ($\boldsymbol{g} \in \mathcal{G}$): These are the intrinsic, transferable visual patterns, ***including semantic information[2] contained in dataset bias (Section 4.3.7)***, that are causally related to the semantic label $\boldsymbol{y}$, such as an object's fundamental geometry and structure.

  - *Unobservable:* Generalizable Feature $\boldsymbol{g}$ is a hidden component of the image $\boldsymbol{x}$ and is not explicitly observed during training.
  - *Invariance:* The distribution of these features, given a label, is independent of the dataset source:

$$P(\boldsymbol{g} \mid \boldsymbol{y}, \boldsymbol{d}) \;=\; P(\boldsymbol{g} \mid \boldsymbol{y}), \quad \forall d \in D.$$

  - *Essence of the Task:* These features form the true basis for classification, meaning the relationship $P(\boldsymbol{y} \mid \boldsymbol{g})$ is robust and generalizable.

[2]Note that dataset bias also contain some useful generalizable semantic information (Section 4.3.7, [Liu and He, 2025].

- Dataset Bias Shortcuts[3]($\boldsymbol{b} \in \mathscr{B}$): This is ***Non-Semantic Information contained in dataset bias*** that is predictive within a specific dataset, like the characteristic low-resolution noise in CIFAR images.

  - *Unobservable:* Dataset bias shortcuts $\boldsymbol{b}$ is a hidden component of the image $\boldsymbol{x}$ and is not explicitly observed during training.
  - *Causally Irrelevant:* By itself, the bias contains no information about the true label:
  $$P(\boldsymbol{y} \mid \boldsymbol{b}) = P(\boldsymbol{y}).$$
  - *Domain-Dependent:* The distribution of the dataset bias shortcuts are tied to the dataset source d and denoted as $\boldsymbol{b}_d$ [4], and different dataset sources contain different dataset bias shortcuts:
  $$\boldsymbol{b}_{d1} = P(\boldsymbol{b} \mid \boldsymbol{d}_1) \neq P(\boldsymbol{b} \mid \boldsymbol{d}_2) = \boldsymbol{b}_{d2}, \quad \text{namely} \quad \boldsymbol{b}_{d_1} \neq \boldsymbol{b}_{d_2}.$$

5.4 The Probabilistic Dataset Bias Shortcuts Generation Process

The model's challenge stems from its *observational limits*. It only sees pairs of an input $\boldsymbol{x}$ and its true label $\boldsymbol{y}$, from which it must learn the function:

$$\begin{aligned} \hat{\boldsymbol{y}} &= \text{softmax}(\boldsymbol{z}) \\ &= \text{softmax}\big(f(\boldsymbol{x})\big) \\ &= \text{softmax}\big(f\big(\phi(\boldsymbol{g}, \boldsymbol{b})\big)\big) \end{aligned} \tag{5.4.1}$$

where $\hat{\boldsymbol{y}}$ is the model's predicted one-hot label vector after softmax, $f(\cdot)$ is a neural network mapping the input $\boldsymbol{x}$ to pre-softmax logits $\boldsymbol{z}$, and $\boldsymbol{x}$ is formed by a generative process ϕ that combines generalizable task-relevant (causal) features $\boldsymbol{g}$ with dataset bias shortcuts (spurious) features $\boldsymbol{b}$.

Crucially, the model can never directly observe or disentangle $\boldsymbol{g}$ and $\boldsymbol{b}$ during training. This inability results in the learning of dataset bias shortcuts that appear predictive on

[3]The term "dataset-bias shortcuts" is used to denote useless *Non-Semantic Information* (e.g.,sensor type, lighting, artifacts, etc.) that exist only in the training set but fail to generalize to other datasets or real-world scenarios. Remember that there are also some useful generalizable *Semantic Information* contained in dataset bias (Section 4.3.7, [Liu and He, 2025]).

[4]When $\boldsymbol{b}$ appears without a subscript, it denotes the entire collection of dataset bias shortcuts, encompassing dataset bias shortcuts from any dataset in arbitrary combinations. In contrast, the notation $\boldsymbol{b}_d$ refers specifically to the dataset bias shortcuts induced by a particular dataset d, isolating the subset of dataset bias shortcuts that arise uniquely from that dataset.

the training distribution but fail on out-of-distribution (O.O.D.) data [Geirhos et al., 2020, Ilyas et al., 2019].

The model's dataset bias shortcuts induced failure to generalize can be decomposed into three phases:

1. **Spurious Correlation:**
   Although the shortcut bias $\boldsymbol{b}_d$ is not causally related to the true label $\hat{\boldsymbol{y}}$, the sampling procedure of a specific dataset $\boldsymbol{d}$ can make the bias *appear* as informative as the genuinely predictive features $\boldsymbol{g}$:

$$\underbrace{P(\hat{\boldsymbol{y}} \mid \boldsymbol{b}_d)}_{\text{illusory on dataset } d} \approx \underbrace{P(\boldsymbol{y} \mid \boldsymbol{g})}_{\text{true task signal}} \quad \text{(within the biased dataset).}$$

   Outside this narrow sampling context the dataset bias shortcut collapses, revealing the underlying mismatch:

$$P(\hat{\boldsymbol{y}} \mid \boldsymbol{b}_d) \neq P(\hat{\boldsymbol{y}} \mid \boldsymbol{g}).$$

   Any dataset bias that happens to occur too frequently in the dataset will make it appear to be related to the label—even if there is no causal basis for this association.

2. **Model's Learning Strategy:**
   Faced with inputs $\boldsymbol{x} = \phi(\boldsymbol{g}, \boldsymbol{b}_d)$ that entangle generalizable features and dataset bias shortcuts, a flexible learner will typically fit a prediction rule of the form

$$\hat{\boldsymbol{y}} = softmax(f_d(\boldsymbol{x}))^{5} = softmax(f_d(\boldsymbol{\phi}(\boldsymbol{g}, \boldsymbol{b}_d))) = softmax(\varphi(h(\boldsymbol{g}), k(\boldsymbol{b}_d))),$$

   where

   - $h(\boldsymbol{g})$ is a function learned by the model focusing on the generalizable, task-relevant features $\boldsymbol{g}$,
   - $k(\boldsymbol{b_d})$ is a function learned by the model that exploits the dataset bias shortcut b_d,
   - the coupling function φ allows arbitrary *non-linear, non-separable* interactions between $h(\boldsymbol{g})$ and $k(\boldsymbol{b_d})$,
   - the relative influence of $\boldsymbol{g}$ and $\boldsymbol{b}$ or $h(\boldsymbol{g})$ and $k(\boldsymbol{b_d})$ is therefore data-driven and cannot be reduced to a simple linear trade-off[6].

[5] When f appears with a subscript, as in $f_d(\boldsymbol{x})$, it denotes the predictive function of a model trained specifically on dataset $\boldsymbol{d}$. In contrast, $f(\boldsymbol{x})$ without a subscript refers to a general predictive mapping that could represent a model trained on any dataset or any combination of datasets.

[6] Although there is a paper that defines $\boldsymbol{g}$ and $\boldsymbol{b}$ as a simple linear relationship [Khosla et al., 2012]

3. **Generalization Failure.** When the learned model is evaluated on a different dataset $d' \neq d$, the dataset bias shortcut statistics typically shift:

$$P(\boldsymbol{b}_{d'} \mid d') \;\neq\; P(\boldsymbol{b}_d \mid d) \quad\Longrightarrow\quad P(\hat{\boldsymbol{y}} \mid \boldsymbol{b}_{d'}) \;\neq\; P(\hat{\boldsymbol{y}} \mid \boldsymbol{b}_d) \;\neq\; P(\boldsymbol{y} \mid \boldsymbol{g}).$$

Because ϕ has encoded brittle dependencies on $k(\boldsymbol{b})$, these distributional changes invalidate the learned associations, leading to a collapse in predictive performance:

$$\hat{\boldsymbol{y}} = softmax(f_d(\boldsymbol{x})) \;\neq\; \hat{\boldsymbol{y}}' \;=\; softmax(f_{d'}(\boldsymbol{x})) \;\neq\; \boldsymbol{y}$$

5.5 The Model's Dilemma: Generalizable Features vs. Dataset Bias Shortcuts

From the model's perspective, its entire "visual universe" is the training dataset. It has no built-in way to tell which patterns will hold across new environments and which are merely dataset bias shortcuts. Also, in standard training, the loss function is defined solely in terms of model outputs and ground-truth labels. Concretely, the standard loss function $\mathscr{L}_{\text{std}}$ is given by

$$\mathscr{L}_{\text{std}} = \mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})\sim P_d}\big[\ell\big(\hat{\boldsymbol{y}}, \boldsymbol{y}\big)\big],$$

where

- $\mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})\sim P_d}[\cdot]$ denotes the expectation over the dataset distribution P_d,
- $\boldsymbol{x} = \phi(\boldsymbol{g}, \boldsymbol{b}_d)$ denotes the observed image composed of generalizable features $\boldsymbol{g}$ and dataset bias shortcuts $\boldsymbol{b}_d$ of dataset $\boldsymbol{d}$,
- $\boldsymbol{y}$ is the associated true label of an input image,
- $\hat{\boldsymbol{y}} = softmax(f_d(\boldsymbol{x}))$ is the prediction of the model trained on dataset $\boldsymbol{d}$ after softmax,
- $\ell(\cdot, \cdot)$ is a loss function such as cross-entropy.

Because $\mathscr{L}_{\text{std}}$ depends only on the pair $(\hat{\boldsymbol{y}}, \boldsymbol{y})$, irrespective of whether the model relies on generalizable features $\boldsymbol{g}$ or on dataset bias shortcuts $\boldsymbol{b}_d$, as long as it produces the same outputs $\hat{\boldsymbol{y}} = softmax(f_d(\boldsymbol{x}))$, it will incur the same loss. Consequently, reliance on $h(\boldsymbol{g})$ (generalizable features) or on $k(\boldsymbol{b}_d)$ (the dataset bias shortcuts) cannot be distinguished by the training criterion. Absent additional supervisory signals, architectural constraints, or biased datasets, the model is free to pursue whichever combination of features yields

minimal loss on the training set. This indifference to the origin of predictive cues lies at the heart of poor cross-domain generalization: when deployed on a new source d', the dataset-specific bias associated with $\boldsymbol{b}$ typically does not hold, resulting in a marked decline in performance.

5.5.1 An Idealized Regularizer: Alleviating the Model's Dilemma

An ideal loss would include a penalty for dependence on the bias component $\boldsymbol{b}$. One possible formulation [Kim et al., 2019] is

$$\mathscr{L}_{\text{ideal}} = \mathscr{L}_{\text{std}} + \lambda\, I\big(f(\boldsymbol{x});\, \boldsymbol{b}\big),$$

in which $I\big(f(\boldsymbol{x});\, \boldsymbol{b}\big)$ measures the mutual information between the model's outputs and the latent dataset bias shortcuts $\boldsymbol{b}$, and $\lambda > 0$ governs the trade-off between task performance and strength of bias suppressing. Under this augmented objective, the learning process is compelled to favor information conveyed by $\boldsymbol{g}$ (which is statistically independent of $\boldsymbol{b}$) and to suppress reliance on dataset bias shortcuts.

5.5.2 An Idealized Regularizer: Intractability to Realize

In practice, both $\boldsymbol{g}$ and $\boldsymbol{b}$ remain latent. The training data for any image i in the training set (n images in total) comprise only triples

$$\{\, (\boldsymbol{x}_i, \boldsymbol{y}_i, \boldsymbol{d}_i) \,\}_{i=1}^{n} \sim P_d(x, y),$$

with no direct access to the latent variables $\boldsymbol{g}$ or $\boldsymbol{b}$. As a result:

- The term $I\big(f(\boldsymbol{x});\, \boldsymbol{b}\big)$ can hardly be computed, since $\boldsymbol{b}$ is latent and difficult to measure.
- Proxy signals (e.g., domain labels d, camera or sensor IDs) may serve as rough surrogates for $\boldsymbol{b}$, but such proxies are inherently coarse and may capture only a subset of the true dataset bias shortcuts.

- The term $I\big(f(\mathbf{x});\, \mathbf{b}\big)$ can hardly be computed, since $\mathbf{b}$ is latent and difficult to measure.
- Proxy signals may serve as rough surrogates for $\mathbf{b}$, but such proxies are inherently coarse and may capture only a subset of the true dataset bias shortcuts. Common examples include:

- **Domain labels (e.g., "web images" vs. "user-generated photos")**: This coarse categorization reflects differences in data sources—professional stock images often have clean backgrounds and consistent composition, whereas casual smartphone snapshots may include cluttered scenes and variable framing—but it cannot distinguish finer biases like indoor vs. outdoor settings or artistic style [Li et al., 2017].
- **Timestamps (e.g., "daytime" vs. "nighttime")**: These labels approximate broad lighting conditions (natural daylight versus artificial or low light) but fail to capture subtler variations such as golden-hour tones, overcast skies, or indoor vs. outdoor illumination [Sakaridis et al., 2019, Yu et al., 2020].
- **Camera or sensor IDs (e.g., "Sony $\alpha 7$ III" vs. "Canon EOS 5D Mark IV")**: Device identifiers [Jackson et al., 2021] hint at resolution, color processing pipelines, and noise characteristics, yet they do not reveal photographer choices like composition, focal length, or scene selection biases [Tommasi et al., 2017].

Chapter 6

Name the Lable NOT Name the Dataset: Bias Mitigation Strategy

6.1 Sectional Introduction

This chapter develops practical procedures for mitigating dataset bias in image classification while preserving semantic performance. The central premise is a dual objective: promote representations that support *Name the Label* (semantic classification) and suppress information that enables *Name the Dataset* (source identification). The tension between these objectives motivates two proposed strategies that operationalise the conceptual regulariser in Chapter 5.

The first strategy, *One-hot MiniMax*, instantiates a Domain-Adversarial Neural Network [Ganin et al., 2016] by attaching a dataset classifier to a shared backbone through a gradient reversal layer (§6.3.1). The backbone is trained to minimise *Name the Label* loss while maximising *Name the Dataset* loss, yielding a minimax game controlled by a schedule for the reversal coefficient λ (§6.3.4). The second strategy, *Uniform Double Mini*, replaces adversarial optimisation with direct alignment to a uniform target: the dataset head is trained toward a uniform target $\boldsymbol{u} = \frac{1}{K}\mathbf{1}$, where K is the number of datasets, jointly minimised objective without gradient reversal (§6.3.6–§6.3.7).

Contributions include explicit loss formulations and gradient updates for both strategies (§6.3.8), a feasibility study of the two debiasing methods on DATASET 04 (§6.4), and a cross-model, cross-dataset evaluation of the two debiasing methods on DATASET 04/06 (§6.5). Results preview: both methods reduce *Name the Dataset* accuracy while slightly improving *Name the Label* accuracy; *Uniform Double Mini* converges earlier and requires less tuning than the *One-hot MiniMax*.

6.2 The Initial Ideal

Chapter 5 established that a model trained on a biased dataset is unable to distinguish generalizable visual features from dataset bias shortcuts. In theory, the difficulty could be overcome by augmenting the standard loss function with an explicit penalty that discourages any statistical prediction dependence on latent dataset bias shortcuts (Section 5.5.1) . Although the mutual-information regularizer proposed in Chapter 5 cannot be implemented directly—because the dataset bias shortcut variables are never observed—its conceptual role remains instructive: the learning algorithm should *simultaneously* reward generalizable features that are essential for the classification task *and* suppress features like dataset bias shortcuts that merely encode non-semantic information of dataset identity.

In essence, all the experiments done in this paper up to now evolves around two tasks—Name the Label and Name the Dataset:

1. **Name the Label.** Images from TinyImageNet and CIFAR-100 are merged and labelled solely by label (class). The corresponding loss quantifies how accurately the model recovers semantic content.
   Minimising this loss is desirable, because lower values indicate stronger alignment with ground-truth semantics.

2. **Name the Dataset.** A linear probe is trained on the frozen backbone obtained from the previous Name the Label task; its output dimension equals the number of source datasets (two in the present study). The probe's accuracy, and hence its loss, reflects how much dataset bias remains in the learned representation.
   A larger loss is preferable, because high values imply that dataset bias shortcuts—such as sensor noise or compression patterns—have NOT been learn by the model's backbone in previous Name the Label task.

Taken together, the two tasks expose the fundamental tension that motivates subsequent dataset bias mitigation methods:

- The loss for Name the Label ought to be *minimised*.
- The loss for Name the Dataset ought to be *maximised*.

The tension between these two objectives crystallises the modelling dilemma introduced in Chapter 5: improving Name the Label accuracy without simultaneously encoding dataset bias. The natural question therefore arises:

Is it possible to minimise the Name-the-Label loss while maximising the Name-the-Dataset loss simultaneously within a single training loop?

Adversarial learning frameworks offer a principled answer. In particular, the gradient-reversal mechanism of Domain-Adversarial Neural Networks (DANN) [Ganin et al., 2016], demonstrate that a network can be trained to excel on one objective while being actively discouraged from succeeding on another that is adversarially defined. By treating the Name the Dataset head (linear probe) as an adversary, gradients that would normally reinforce Dataset bias shortcut features can be inverted, thereby penalising reliance on dataset bias without explicit access to the latent dataset bias shortcuts component. The present chapter expands upon this insight, translating the abstract regulariser (Section 5.5.1) of Chapter 5 into a concrete mitigation strategy that alternates cooperative and adversarial optimisation signals.

6.3 Proposed Two Dataset Bias Mitigation Methods in Theory

6.3.1 DANN Structure for Dataset Bias Mitigation

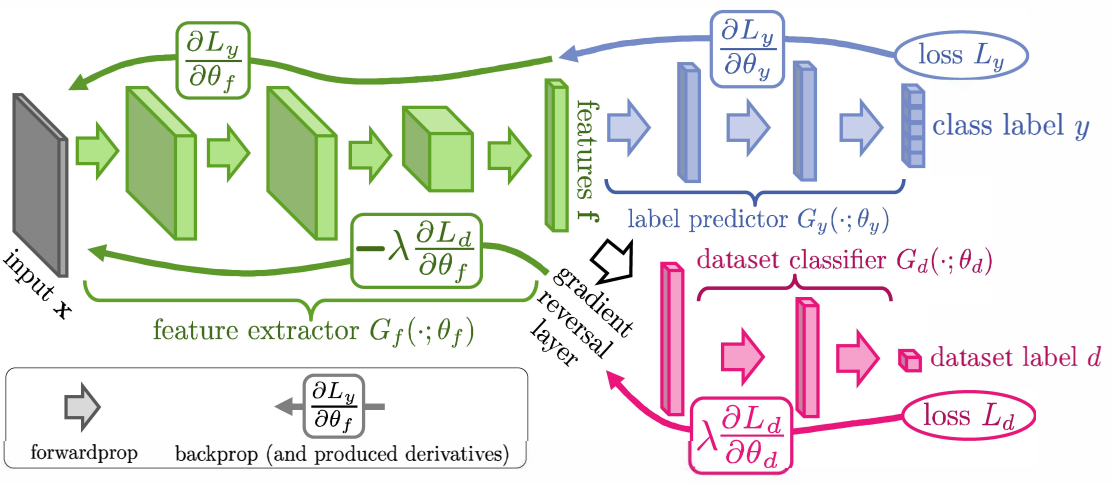


FIGURE 6.1: The proposed architecture based on DANN [Ganin et al., 2016] includes feature extractor(backbone) (green) and a label classification head(fully connected layer) (blue), which together form a standard feed-forward architecture. The Name the Dataset adversarial training is achieved by adding a dataset classification head(fully connected layer) (red) connected to the feature extractor via a gradient reversal layer that multiplies the gradient by a certain negative constant during the backpropagation-based training.

The proposed mitigation strategy extends a Name the Label model by incorporating an auxiliary Name the Dataset branch and a gradient reversal layer (GRL) like DANN [Ganin et al., 2016]. As illustrated in Figure 6.1, the network comprises three parameter sets:

$$\{\theta_f,\ \theta_y,\ \theta_d\},$$

where:

- θ_f parameterizes the *shared feature extractor* f, also called the backbone, which may be instantiated as the feature extractor of a ResNet, a ViT, or any other deep architecture for encoding images into feature representations.
- θ_y parameterizes the *Name the Label head* $f^{(y)}(\boldsymbol{x})$, a fully-connected layer projecting the feature vector to one logit per class before softmax.
- θ_d parameterizes the *Name the Dateset head* $f^{(d)}(\boldsymbol{x})$, a fully-connected layer projecting the same feature vector to one logit per dataset of origin before softmax.

6.3.2 DANN: Adversarial Interplay via Gradient Reversal Layer

This adversarial interplay is implemented through a Gradient Reversal Layer (GRL) [Ganin et al., 2016]. During forward propagation, the GRL behaves as an identity mapping, passing the feature extractor output to the Name the Dataset head without any modification. During backpropagation, however, it multiplies the gradient of the Name the Dataset loss with respect to θ_f by $-\lambda$, effectively updating the feature extractor to "fool" the Name the Dataset classifier while preserving its capacity for the Name the Label task.

6.3.3 One-hot MiniMax: Loss Functions

To begin with, the network is trained using two complementary loss functions that reflect its dual objectives: accurate semantic label classification and dataset bias invariant feature learning.

1. **Name the Label Loss**
   The principal objective is to learn generalizable feature representations that reliably Name the Label of each input image. This is realised by minimising the Name the Label loss

$$\mathscr{L}_{\mathrm{y}}(\theta_f, \theta_y) = \mathscr{L}_{\mathrm{label}}(\theta_f, \theta_y) = \frac{1}{N}\sum_{i=1}^{N} \mathrm{CE}\big(\hat{\boldsymbol{y}}_i,\ \boldsymbol{y}_i\big) \tag{6.3.1}$$

where $\hat{\boldsymbol{y}}_i = \text{softmax}\big(f^{(y)}(\boldsymbol{x}_i)\big)$ stands for one-hot model predicted class labels after softmax, $\{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_{i=1}^N$ denotes a set of N samples with image $\boldsymbol{x}_i$ and one-hot true class labels $\boldsymbol{y}_i$, and $\text{CE}(\cdot, \cdot)$ is the cross-entropy function.

Both feature-extractor parameters θ_f and Name the Label head θ_d are optimised to minimise this loss.

2. **Name the Dataset Loss (One-hot Minimax)**
   In contrast, an adversarial branch penalises features that reveal dataset ID. The Name the Dataset loss is defined as

$$\mathscr{L}_{\text{d}}(\theta_f, \theta_d) = \mathscr{L}_{\text{dataset}}(\theta_f, \theta_d) = \frac{1}{N} \sum_{i=1}^{N} \text{CE}\big(\hat{\boldsymbol{d}}_i,\, \boldsymbol{d}_i\big) \tag{6.3.2}$$

   where $\hat{\boldsymbol{d}}_i = \text{softmax}\big(f^{(d)}(\boldsymbol{x}_i)\big)$ one-hot model predicted dataset ID vector after softmax, $\{(\boldsymbol{x}_i, \boldsymbol{d}_i)\}_{i=1}^N$ denotes a set of N samples with image $\boldsymbol{x}_i$ and one-hot true dataset ID $\boldsymbol{d}_i$, and $\text{CE}(\cdot, \cdot)$ is the cross-entropy function.

   The Name the Dataset head parameters θ_d are optimised to minimise this loss, while the feature-extractor (backbone) parameters θ_f are driven to maximise it, thereby encouraging the learning of dataset bias-invariant features.

3. **Minimax Objective for Feature Extractor**
   By combining these two components, the training can be cast as a minimax game:

$$\mathscr{L}_{\text{minimax}}(\theta_f, \theta_y, \theta_d) = \min_{\theta_f, \theta_y} \max_{\theta_d} \Big[\mathscr{L}_{\text{y}}(\theta_f, \theta_y) \;-\; \lambda\, \mathscr{L}_{\text{d}}(\theta_f, \theta_d)\Big] \tag{6.3.3}$$

$$= \min_{\theta_f, \theta_y} \max_{\theta_d} \frac{1}{N} \sum_{i=1}^{N} \Big[\text{CE}\big(\hat{\boldsymbol{y}}_i,\, \boldsymbol{y}_i\big) \;-\; \lambda\, \text{CE}\big(\hat{\boldsymbol{d}}_i,\, \boldsymbol{d}_i\big)\Big]. \tag{6.3.4}$$

   where the hyperparameter $\lambda > 0$ controls the trade-off between Name the Label accuracy and dataset bias invariance.

   The approach is termed **One-hot MiniMax**: In the view of feature-extactor (backbone), the Name-the-Label loss is minimized while the Name-the-Dataset loss is maximized and Name-the-Dataset loss is defined on one-hot targets and solved via the minimax game, ensuring that the feature extractor learns representations that simultaneously minimise semantic classification error and maximise confusion of the dataset discriminator.

6.3.4 One-hot Minimax: Gradient Updates & Parameter Optimisation

Let η_f, η_y, and η_d be the learning rates for the feature extractor, the Name-the-Label head, and the Name-the-Dataset head, respectively. For a mini-batch of size N, the required gradients are

$$\frac{\partial \mathscr{L}_y}{\partial \theta_y} = \frac{1}{N}\sum_{i=1}^{N}\frac{\partial\,\mathrm{CE}(\hat{\boldsymbol{y}}_i, \boldsymbol{y}_i)}{\partial \theta_y}, \tag{6.3.5}$$

$$\frac{\partial \mathscr{L}_d}{\partial \theta_d} = \frac{1}{N}\sum_{i=1}^{N}\frac{\partial\,\mathrm{CE}(\hat{\boldsymbol{d}}_i, \boldsymbol{d}_i)}{\partial \theta_d}, \tag{6.3.6}$$

$$\frac{\partial \mathscr{L}_{\text{minimax}}}{\partial \theta_f} = \frac{\partial \mathscr{L}_y}{\partial \theta_f} - \lambda\frac{\partial \mathscr{L}_d}{\partial \theta_f} \tag{6.3.7}$$

where the Gradient Reversal Layer multiplies $\partial\mathscr{L}_d/\partial\theta_f$ by $-\lambda$ during back-propagation, forcing the feature extractor to discard dataset bias.

The corresponding parameter updates for a single mini-batch are

$$\theta_y \leftarrow \theta_y - \eta_y\frac{\partial \mathscr{L}_y}{\partial \theta_y}, \tag{6.3.8}$$

$$\theta_d \leftarrow \theta_d - \eta_d\frac{\partial \mathscr{L}_d}{\partial \theta_d}, \tag{6.3.9}$$

$$\theta_f \leftarrow \theta_f - \eta_f\big(\frac{\partial \mathscr{L}_y}{\partial \theta_f} - \lambda\frac{\partial \mathscr{L}_d}{\partial \theta_f}\big). \tag{6.3.10}$$

The first two expressions guide θ_y and θ_d toward more accurate classification of labels and datasets, respectively. The last expression steers θ_f toward features that improve Name the Label classification while simultaneously suppressing features that facilitate Name the Dataset, thus realising the adversarial objective of the Domain-Adversarial Neural Network.

6.3.5 One-hot Minimax: Adaptive Coefficient λ Schedule

The hyperparameter λ governs the strength of the adversarial signal by scaling the gradient reversal layer. It is defined as a function (Fig.6.2) of the normalized training progress $p \in [0, 1]$:

$$\lambda(p) = \lambda_{\max}\left(\frac{2}{1+\exp(-\gamma\, p)} - 1\right),$$

where

- $\lambda_{\max}$ specifies the maximum adaptation coefficient,

- γ controls the curvature of the increasing schedule,
- $p \in [0, 1]$ denotes the normalized training progress, computed as

$$p = \frac{e}{E},$$

where e is the current epoch index and E is the total number of epochs.

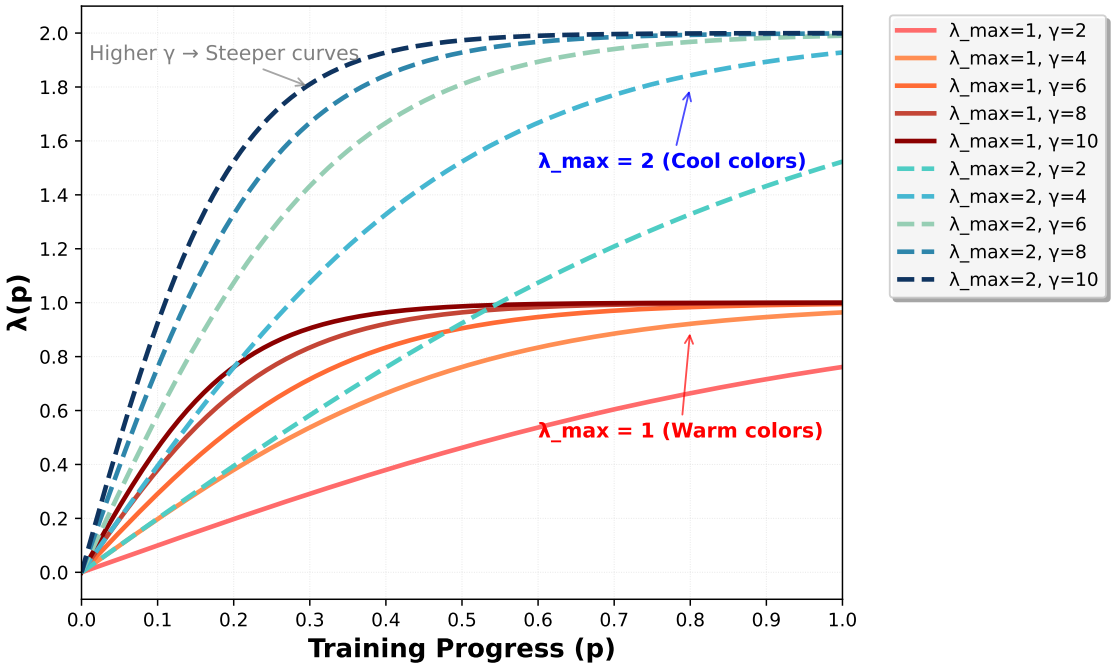


FIGURE 6.2: The change curve of $\lambda(p)$ with training progress under different γ and $\lambda_{\max}$ conditions.

6.3.6 One-hot Minimax to Uniform Double Mini

The one-hot minimax formulation in Section 6.3.4 maximises the Name-the-Dataset loss with respect to the feature-extractor parameters θ_f by means of a gradient-reversal layer. In practice, this requires careful tuning of the coefficient schedule $\lambda(p)$ (see Fig. 6.2) to keep the Name-the-Dataset classifier accuracy close to chance level (50% for two datasets). *Suppressing accuracy below chance (e.g. to 20%) is not desirable: an accuracy significantly lower than 50% merely indicates that the model has learned to invert its predictions to reduce the loss.* In such cases, the backbone still encodes dataset biases, but the model achieves a low loss by flipping its outputs (for instance, naming "ImageNet" when the sample is predicted by the model to originate from CIFAR-100). Thus, maintaining the Name the Dataset accuracy at 50% ensures that the model is genuinely unable to distinguish datasets, rather than simply reversing its decisions.

Maintaining this 50% equilibrium throughout training is difficult: either λ is too small or too large, dataset bias remains. An alternative is to replace the adversarial Name the Dataset loss maximisation with *direct alignment* of the model's Name the Dataset output with the uniform distribution.

Instead of driving θ_f to *maximise* a loss that the Name the Dataset head *minimises*, both branches are trained to *minimise* a common objective and the Name the Dataset head now penalises any deviation from uniformity. Because the optimum of this objective is reached when the dataset classifier cannot distinguish between source datasets, the approach is termed **Uniform Double Mini** (UDM): both the Name-the-Label and Name-the-Dataset objectives are minimised simultaneously.

6.3.7 Uniform Double Mini: Loss Functions

The *Uniform Double Mini* (UDM) strategy removes that fragility by **replacing the minimax game with two simultaneous minimisations**:

- **Main task:** The feature-extractor (backbone) acquires the Name the Label power by minimising the usual Name-the-Label loss $\mathscr{L}_{\mathrm{y}}$ and simultaneously maximizing the Name-the-Dataset loss $\mathscr{L}_{\mathrm{d}}$, but maintaining a 50% equilibrium of Name the Dataset accuracy during maximizing is difficult.
- **Uniform domain alignment:** *Directly* force the Name-the-Dataset head to output the uniform distribution $\boldsymbol{u} = \frac{1}{K}\mathbf{1}$ rather than the one-hot ground truth. Aligning every prediction with $\boldsymbol{u}$ ensures the Name the Dataset classifier cannot distinguish dataset origin, automatically clamping its accuracy to guess by chance level without the need for a gradient-reversal layer or a schedule for λ.

Because both losses are minimised together ("double mini"), implementation reduces to a single weighted sum objective and *standard* back-propagation; no adversarial optimisation loop is required.

1. **Name the Label Loss**
   The main-task Name the Label loss remains identical to equation (6.3.2):

$$\mathscr{L}_{\mathrm{y}}(\theta_f, \theta_y) = \mathscr{L}_{\mathrm{label}}(\theta_f, \theta_y) = \frac{1}{N}\sum_{i=1}^{N} \mathrm{CE}(\hat{\boldsymbol{y}}_i,\, \boldsymbol{y}_i), \quad \hat{\boldsymbol{y}}_i = softmax\big(f^{(y)}(\boldsymbol{x}_i)\big)$$

2. **Name the Dataset Loss (Uniform Double Mini)**
   Let K denote the number of source datasets ($K = 2$ in the present study) and

define the uniform target distribution

$$\boldsymbol{u} \;=\; \Big[\tfrac{1}{K},\,\ldots,\,\tfrac{1}{K}\Big]^{\top} \in \mathbb{R}^{K}. \tag{6.3.11}$$

Instead of using the true dataset labels $\boldsymbol{d}_i$ as targets, the dataset branch is trained to output the uniform distribution $\boldsymbol{u}$:

$$\mathscr{L}_{\mathrm{u}} = \mathscr{L}_{\mathrm{uniform}}(\theta_f, \theta_d) \;=\; \frac{1}{N}\sum_{i=1}^{N} \mathrm{CE}\big(\hat{\boldsymbol{d}}_i,\; \boldsymbol{u}\big), \quad \hat{\boldsymbol{d}}_i = \mathrm{softmax}\big(f^{(d)}(\boldsymbol{x}_i)\big). \tag{6.3.12}$$

where $\hat{\boldsymbol{d}}_i = \mathrm{softmax}\big(f^{(d)}(\boldsymbol{x}_i)\big)$ one-hot model predicted dataset ID vector after softmax, $\{(\boldsymbol{x}_i, \boldsymbol{d}_i)\}_{i=1}^{N}$ denotes a set of N samples with image $\boldsymbol{x}_i$ and one-hot true dataset ID $\boldsymbol{d}_i$, and $\mathrm{CE}(\cdot,\cdot)$ is the cross-entropy function.

Minimising $\mathscr{L}_{\mathrm{u}}$ forces the Name the Dataset logits to converge toward equal posterior mass, thereby eliminating domain-specific information from the shared representation without the need for explicit adversarial optimisation.

3. **Composite Objective**
   The total loss minimised during training is

$$\mathscr{L}_{\mathrm{UDM}}(\theta_f, \theta_y, \theta_d) = \min_{\theta_f,\theta_y,\theta_d} \Big[\mathscr{L}_{\mathrm{y}}(\theta_f, \theta_y) \;+\; \lambda\, \mathscr{L}_{\mathrm{d}}(\theta_f, \theta_d)\Big] \tag{6.3.13}$$

$$= \min_{\theta_f,\theta_y,\theta_d} \frac{1}{N}\sum_{i=1}^{N}\Big[\mathrm{CE}\big(\hat{\boldsymbol{y}}_i,\, \boldsymbol{y}_i\big) \;+\; \lambda\, \mathrm{CE}\big(\hat{\boldsymbol{d}}_i,\, \boldsymbol{u}\big)\Big]. \tag{6.3.14}$$

where $\lambda \geq 0$ modulates the strength of the Name the Dataset uniformity constraint and is held constant.

Simultaneous minimisation of $\mathscr{L}_{\mathrm{y}}$ and $\mathscr{L}_{\mathrm{u}}$ preserves main-task Name the Dataset performance while encouraging the learned representation to be invariant to dataset bias features. Empirically, Uniform Double Mini attains Name the Dataset accuracies close to the theoretical chance level (50 % when $K = 2$) without extensive hyperparameter tuning.

6.3.8 Uniform Double Mini: Gradient Updates & Parameter Optimisation

Let η_f, η_y, and η_d denote the learning rates assigned to the feature extractor, the Name-the-Label head, and the Name-the-Dataset head, respectively. For a set of size N, the gradients of the Uniform Double Mini objective

$$\mathscr{L}_{\mathrm{UDM}}(\theta_f, \theta_y, \theta_d) = \min_{\theta_f, \theta_y, \theta_d} \Big[\mathscr{L}_{\mathrm{y}}(\theta_f, \theta_y) \; + \; \lambda \, \mathscr{L}_{\mathrm{d}}(\theta_f, \theta_d) \Big] \tag{6.3.15}$$

$$= \min_{\theta_f, \theta_y, \theta_d} \; \frac{1}{N} \sum_{i=1}^{N} \Big[\mathrm{CE}\big(\hat{\boldsymbol{y}}_i, \, \boldsymbol{y}_i\big) \; + \; \lambda \, \mathrm{CE}\big(\hat{\boldsymbol{d}}_i, \, \boldsymbol{u}\big) \Big]. \tag{6.3.16}$$

are given by

$$\frac{\partial \mathscr{L}_{\mathrm{UDM}}}{\partial \theta_y} = \; \frac{\partial \mathscr{L}_{\mathrm{y}}}{\partial \theta_y} \; = \; \frac{1}{N} \sum_{i=1}^{N} \frac{\partial \, \mathrm{CE}\big(\hat{\boldsymbol{y}}_i, \boldsymbol{y}_i\big)}{\partial \theta_y}, \tag{6.3.17}$$

$$\frac{\partial \mathscr{L}_{\mathrm{UDM}}}{\partial \theta_d} = \; \lambda \, \frac{\partial \mathscr{L}_{\mathrm{u}}}{\partial \theta_d} \; = \; \frac{\lambda}{N} \sum_{i=1}^{N} \frac{\partial \, \mathrm{CE}\big(\hat{\boldsymbol{d}}_i, \boldsymbol{u}\big)}{\partial \theta_d}, \tag{6.3.18}$$

$$\frac{\partial \mathscr{L}_{\mathrm{UDM}}}{\partial \theta_f} = \; \frac{\partial \mathscr{L}_{\mathrm{y}}}{\partial \theta_f} \; + \; \lambda \, \frac{\partial \mathscr{L}_{\mathrm{u}}}{\partial \theta_f}. \tag{6.3.19}$$

Unlike the One-hot MiniMax formulation, no gradient-reversal layer is required; both loss components contribute additively to the feature-extractor (backbone) update.

The corresponding parameter updates for a single optimisation step are

$$\theta_y \leftarrow \theta_y - \eta_y \, \frac{\partial \mathscr{L}_{\mathrm{y}}}{\partial \theta_y}, \tag{6.3.20}$$

$$\theta_d \leftarrow \theta_d - \eta_d \, \lambda \frac{\partial \mathscr{L}_{\mathrm{u}}}{\partial \theta_d}, \tag{6.3.21}$$

$$\theta_f \leftarrow \theta_f - \eta_f \Big(\frac{\partial \mathscr{L}_{\mathrm{y}}}{\partial \theta_f} \; + \; \lambda \frac{\partial \mathscr{L}_{\mathrm{u}}}{\partial \theta_f} \Big). \tag{6.3.22}$$

The constant coefficient λ modulates the influence of the Name the Dataset uniformity constraint throughout training.

6.4 Experiment 5: A Feasibility Study for One-Hot Minimax & Uniform Double Mini

6.4.1 Experiment 5: Objective

The primary goal of this experiment is to verify whether the proposed dataset bias mitigation strategies—One-Hot Minimax and Uniform Double Mini—can effectively suppress the learning of dataset bias, thereby encouraging the model to capture truly semantic, generalizable features. Each method can be assessed by comparing:

- **Name the Label accuracy:** the Top-1/Top-5 semantic classification test performance on the Dataset 04 test set.
- **Name the Dataset accuracy:** the test performance of a linear probe to recover source-dataset identity of an image from the frozen backbone on the Dataset 04 test set.

Results are benchmarked against a baseline ResNet-18 trained without any bias mitigation, to quantify the improvement in semantic generalization and the reduction in residual dataset bias.

6.4.2 Experiment 5: Workflow

The procedural pipeline, illustrated in Figure 6.3, unfolds in five sequential stages:

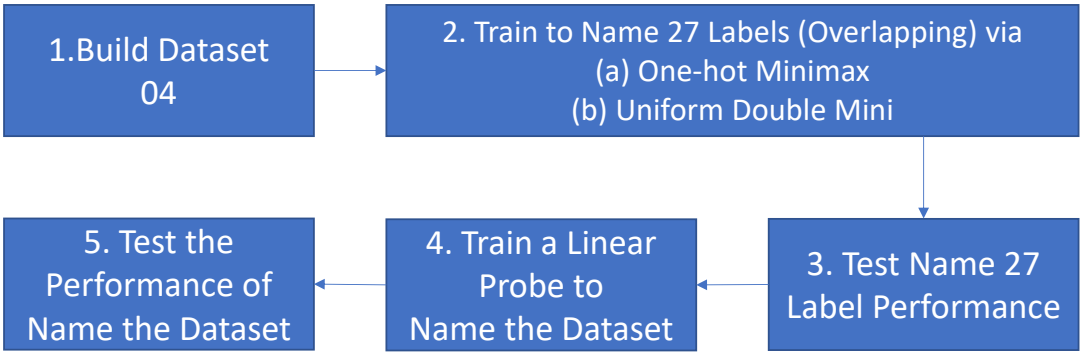


FIGURE 6.3: Workflow of Feasibility Study for One-Hot Minimax & Uniform Double Mini

1. **Construct Dataset 04**
   Assemble *Dataset 04* following the Dataset construction workflow in Section 3.3.2. The dataset contains 27 semantically overlapping labels drawn from TinyImageNet and CIFAR-100.
2. **Train to Name 27 Labels**
   Apply Protocol A (with dataset bias mitigation; Section 3.5.2) to train ResNet18 on the Dataset 04 to Name 27-label under two alternative dataset bias-mitigation methods proposed in Section 6.3:

   (a) *One-Hot MiniMax* (adversarial, gradient-reversal)

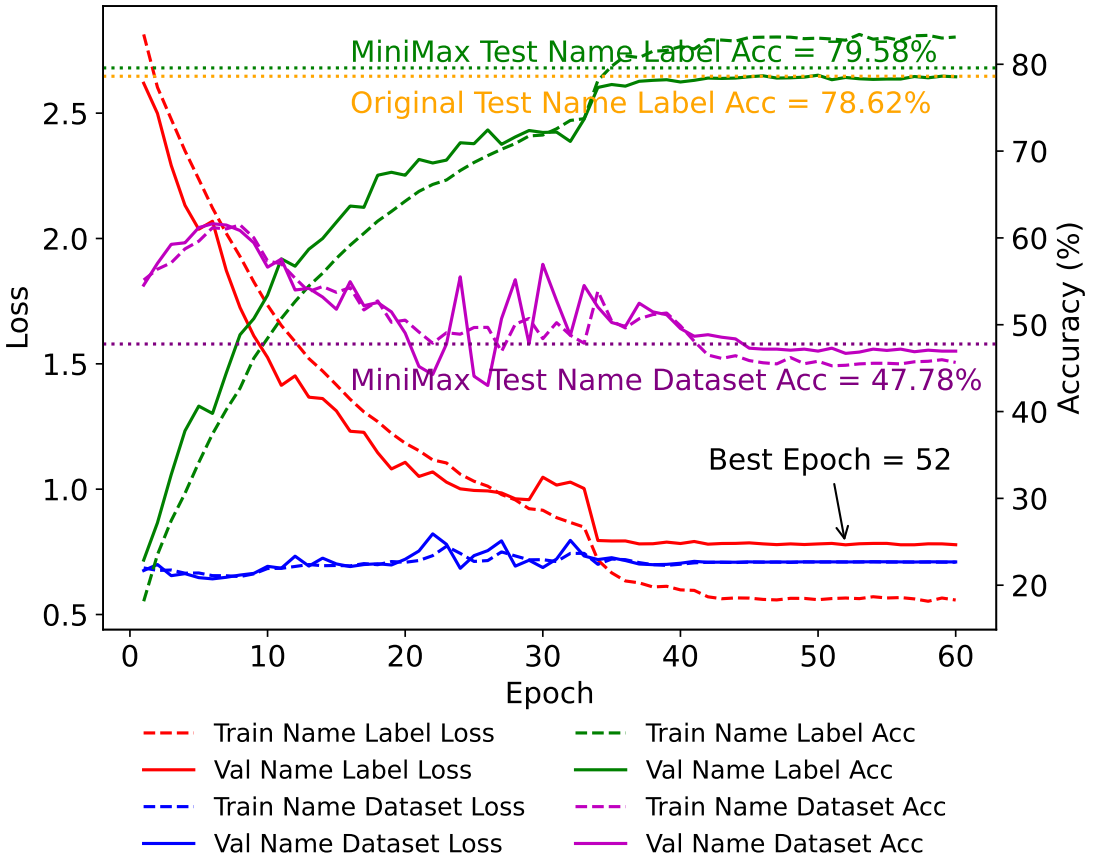
   (b) *Uniform Double Mini* (direct uniform-alignment)

3. **Evaluate Label Classification Performance**
   Record Top-1 and Top-5 Name-the-Label accuracy on the held-out test split of Dataset 04 for every trained model.

4. **Train a Linear Probe to Name the Dataset**
   Following Protocol C (Section 3.5.4), freeze the feature extractor (backbone) from step 2 and fit a two-way linear classifier that distinguishes between the two source datasets.

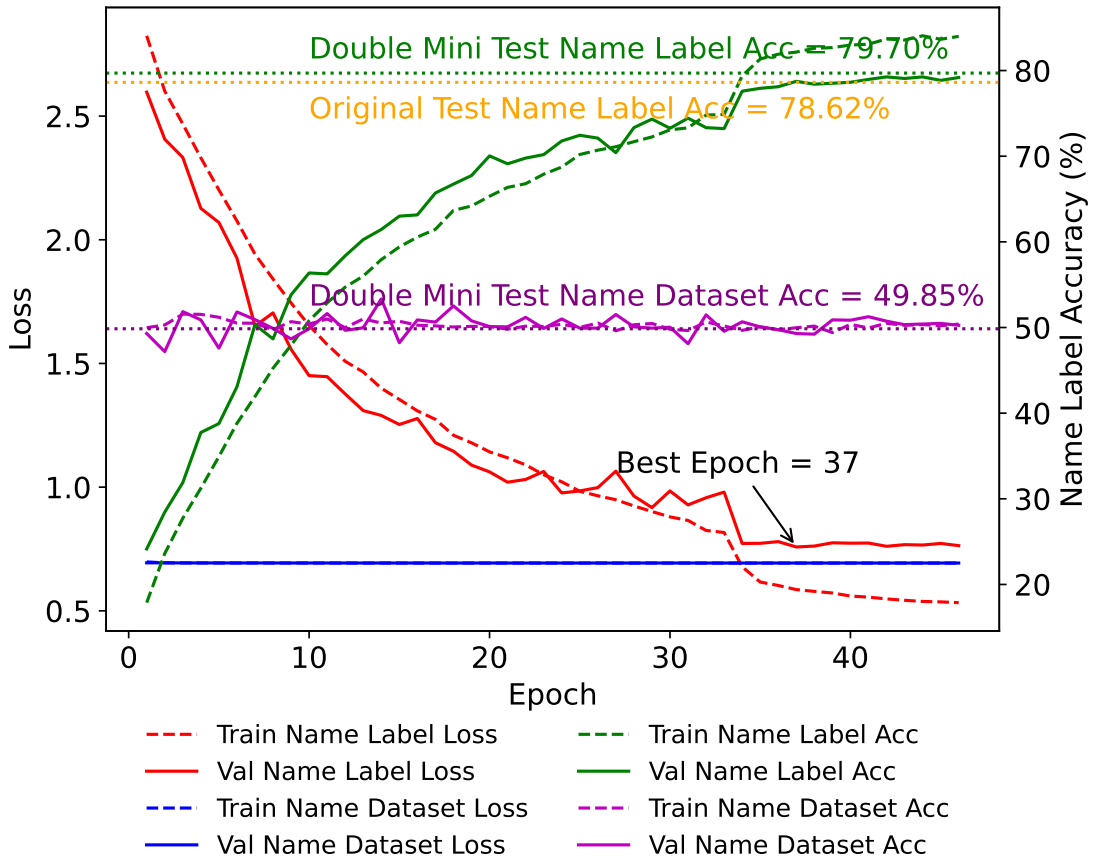
5. **Evaluate Name the Dataset Performance**
   Measure the probe's accuracy on the same test split. A result closer to 50% suggests that the backbone contains weaker dataset bias.

6.4.3 Experiment 5: Results



Note: From Figure B.1, A Resnet-18 Trained Using Protocoal A to Name 27 Label without One-hot MiniMax Dataset Bias Mitigation Method is 78.62%

FIGURE 6.4: Train & Test Log of A ResNet-18 Model Trained to Name 27 Label with **One-hot MiniMax**($\lambda = 0.4, \gamma = 2$) Dataset Bias Mitigation Method Following Protocol A (Section 3.5.2) on Dataset 04 (Tab.A.1)



Note: From Figure B.1, A Resnet-18 Trained Using Protocoal A to Name 27 Label without One-hot MiniMax Dataset Bias Mitigation Method is 78.62%

FIGURE 6.5: Train & Test Log of A ResNet-18 Model Trained to Name 27 Label with **Uniform Double Mini**($\lambda = 0.4$) Dataset Bias Mitigation Method Following Protocol A (Section 3.5.2) on Dataset 04 (Tab.A.1)

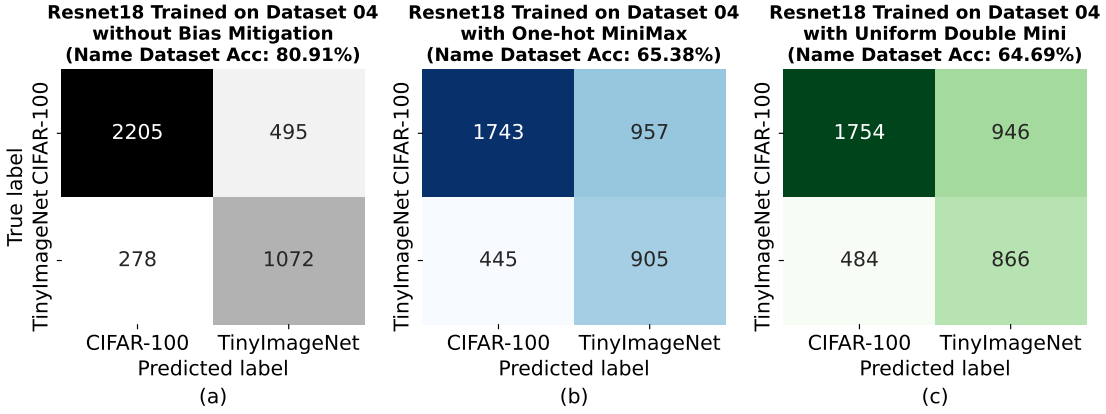


FIGURE 6.6: Comparison of Name Dataset ***Test*** Confusion Matrices of ResNet18 Pretrained (Protocal A) and Linear Probed (Protocal C) Using No Bias Mitigation, One-Hot MiniMax and Uniform Double Mini Respectively on Dataset 04

6.4.4 Experiment 5: Analysis

Analysis of the experimental results from Figures 6.4, 6.5 and 6.6 highlights four key findings regarding the performance of the bias mitigation methods:

1. **Semantic Performance Improvement:** Both mitigation strategies successfully improve upon the baseline's **Name-the-Label** accuracy of 78.62%. The One-hot MiniMax method achieves a final test accuracy of 79.58%, while the Uniform Double Mini method reaches a slightly higher accuracy of 79.70%.
2. **Effective Bias Suppression in Confusion Matrices:** The primary measure of dataset bias, the **Name-the-Dataset** linear probe accuracy, shows a dramatic reduction from the baseline's 80.91%. The One-hot MiniMax and Uniform Double Mini methods lower this accuracy to 65.38% and 64.69%, respectively.
3. **Training Efficiency:** The Uniform Double Mini method demonstrates higher training efficiency. It achieves its best validation performance at epoch 37, significantly earlier than the One-hot MiniMax model, which peaks at epoch 52.
4. **Adversarial Minimax Training Dynamics:** The training logs (Figures 6.4 and 6.5) show that the adversarial classifier's accuracy ("Name the Dataset Acc") hovers near 50% in final epochs. The One-hot MiniMax method fluctuates intensively in early 40 epochs, while the Uniform Double Mini method fluctuates in a much more stable way.

6.4.5 Experiment 5: Discussion

The results confirm that both proposed strategies are viable for mitigating dataset bias while simultaneously improving Name the Label performance.

- **Achievement of Dual Objectives:** The experiments successfully demonstrate that it is possible to both suppress dataset bias and enhance semantic classification generalization. The models achieved higher Name the Label accuracy (the primary goal) precisely because they were discouraged from relying on spurious, dataset-specific shortcuts.
- **Incomplete but Substantial Bias Removal:** The Name the Dataset linear probe accuracies of $\sim 65\%$ are significantly lower than the baseline's $\sim 81\%$ but remain above the ideal chance level of 50%. This crucial finding implies that the feature representations are made *less biased*, but are not rendered completely *unbiased*. The mitigation is effective but not absolute.

The Uniform Double Mini method may be a more efficient strategy than One-hot MiniMax for this task.

- **Training Efficiency:** The Uniform Double Mini method demonstrates superior training efficiency. It achieves its best validation performance at epoch 37, significantly earlier than the One-hot MiniMax model, which peaks at epoch 52. The following experiments will focus on Uniform Double Mini method since it is much more computationally efficient.

6.5 Experiment 6: Uniform Double Mini Name Label Accuracy Performance Across Models and Datasets

6.5.1 Experiment 6: Objective

Building on the findings from Experiment 5, which identified the Uniform Double Mini method as a more efficient and effective strategy for bias mitigation. The primary objective of this experiment is to evaluate the performance of the Uniform Double Mini dataset bias mitigation method's Name-the-Label accuracy across combinations of different datasets (Dataset 04 and Dataset 06) and different model architectures (ResNet-18 and Vision Transformer), using a fixed mitigation strength parameter $\lambda = 0.4$. This will test whether the benefits observed in the previous experiment hold true under varied conditions.

6.5.2 Experiment 6: Workflow

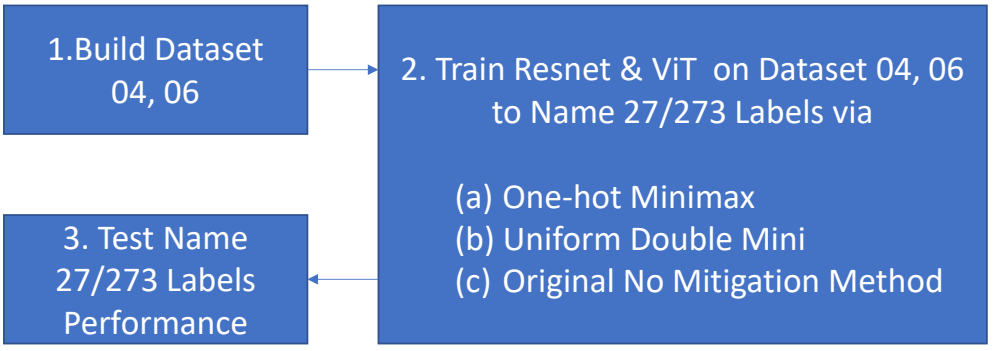
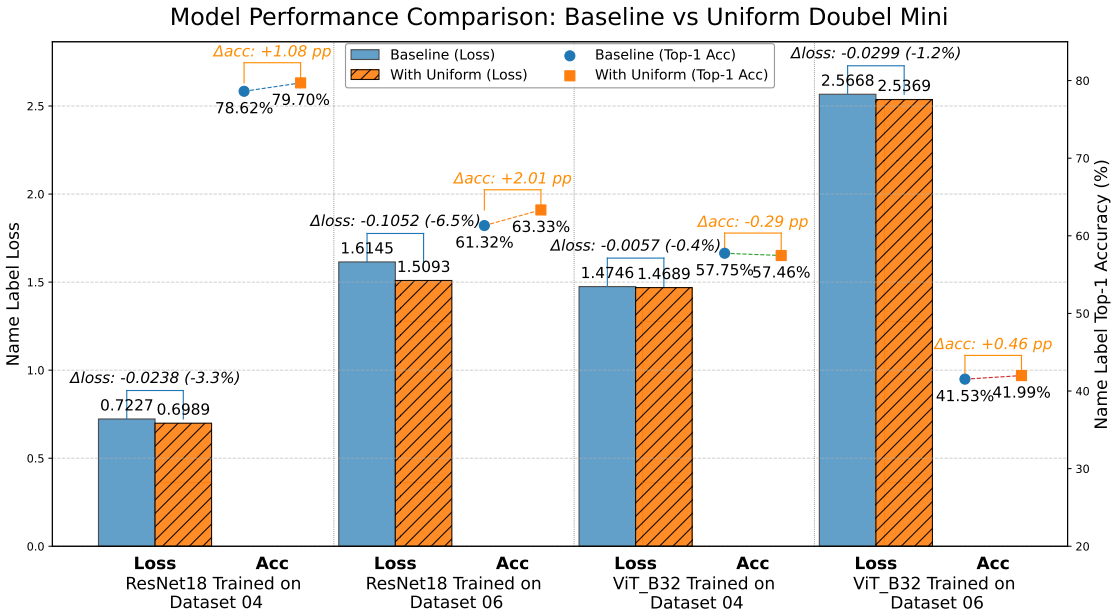


FIGURE 6.7: Uniform Double Mini Dataset Bias Mitigation Method Name Label Accuracy Performance Across Models and Datasets Workflow

The experimental procedure, as illustrated in Figure 6.7, is streamlined to focus on the impact of the Uniform Double Mini method on the primary classification task. It follows these three steps:

1. **Construct Datasets 04 & 06.** Assemble both Dataset 04 (27 overlapping labels) and Dataset 06 (273 overlapping labels) according to the construction workflows defined previously.
2. **Train Models to Name Labels.**
   (a) Train ResNet-18 using Protocol A on each dataset (04 and 06) twice: once using the Uniform Double Mini bias mitigation method with $\lambda = 0.4$, and once using the original training protocol (no mitigation) to establish a baseline.
   (b) Train a Vision Transformer (ViT) using Protocol B on each dataset (04 and 06) twice: once using the Uniform Double Mini bias mitigation method with $\lambda = 0.4$, and once using the original training protocol (no mitigation) to establish a baseline.
3. **Evaluate Name-the-Label Performance.** Record and compare the final Top-1 Name the Label test accuracy for all trained models. This allows for a direct comparison of Name-the-Label performance with and without the Uniform Double Mini mitigation across the different models and datasets.

6.5.3 Experiment 6: Results



Note: Within-model comparisons of Baseline vs. Uniform Double Mini. Loss bars show Name the Label classification loss; accuracy markers show Name the Label top-1 accuracy. Reported values are from individual runs—due to stochastic training variability, results of both loss and accuracy can fluctuate by about ±1%.

FIGURE 6.8: Name 27/273 Labels Accuracy Comparison Across Models and Datasets Using Uniform Double Mini ($\lambda = 0.4$ ***for all combinations)*** Dataset Bias Mitigation Method Trained with Protocol A (Section 3.5.2) for ResNet18 or Protocol B (Section 3.5.3) for ViT_B32

6.5.4 Experiment 6: Analysis

Figure 6.8 compares the baseline models with their Uniform Double Mini ($\lambda = 0.4$) de-biased counterparts across two architectures and two datasets. Four principal observations emerge:

1. **Consistent loss reduction:** Compared with the baseline, the Uniform Double Mini lowers the Name–the–Label loss for all of the model–dataset combinations. The largest absolute decrease (Δloss $= -0.1052$, $-6.5\,\%$) is observed for RESNET-18 on DATASET 06.

2. **Marked accuracy gains for CNNs:** RESNET-18 benefits consistently: accuracy improves by +1.08 percentage points (pp) on DATASET 04 and by +2.01 pp on DATASET 06.

3. **Mixed response for Vision Transformers:** The ViT_B32 model records a modest accuracy gain on Dataset 06 (+0.46 pp) but a negligible accuracy decline on Dataset 04 (−0.29 pp).

4. **Better debiasing effect with increasing dataset scale and label richness:** The Uniform Double Mini method produces larger relative gains on the bigger, denser Dataset 06 (273 labels; approximately 160,000 images) than on Dataset 04 (27 labels; approximately 30,000 images); the reduction in Name-the-Label loss for Dataset 06 notably greater than that of Dataset 04: For ResNet-18, the reduction increases from 0.0238 (3.3%) to 0.1052 (6.5%), while for ViT_B32 it increases from 0.0057 (0.4%) to 0.0299 (1.2%).

6.5.5 Experiment 6: Discussion

The Uniform Double Mini is a generally effective dataset biasmitigation strategy across different datasets and architectures combinations; And its debiasing effect increases with dataset scale and label richness.

- **Consistent loss reduction:** For every model–dataset pair, the Name the Label loss is lower than the baseline on the held-out test split (Fig. 6.8).
- **Increasing gain with increasing dataset scale:** The method becomes more beneficial when moving from Dataset 04 (27 labels) to the far larger and more diverse Dataset 06 (273 labels).

Convolutional models like ResNets profit more than transformer models like ViT from the imposed uniformity.

- **Larger absolute gains:** ResNet-18 enjoys accuracy increases of +1.08 pp (Dataset 04) and +2.01 pp (Dataset 06), whereas the ViT_B32 model records a modest gain on Dataset 06 (+0.46 pp) but a negligible decline on Dataset 04 (−0.29 pp).
- **Architectural inductive biases:** The locality and translational equivariance baked into CNNs may synergise with the uniform-alignment objective, constraining feature space in a way that retains task-relevant spatial cues while suppressing dataset-specific artefacts; by contrast, the global self-attention of ViTs may require stronger or more granular regularisation.
- **Tuning opportunities:** Adaptive selection of the alignment weight λ or layer-wise application of the loss could help unlock comparable benefits for transformer backbones, a hypothesis warranting systematic exploration.

Debiasing does NOT necessarily trade off against downstream utility.

- **Favourable bias–utility balance:** In most cases, the Uniform Double Mini maintains Name-the-Label accuracy while prior experiments (Experiment 5 Sec. 6.4.4) showed a simultaneous drop of the Name-the-Dataset Accuracy from $\sim$ 81% to $\sim$ 65%. Thus, representation quality for the target semantics improves even as dataset bias is attenuated.
- **Practical implication:** These findings challenge the common assumption that debiasing inevitably harms performance, and position Uniform Double Mini as a competitive option for real-world applications that demand both fairness and accuracy.

Chapter 7

Global Discussion

7.1 Sectional Introduction

This chapter synthesizes empirical findings from Chapters 2–6 into a unified discussion addressing four core research questions. Integrating experimental results with the broader literature, the chapter aims to:

- consolidate evidence for the persistence of dataset bias in modern deep neural networks (*RQ1*);
- decompose the constituent elements of dataset bias and clarify their hierarchical relationships (*RQ2*);
- analyze how architectural properties mediate the propagation of bias across network layers (*RQ3*); and
- formalize dataset bias theoretically and critically evaluate mitigation strategies (*RQ4*).

In addition, the chapter concludes with a critical appraisal of this study's limitations, noting constraints in image resolution, architectural scope, bias modality coverage, and evaluation settings. These considerations delineate directions for future research aimed at improving external validity, enhancing robustness across data regimes, and strengthening the generality and practical applicability of the findings.

7.2 RQ1 – Can Modern DNNs Still Capture Dataset Bias?

Conclusion: Yes. State-of-the-art convolutional and transformer architectures (e.g., ResNet, ViT) continue to *reliably identify* the source dataset of an image. Even when only a *single frozen linear probe* is trained to Name the Dataset on top of a pre-trained backbone to Name the Label, accuracy remains unexpectedly high.

7.2.1 Evidence (Linear probes to Name the Dataset)

Freeze the backbone of a pre-trained ResNet18 to Name the Labels and then Replace the final classifier to Name 27 Labels by a *two-way* fully-connected layer and train *only that layer* to Name the Dataset still yielded 97.75 % accuracy on ResNet-18 (Section 4.2, Fig.4.3, 4.4), evidencing that dataset fingerprints are already linearly separable in the penultimate feature space.

7.2.2 Mechanistic analysis

These empirical findings build upon and extend the original *Name the Dataset* study by Torralba and Efros [2011], which—using hand-crafted features and shallow classifiers—first demonstrated that datasets carry distinctive statistical signatures. Subsequent work, including the deep-feature revisit by Tommasi et al. [2017], showed that increasing representational power does not eliminate such signatures; if anything, modern high-capacity models tend to *amplify* dataset-level idiosyncrasies. The phenomenon arises probably because overparameterized networks (like DNNs) can fit both semantic and subtle non-semantic correlations, making the dataset origin an easily separable signal in the embedding space. Deep layers transform and reorder information, but do not orthogonalize away dataset-specific non-semantic information (e.g., camera sensor noise, artifacts, JPEG quantisation patterns and other low-level cues introduced during data acquisition or preprocessing); instead, these non-semantic information become entangled with semantic features, hence a linear probe still retrieves dataset identity with high probability.

This phenomenon of high Name the Dataset accuracy aligns with the synthesized perspective in recent retrospectives [Liu and He, 2025, Zeng et al., 2024], which argue that dataset bias is resilient and often accentuated by modern architectures. The persistence of dataset fingerprints in the final semantic representations suggests that learning dynamics do not preferentially discard dataset bias unless explicitly constrained. In other

words, modern DNNs provide the flexibility to encode *both* generalizable semantic information and dataset specific shortcuts simultaneously, rather than forcing a trade-off that would eliminate the latter.

7.2.3 Practical implications

The fact that merely switching to use modern DNN or more expressive architectures does not erase dataset bias implies that "bigger models" are not a substitute for careful dataset-aware engineering.

Moreover, the ease with which dataset identity is recovered even after semantic training implies that any downstream evaluation or fairness audit should include explicit checks for dataset leakage: if a downstream task's performance is unexpectedly high on I.I.D. test set, it may be worthwhile to test whether the representations are implicitly encoding dataset origin, thereby creating spurious correlations that may induce generalization failure in O.O.D. scenarios.

In sum, the resilience and amplifying behavior of dataset bias necessitate operational safeguards beyond improvement in architecture—continuous monitoring, explicit bias diagnostics, and controlled adaptation are required to avoid brittle generalization when data provenance shifts.

7.3 RQ2 – What Components Constitute Dataset Bias, Which Dominate?

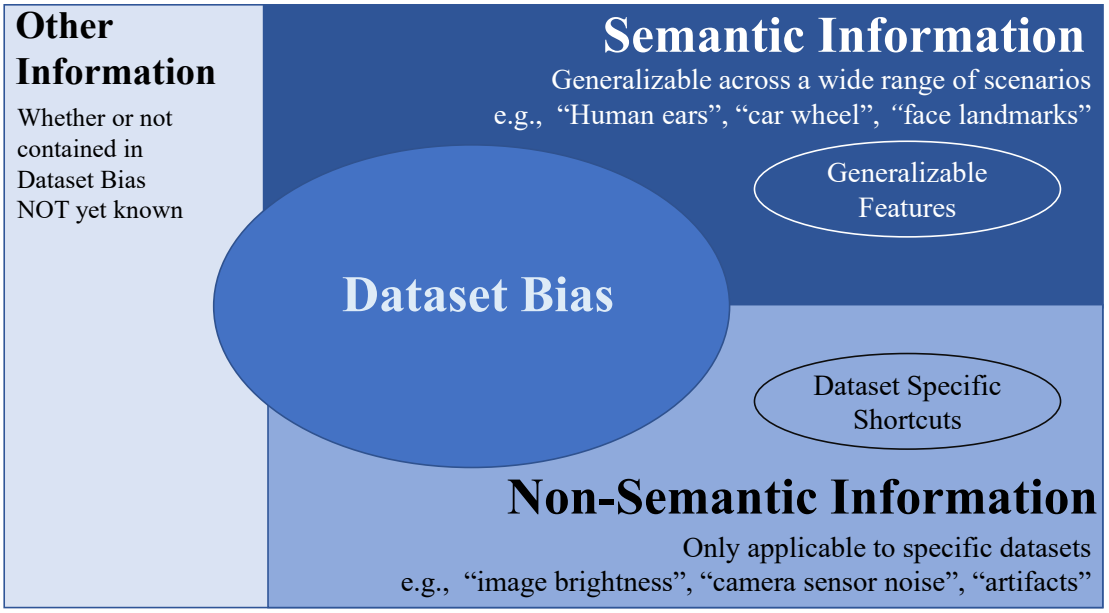


FIGURE 7.1: Decomposition of dataset bias into semantic, non-semantic information and others; non-semantic signals (e.g., interpolation artifacts, resolution and compression footprints) only applicable in specifics datasets, whereas semantic information is the transferable semantics that generalizable across a wide range of scenarios, becoming more visible when the dominant non-semantic shortcuts are weakened; whether other information is contained in dataset bias or not and the concrete form of other information is not yet known.

Conclusion: Dataset bias is inherently multi-componential (Fig.7.1). In particular, semantic and non-semantic information constitute its primary components; Non-semantic information—such as resolution, interpolation artifacts, and compression residues— dominate, *in some cases*, the dataset bias, while semantic information acts as a robust but weaker "backup" source of identifiability that surfaces when those non-semantic information are attenuated. The interaction is not strictly antagonistic; rather, the two operate hierarchically and complementarily, and in settings where non-semantic information is suppressed, semantic information can meaningfully contribute to dataset discrimination.

7.3.1 Evidence

The Experiment 2 results in Section 4.3 (Fig.4.7, 4.8) yield a coherent evidence chain about the relative roles of non-semantic and semantic components in dataset bias:

1. **Non-semantic information alone provides strong dataset identifiability:** On the "non-semantic" test split (246 labels of two datasets with no semantic overlap), a ResNet-18 trained to Name the Label only on the 27 overlapping semantic labels[1] but evaluated to Name the Dataset on the 246 non-semantic overlapping test split[2] still achieves approximately 96.10% Name-the-Dataset accuracy. This demonstrates that superficial, non-semantic information (e.g., resolution/interpolation fingerprints and compression residues) are sufficient as first-order discriminative signals.

2. **Suppressing dominant non-semantic shortcuts sharply degrades performance:** Altering preprocessing to Dataset 04—by inserting the intermediate downsampling stage (Tab.4.4) that weakens interpolation "watermarks" and related non-semantic signatures—causes the Name-the-Dataset accuracy on the corresponding 246 non-semantic test split to drop from 96.10% to about 73.06%, a decline of roughly 23 percentage points. This confirms the heavy reliance of the original model on those non-semantic information.

3. **Semantic information provides a measurable secondary uplift that depends on the strength of non-semantic shortcuts:** In Dataset 03, where dominant non-semantic information remain intact, adding semantic information of the 27 semantically overlapping labels only raises Name-the-Dataset accuracy from 96.10% to 97.75% (+1.65 pp), indicating that semantics play a marginal backup role when easy non-semantic signals are available. By contrast, under Dataset 04 preprocessing—where those non-semantic information is weakened—the same semantic information addition yields a much larger uplift, from 73.06% to 80.91% (+7.85 pp), demonstrating that latent semantic information becomes substantially more influential once first-order non-semantic information is attenuated.

4. **Primary vs. backup channel interpretation:** The pattern of high accuracy with intact non-semantic artifacts, steep degradation when they are suppressed, and partial recovery through semantics supports a hierarchical interpretation: non-semantic information acts as the principal (easy-to-exploit) driver of dataset separability, while semantic content functions as a secondary "backup" channel that becomes salient only after the leading shortcut is weakened.

[1] Note that since the ResNet-18 is trained on those 27 semantically overlapping labels, the model can use both semantic information and non-semantic information of the 27 labels to Name the Label or Name the Dataset

[2] Because the ResNet-18 backbone was trained only on the 27 semantically overlapping classes, when evaluated on the 246 non-overlapping labels it cannot rely on 27 Labels semantics; instead, it must depend solely on *dataset-wide* non-semantic information to infer provenance. Crucially, both the overlapping and non-overlapping labels originate from the same two source datasets (CIFAR-100 and TinyImageNet), so images from a given dataset share consistent non-semantic information (e.g., interpolation artifacts, compression residues, resolution-induced textures), which the model exploits to name the dataset despite the absence of semantics.

7.3.2 Fine-grained decomposition

The controlled preprocessing interventions reveal the concrete mechanisms by which non-semantic bias is instantiated:

- **Interpolation as implicit dataset "watermarks."** The two pipelines (Tab.4.4)—direct 4× upsampling (Dataset 03) versus an 8× effective interpolation path (downsample to 32×32 then upsample to 256×256, Dataset 04)—introduce systematically distinct artifact patterns. Bicubic interpolation, despite its ostensibly smooth output, imprints consistent textures (e.g., subtle ringing, frequency distortion, aliasing remnants) that persist through deep feature extraction and serve as reliable signals of provenance. The sharp degradation in non-semantic identification when these interpolation footprints are weakened indicates their primacy as first-order shortcuts.

- **Resolution and compression residues.** Discrepancies in native resolution and JPEG compression (even at high quality) may introduce low-level statistical regularities—blocking, quantization noise, and other acquisition/preprocessing-induced artifacts—that are stable within a source dataset and exploitable for dataset discrimination. These signals compound with interpolation effects to form a constellation of non-semantic information.

- **Hierarchical complementarity with semantic content.** The fact that semantic information contributes more noticeably only after dominant non-semantic information is suppressed (Dataset 04) refines the interpretation in [Liu and He, 2025], which discovered that dataset bias contain some generalizable and transferable semantic features. Rather than casting semantic and non-semantic information as conflicting, the data support a hierarchical interplay: non-semantic information dominate the "easy" separability dimensions, while latent semantic structure becomes an auxiliary discriminative source when those easy signals are degraded. This layered behavior is also complementary with broader syntheses of dataset bias resilience and transferability [Zeng et al., 2024], where multiple bias modalities coexist and surface under different conditioning.

7.3.3 Practical implications

- **Design explicit non-semantic OOD splits.** To diagnose and quantify dependence on dataset specific non-semantic information, benchmark suites should

include controlled splits that isolate non-semantic information (as in the non-semantic test sets), enabling separation of semantic generalization from exploitation of dataset specific fingerprints.

- **Audit preprocessing heterogeneity.** Pipelines should codify and monitor variations in the data acquisition and transformation stages (e.g., upsampling factors, interpolation kernels, compression quality), since such choices materially affect the dataset bias landscape. Reproducibility and datset bias mitigation may demand provenance tracking of these low-level signatures.
- **Joint modeling of semantic and non-semantic bias is necessary.** Interventions targeting one component (e.g., randomizing or smoothing interpolation artifacts to reduce non-semantic shortcuts) may inadvertently elevate the relative influence of the other. Effective mitigation thus may requires combined strategies that perturb, regularize, or disentangle both semantic and non-semantic information.

7.4 RQ3 – How Do Model Architectures Mediate Dataset Bias?

Conclusion: As discussed in RQ2 (Section 7.3), dataset bias, comprised primarily of semantic and non-semantic information, seems to propagate through a network in a layer–dependent fashion: *early* layers of both convolutional networks (CNNs) and Vision Transformers (ViTs) capture abundant *non-semantic* information (e.g., interpolation artifacts, compression traces) from input images, whereas *deeper* layers, through training, progressively transform a subset of these non-semantic information into higher-level, task-aligned semantic information: in CNNs via hierarchical composition with selective retention or compression, and in ViTs through global self-attention; neither architecture eliminates the non-semantic information completely. The trajectory and completeness of this transformation, however, are highly likely to be architecture–specific.

7.4.1 Evidence (Cross-layer Linear Probes)

In Experiment 3 (Section 4.4), linear probes were inserted after each residual block of ResNet-18 / ResNet-50 and after each transformer block of ViT-B/32, respectively measuring three types of test accuracies:

1. Name-the-Label (semantic classification),

2. Name-the-Dataset accuracy on 27 semantically overlapping labels[3],

3. Name-the-Dataset accuracy on 246 non-semantically overlapping classes (i.e., the non-semantic test; see Fig. 4.8), which can serve as a proxy for the strength of non-semantic bias signals in the representations.[4]

Figure 4.11 (the relevant probe performance plots in this manuscript) reveals the following key phenomena:

1. **Transformation of non-semantic signals after early peaks:** Across all three architectures, the Name-the-Dataset accuracy on non-semantically overlapping labels (246-label non-semantic test), a direct proxy for the strength of non-semantic information, attains its maximum at certain probe positions before the output layer and then decreases in deeper layers until the final layer. In contrast, the Name-the-Label accuracy and the semantically overlapping-label Name-the-Dataset accuracy (on 27 labels), which contain mixtures of semantic and non-semantic information, generally increase or exhibit rebound-like recoveries. The divergence of these trends—diminishing direct non-semantic information concurrent with rising task-aligned discriminability(Name-the-Label accuracy & Name-the-Dataset accuracy (on 27 semantically overlapping labels))—supports the hypothesis that non-semantic information is progressively transformed into semantic information: early layers strongly encode non-semantic information, and with increasing depth the non-semantic information is reconstructed, fused, or "purified," reducing Name-the-Dataset accuracy (on 246 non-semantically overlapping labels) while portions of the non-semantic information is re-expressed in higher-order semantics, as reflected by renewed dataset distinguishability on 27 overlapping labels in later stages.

2. **Pronounced oscillatory behavior in CNNs (large-amplitude fluctuations):** In ResNet-18 and ResNet-50, the two curves associated with dataset identification (the Name-the-Dataset accuracies for overlapping labels and non-semantic labels) exhibit substantial up-and-down fluctuations rather than a simple monotonic progression: initial probes (e.g., P1) already attain relatively high levels (around 60%), decline to local minima in intermediate layers (e.g., P3–P4), and then recover in

[3]Note that since the ResNet-18 is trained on those 27 semantically overlapping labels, the model can use both semantic information and non-semantic information of the 27 labels to Name the Label or Name the Dataset

[4]Because the ResNet-18 backbone was trained only on the 27 semantically overlapping classes, when evaluated on the 246 non-overlapping labels it cannot rely on 27 Labels semantics; instead, it must depend solely on *dataset-wide* non-semantic information to infer provenance. Crucially, both the overlapping and non-overlapping labels originate from the same two source datasets (CIFAR-100 and TinyImageNet), so images from a given dataset share consistent non-semantic information (e.g., interpolation artifacts, compression residues, resolution-induced textures), which the model exploits to name the dataset despite the absence of semantics.

deeper residual blocks (e.g., P7–P8), with a modest drop just before the final global average pooling. This indicates that residual blocks first integrate low-level non-semantic shortcuts into more task-aligned abstractions—temporarily diminishing the direct discriminability of the original non-semantic information to Name the Dataset—and may subsequently give rise to new semantic representations (some still retaining dataset-bias shortcuts) that are further processed or compressed in later stages.

3. **Smooth parallel accumulation with mild late attenuation in ViT:** Although ViT-B/32, like the ResNets, exhibits a peak in the non-semantic Name-the-Dataset accuracy (246-label test)—reaching about 65.8% at probe 8 and then gently declining to $\approx 64.4\%$ toward the final layers—its overall probe trajectories remain much smoother and far less oscillatory than in ResNet. The small downturn after probe 8 likely reflects subtle transformation or compression of non-semantic information, but unlike the pronounced peak–valley dynamics in CNNs, there are no sharp local troughs; this pattern suggests that global self-attention incrementally entangles and refines semantic and non-semantic information in parallel, without first transforming the non-semantic one in order to reconstruct it later.

7.4.2 Architectural Mechanism Analysis

The behaviors described above reflect intrinsic information flow mechanisms of different architectures in representation processing:

1. **Hierarchical purification and reconstruction in ResNet:** Convolutional networks tend to progressively abstract input information via local receptive fields and residual pathways [Luo et al., 2016]. Non-semantic information learned in early layers (e.g., interpolation artifacts, compression residues) is integrated into more task-aligned abstractions in deeper residual blocks, yielding abrupt improvements in semantic classification performance; this integration impairs the original non-semantic discriminability, manifesting as local dips of Name-the-Dataset Accuracy (246 non-semantically overlapping labels). As depth increases further, higher-order structures can transform part of the non-semantic information into semantic dataset-related information , producing phased fluctuations. Deeper networks (e.g., ResNet-50) provide additional processing stages, thereby increasing the opportunity to dilute the direct influence of non-semantic information before the final output and to enhance semantic purity.

2. **Persistent entanglement and global integration in ViT:** Vision Transformers, built on self-attention, enable interactions among all spatial tokens from the

outset, yielding globally contextualized representations [Raghu et al., 2021]. Semantic and non-semantic signals are incrementally and jointly reinforced across blocks without an explicit separation or cleansing stage, so both types of information persistently coexist throughout the depth. This pervasive global fusion complicates lossless removal of dataset-bias shortcuts in the absence of additional regularization, since the entanglement makes isolating or suppressing any the two types of the information much more difficult.

3. **External validity and benchmarking:** The waveform structure revealed by the cross-layer linear probes reproduces and extends two classes of phenomena documented in prior work. On one hand, it is complementary with the non-monotonic bias–capacity interaction described for convolutional networks in Hall et al. [2022], whereby the modulation of bias-strength varies with capacity and depth in a non-trivial manner. On the other hand, the smooth inter-layer coherence observed in ViT reinforces the representation consistency reported for transformers in Raghu et al. [2021], thereby bolstering the external validity and generality of the present findings.

7.4.2.1 Practical implications

- **Debiasing strategies should be architecture-aware:** Convolutional architectures naturally perform partial purification of non-semantic shortcuts through hierarchical composition; debiasing methods can exploit their stagewise antagonistic dynamics (e.g., by injecting auxiliary losses around intermediate residual blocks). In contrast, transformers maintain semantic and non-semantic information concurrently over extended depths, necessitating cross-layer or global constraints—such as attention regularization [Attanasio et al., 2022], or explicit disentanglement mechanisms [Bhatt et al., 2023]—to achieve comparable attenuation of dataset bias.

- **Layer-wise diagnosis and intervention:** Cross-layer linear probes [Alain and Bengio, 2016] provide effective localization of prominent accumulation points of dataset bias. For ResNets, examining the dynamics of non-semantic transformation around local troughs enables targeted insertion of disentangling regularization at critical stages. For ViTs, distributed, low-intensity interventions spanning multiple transformer blocks are required to gradually weaken the persistent co-occurrence of shortcut signals.

- **Model selection should consider bias propagation pathways:** In deployment contexts characterized by data-source shift or demanding robustness, preference should be given to architectures that either inherently dilute non-semantic

bias in deeper layers (e.g., deeper ResNet variants) or are amenable to embedding disentanglement modules in their representations (e.g., structured transformer variants), so as to synergize with pre- or mid-training bias mitigation mechanisms.

- **Synergistic data–model design:** The results suggest that interventions limited to either the data or model side alone are insufficient. Integrating architectural priors (e.g., local–global mixing, texture–shape disentanglement) with explicit training-time debiasing methods (such as Uniform Double Mini) can foster more bias-robust representations without significant sacrifice in semantic performance.

7.5 RQ4 – What is Dataset Bias? How Can Dataset Bias Be Mitigated? What Challenges Remain?

Conclusion: Dataset bias refers to the fact that any finite dataset is merely a sampled approximation of the real world, so that the finite dataset distribution $P_{dataset}$ diverges from the true real world distribution P_{real}. This distributional discrepancy can induce systematic deviations in the model's outputs relative to real-world behavior.

Mitigating such systematic errors requires interventions along three complementary dimensions:

- **Dataset:** improve representativeness through better sampling, importance reweighting, augmentation, or additional data collection to reduce the gap between $P_{dataset}$ and P_{real}.
- **Architecture:** incorporate inductive biases or structural mechanisms that both mitigate reliance on dataset bias shortcuts (e.g., through invariant representations, domain-adaptive components, or adversarial regularization) and, when feasible, disentangle and transform non-semantic information contained in dataset bias into useful semantic information (Section 7.4).
- **Training procedure:** employ loss modification, regularization, robust or adversarial optimization, and calibration techniques that explicitly account for distribution shift between dataset and the real world.

A coordinated application of these strategies increases the likelihood that learned models will produce outputs that faithfully reflect the underlying real-world distribution.

7.5.1 Dataset Bias Definition

7.5.1.1 Origin of Dataset Bias

Let $P_{\text{real}}(\boldsymbol{x}, \boldsymbol{y})$ denote the true joint distribution over images $\boldsymbol{x}$ and labels $\boldsymbol{y}$ in the real visual world, and let $P_{dataset}(\boldsymbol{x}, \boldsymbol{y})$ denote the joint distribution induced by sampling under dataset source d. Dataset bias associated with d is defined as

$$\text{DatasetBias}_d \;:=\; \mathscr{D}\big(P_{dataset}(\boldsymbol{x}, \boldsymbol{y}) \,\|\, P_{\text{real}}(\boldsymbol{x}, \boldsymbol{y})\big),$$

where

- $\boldsymbol{x}$: the observed input image sample, generated as $\phi(\boldsymbol{g}, \boldsymbol{b})$ from generalizable features $\boldsymbol{g}$ and dataset-bias shortcuts $\boldsymbol{b}$.
- $\boldsymbol{y}$: the true semantic label associated with $\boldsymbol{x}$, typically represented as a one-hot vector.
- d: the dataset source identifier, encoding the particular sampling protocol or origin that induces a biased empirical distribution.
- $P_{\text{real}}(\boldsymbol{x}, \boldsymbol{y})$: the (unknown) true joint distribution over images and labels in the real world.
- $P_{dataset}(\boldsymbol{x}, \boldsymbol{y})$: the joint distribution over $\boldsymbol{x}$ and $\boldsymbol{y}$ resulting from sampling according to dataset source d; this is the distribution observed during training.
- DatasetBias_d: the quantified bias of dataset d, defined as the divergence between the sampled distribution $P_{dataset}$ and the real-world distribution P_{real}.
- $\mathscr{D}(\cdot\|\cdot)$: a nonnegative divergence or distance measure between probability distributions (e.g., KL divergence $D_{\text{KL}}(P\|Q)$, total variation $\frac{1}{2}\|P-Q\|_1$, or Wasserstein $W_p(P, Q)$).

7.5.1.2 Systematic Model Deviation

A model trained on the biased dataset d produces predictions

$$\hat{\boldsymbol{y}} = \text{softmax}(f_d(\boldsymbol{x})),$$

- $f_d(\boldsymbol{x})$: the model trained on dataset d, mapping $\boldsymbol{x}$ to pre-softmax logits; the subscript d emphasizes dependence on the sampled (biased) training set distribution.

- softmax$(\cdot)$: the normalization that converts logits into a probability vector over labels.
- $\hat{\boldsymbol{y}}$: the model's predicted label distribution.

The expected loss under the real-world distribution versus the training (biased) distribution is

$$\mathscr{L}_{\text{real}}(f_d) := \mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})\sim P_{\text{real}}}\left[\ell(\text{softmax}(f_d(\boldsymbol{x})), \boldsymbol{y})\right], \quad \mathscr{L}_{\text{biased}}(f_d) := \mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})\sim P_{dataset}}\left[\ell(\text{softmax}(f_d(\boldsymbol{x})), \boldsymbol{y})\right].$$

- $\ell(\hat{\boldsymbol{y}}, \boldsymbol{y})$: the loss function (e.g., cross-entropy) measuring discrepancy between prediction and ground-truth label.
- $\mathscr{L}_{\text{real}}(f_d)$: expected risk of f_d when evaluated on real-world data.
- $\mathscr{L}_{\text{biased}}(f_d)$: expected risk of f_d on the biased training distribution.

The systematic deviation (generalization gap) induced by dataset bias is

$$\Delta_{\text{sys}}(f_d) := \mathscr{L}_{\text{real}}(f_d) - \mathscr{L}_{\text{biased}}(f_d),$$

Equivalently, the point-wise bias in the predictive distribution is

$$\text{Bias}_{\text{output}}(\boldsymbol{x}) := \text{softmax}(f_d(\boldsymbol{x})) - P_{\text{real}}(\boldsymbol{y} \mid \boldsymbol{x}),$$

- $P_{\text{real}}(\boldsymbol{y} \mid \boldsymbol{x})$: the true conditional distribution over labels given input $\boldsymbol{x}$ in the real world.
- $\text{Bias}_{\text{output}}(\boldsymbol{x})$: the discrepancy between the model's predicted label distribution and the true label distribution for a specific input, capturing pointwise systematic error induced by dataset bias.

Point-wise bias example: Suppose a binary classification task (dog vs. cat). For a given input $\boldsymbol{x}$, the true conditional label distribution in the real world is

$$P_{\text{real}}(\boldsymbol{y} \mid \boldsymbol{x}) = [0.9,\ 0.1],$$

meaning that the correct label is "dog" with probability 0.9. The model trained on the biased dataset predicts

$$\text{softmax}(f_d(\boldsymbol{x})) = [0.6,\ 0.4].$$

The pointwise output bias is then

$$\text{Bias}_{\text{output}}(\boldsymbol{x}) = \text{softmax}(f_d(\boldsymbol{x})) - P_{\text{real}}(\boldsymbol{y} \mid \boldsymbol{x}) = [0.6,\, 0.4] - [0.9,\, 0.1] = [-0.3,\, +0.3].$$

This vector indicates that the model underestimates its confidence in "dog" by 0.3 and overestimates its confidence in "cat" by 0.3, representing a pointwise systematic deviation in the predictive distribution.

7.5.2 Mitigation: A Three–pronged Landscape

Mitigation strategies can be grouped into complementary, yet often orthogonal, dimensions:

(i) *Data–centred interventions*: Data-centred mitigation approaches primarily focus on changing the input distribution to reduce inherent bias from datasets. Rebalancing (Li and Vasconcelos 2019), maximum–entropy re-weighting [Celis et al., 2020], targeted sim–to–real augmentation [Jaipuria et al., 2020], low-biased generative synthesis [Jiang et al., 2024], and large-scale web expansion [Sevetlidis et al., 2022] all attempt to narrow the gap $D_{\text{KL}}(P_{dataset} \,\|\, P_{\text{real}})$ *before* learning begins.

(ii) *Architecture–centred interventions*: Bias-aware inductive priors include attention geometry constraints [Talebpour et al., 2025], hybrid scale-routing [Herranz et al., 2016], and explicit disentanglement blocks [Bassi et al., 2024]. Section 7.4 shows that ResNet hierarchies seems to naturally attenuate a subset of non-semantic dataset bias shortcuts, whereas Vision Transformers also attenuate such shortcuts only minimally—virtually not at all—through their global self-attention—highlighting the need for architecture-specific design.

(iii) *Training-centred interventions*: Classical formulations suppress bias by adversarially minimising the mutual information between dataset bias and model output $I\big(b(X); f(X)\big)$ [Ganin et al., 2016, Kim et al., 2019]. Other lines include MMD alignment [Tzeng et al., 2014], functional–entropy regularisation [Gat et al., 2021], and fairness-oriented gradient projection [Zhang et al., 2018]. The present work contributes two variants:

1. *One-hot MiniMax (OHM)*, an explicit gradient-reversal formulation that maximises Name the Dateset loss while minimising Name the Label loss (Eq. 6.3.4); and

2. *Uniform Double Mini (UDM)*, which replaces the maximization of Name the Dataset loss with a minimization objective that aligns model's Name the Dataset output logits with the uniform distribution prior (Eq. 6.3.14).

Both reduce to $\approx 65\%$ Name-the-Dataset accuracy (vs. 81% baseline) while *raising* Name-the-Label accuracy by around 1–2 pp (Section 6.4.3), demonstrating that dataset bias attenuation and main task objective need not be antagonistic.

7.5.3 Challenges That Persist

1. Semantic *vs.* non-semantic disentanglement: Dataset bias is a *mixture* (Fig.7.1): transferable semantic patterns [Liu and He, 2025, Zeng et al., 2024] *plus* harmful non-semantic shortcuts (Fig.4.8). Both One-hot MiniMax and Uniform Double Mini reduce the *aggregate* dataset bias; however, useful semantics are suppressed alongside non-semantic shortcuts. Decoupling these components—removing only non-semantic shortcuts while *retaining* generalizable semantics—is an open optimization problem, probably requiring fine-grained latent modelling.

2. Synthetic Data Limitations: Generative models trained on biased datasets inherit and amplify biases [Fabbrizzi et al., 2022, Mehrabi et al., 2021, Zeng et al., 2024], and evaluation often neglects O.O.D. performance [Jaipuria et al., 2020]. Without unbiased priors, synthetic augmentation may perpetuate rather than eliminate bias.

3. Objective conflicts and over-correction: Most methods target a single bias factor (colour, gender, resolution) [Merler et al., 2019, Panda et al., 2018], neglecting interactions and risking amplification elsewhere. Balancing fairness, robustness, and task accuracy [Gat et al., 2021] calls for mature multi-objective optimisation. Excessive adversarial weighting (λ) can erase task-relevant signal, while insufficient weighting leaves residual bias.

4. Multi-modality and Temporal Bias: Vision–language models exhibit coupled biases across modalities [Zeng et al., 2024], and streaming or continuously updated datasets introduce evolving biases that static methods cannot address. Extensions to multimodal and updated data streams remain largely unexplored.

7.6 Limitations of the Study

Although the present study provides comprehensive insights into the persistence, composition and mitigation of dataset bias in modern deep networks, several limitations constrain the generality and scope of its conclusions:

1. **Image resolution and sensor variety:** All experiments employed relatively low–resolution datasets (CIFAR-100 at 32×32 and TinyImageNet at 64×64), precluding analysis of bias phenomena arising in high-resolution imaging or professional sensor pipelines. This limitation may restrict the applicability of our findings to domains such as medical imaging or autonomous driving, where acquisition artifacts and noise characteristics differ substantially from those in consumer photographs.

2. **Model capacity and architectural diversity:** All experiments focused on ResNet-18, ResNet-50 and ViT-B/32 (Experiments 1-6). Due to computational constraints, deeper and more recent architectures (e.g. ViT-Large, ConvNeXt-XL, hybrid CNN–Transformer models) were not evaluated. Consequently, the extent to which the observed cross-layer bias propagation patterns (Section 7.4) and the efficacy of Uniform Double Mini scale to very high-capacity models remains an open question.

3. **Limited Spectrum of Bias Types:** While interpolation footprints was systematically manipulated (Section 4.5), other critical bias sources—including illumination conditions (indoor vs. outdoor, HDR scenarios), temporal factors (time-of-day, seasonal variations), and demographic/socioeconomic biases (e.g., skin-tone imbalances)—were not explicitly investigated. Consequently, the mitigation efficacy of Uniform Double Mini on these prevalent real-world distortions remains unevaluated.

4. **Static, single-modality evaluation:** The study considered bias mitigation within static, pre-collected datasets. The dynamics of bias evolution in streaming or continually updated data sources—where both $P_{dataset}$ and the latent biases $\mathbf{b}$ may shift over time—were not addressed. Moreover, multimodal biases in vision–language models were beyond the scope of this work.

Addressing these limitations in future research—by incorporating higher-resolution and professional imaging pipelines, expanding the set of bias modalities, developing more precise latent-bias estimators, evaluating a broader range of architectures, and studying temporal bias dynamics—will be critical to fully understand and mitigate dataset bias in real-world applications.

Chapter 8

Conclusion and Recommendations

8.1 Synthesis of Findings

This dissertation set out to re-examine *dataset bias* in modern computer-vision pipelines, reproduce and extend the classical "Name That Dataset" paradigm of Torralba and Efros [2011], and propose mitigation strategies grounded in a unified mathematical framework (Chapter 5). Across six controlled experiments (Chapters 4–6) the following conclusions were established:

1. **Persistence of Bias.** State-of-the-art architectures—ResNets and Vision Transformers—retain highly separable dataset fingerprints; a single frozen linear probe trained on top of a semantic backbone reaches $\approx 98\%$ Name-the-Dataset accuracy (Section 7.2).

2. **Composition of Bias.** Dataset bias is primarily composed (Fig. 7.1) of non-semantic information and semantic information. In Name the Dataset tasks of Experiments 1-6 of this study *non-semantic information* dominates first-order Name the Dataset separability, while *semantic information* emerge as a back-up channel once the non-semantic shortcuts are suppressed (Section 7.3.1).

3. **Architecture-Mediated Propagation.** Cross-layer probes (Fig. 4.11) reveal that, presumably, ResNets are capable of "purifying" part of non-semantic shortcuts and then reconstructing them to be higher-level semantics, whereas Vision Transformers—although capable of forming advanced semantic representations—appear comparatively less proficient at this purification-and-reconstruction process, instead maintaining a smoother entanglement of semantic and non-semantic information across their depth (Section 7.4).

4. **Effective Bias Mitigation without Accuracy Loss.** Two training-centred methods were introduced: *One-Hot MiniMax* and *Uniform Double Mini*. The latter reduces Name-the-Dataset accuracy from 81% to ≈ 65% while *raising* Name-the-Label accuracy by 1–2 pp on both ResNet-18 and ViT-B/32 (Section 6.4, 6.5). Hence bias attenuation and task utility are not inherently antagonistic.

8.2 Practical Implications

- **Bias diagnostics should be routine.** Linear-probe audits similar to Experiment 1 offer a low-cost check for hidden shortcuts before deployment, complementing fairness and robustness evaluations.
- **Pre-processing choices matter.** Seemingly innocuous operations—e.g. up-sampling kernels or JPEG quality—leave persistent "watermarks." Ensuring that all data sources share identical pipelines mitigates first-order shortcuts.
- **Architecture-aware debiasing.** Residual networks—which can "purify" early non-semantic shortcuts and then rebuild them into higher-level semantic features; in contrast, Vision Transformers—which maintain a smoother depth-wise entanglement of semantic and non-semantic information. Consequently, bias mitigation strategies should be tailored to match the inductive biases of each architecture.
- **Loss design over "bigger models."** Simply scaling parameters is insufficient. Joint loss formulations that enforce distributional uniformity (Eq. 6.3.14) offer a parameter-free lever to curb shortcut learning in large backbones.

8.3 Recommended Future Work

1. **Semantic vs. non-semantic separation.** Design latent-factor models or contrastive objectives that *retain* transferable semantics from dataset bias while discarding non-semantic dataset bias shortcuts, addressing the over-correction risk noted in Section 7.5.3.
2. **Multimodal bias disentanglement.** Investigate joint vision–language models to quantify and mitigate cross-modal bias.
3. **Temporal bias dynamics.** Develop streaming benchmarks that reveal how bias evolves as new data accumulate; incorporate continual-learning regularisers to counter drift.

4. **Synthetic data validation.** Formulate O.O.D. evaluation protocols for generative augmentation pipelines [Jiang et al., 2024] to prevent hidden synthetic-induced bias inheritance before large-scale adoption.

8.4 Concluding Remarks

Dataset bias arises because any dataset constitutes a finite, biased sample of the real world, leading to degraded generalization performance manifested as systematic deviations of model outputs from reality. Although some works advocate the construction of larger and more diverse datasets to alleviate bias [Liu and He, 2025, Torralba and Efros, 2011], it is well understood that model behavior is jointly determined by (1) dataset composition and preprocessing, (2) model architecture, and (3) training methodology. Effective correction of systematic deviations induced by dataset bias therefore demands a co-design paradigm across these three dimensions. Furthermore, not all components of dataset bias require suppression; only non-semantic information that cannot be translated into transferable and generalizable semantic content should be mitigated, while genuine semantic information ought to be preserved. It is noteworthy that, as training progresses and model depth increases, a subset of non-semantic information may gradually transform into semantic information. By co-designing dataset composition, model architecture, and training methodology, and by selectively suppressing non-semantic components while preserving meaningful semantic information contained in dataset bias, this holistic framework enables the development of robust, fair, and generalizable neural networks capable of operating effectively beyond their original training domains.

Appendix A

Summary of Datasets 03 - 06

In the following page, there is a table summarizing all the construction details of all the datasets incorporated in the dissertation.

Feature		**Dataset03**	**Dataset04**	**Dataset05**	**Dataset06**
Preprocessing	TinyImageNet (64×64)	64×64 → 256×256 (Bicubic) → 224×224 (Random Crop)	64×64 → 32×32 (Bicubic) → 256×256 (Bicubic) → 224×224 (Random Crop)	64×64 → 256×256 (Bicubic) → 224×224 (Random Crop)	64×64 → 32×32 (Bicubic) → 256×256 (Bicubic) → 224×224 (Random Crop)
	CIFAR-100 (32×32)	32×32 → 256×256 (Bicubic) → 224×224 (Random Crop)			
Training Class Composition		**27 classes** (Semantically overlapping)		**273 classes total:** - 27 overlapping - 73 CIFAR-only - 173 Tiny-only	
Image Format		JPEG with quality setting = 95			
Data Split - Test		**27 classes Normal Test:** - CIFAR-100 test (overlap) - TinyImageNet val (overlap) **246 classes Non-semantic test:** - 73 classes: CIFAR-only test - 173 classes: Tiny-only val		**Test Set of Three groups:** 1. 27 classes: CIFAR-100 test + Tiny val (overlap) 2. 73 classes: CIFAR-only test 3. 173 classes: Tiny-only val	
Data Split - Train/Val		90%/10% split of: 1. 27 classes CIFAR-100 train (overlap) 2. 27 classes TinyImageNet train (overlap)		90%/10% split of **three groups:** 1. 27 classes: CIFAR+Tiny train (overlap) 2. 73 classes: CIFAR-only train 3. 173 classes: Tiny-only train	
Key Characteristics		- Pure overlapping classes - Single preprocessing path - Unified test set	- Pure overlapping classes - **Extra downscaling** for TinyImageNet - Unified test set	- **Mixed class groups** - Separate non-overlapping tests - Full class utilization	- **Mixed class groups** - **Extra downscaling** for TinyImageNet - Full class utilization

Note: The "224×224 (Random Crop)" step is actually applied as a preprocessing augmentation before each training epoch; in fact, the dataset images are stored at 256×256.

TABLE A.1: Dataset 03-06 Comparison Table

Appendix B

Train & Test Log for Models

This Appendix B incorporates some models' train & test log.

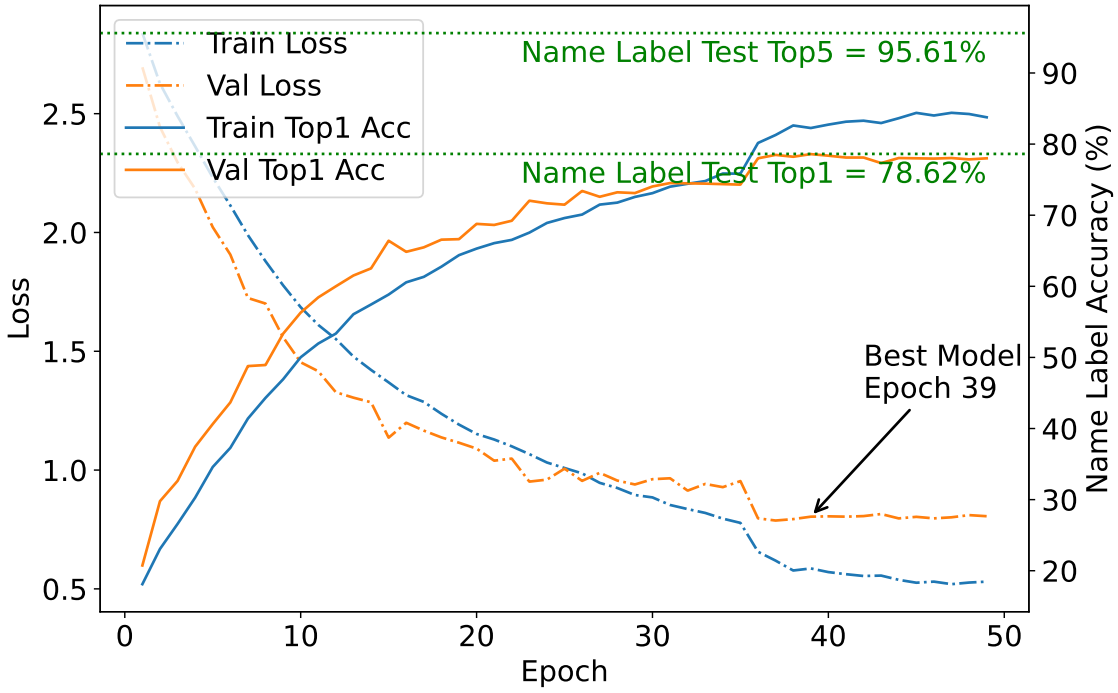


FIGURE B.1: Train & Test Log of A ResNet-18 Model Trained to Name 27 Label without Dataset Bias Mitigation Method Following Protocol A (Section 3.5.2) on Dataset 04 (Tab.A.1)

Appendix C

Linear Probe Heatmaps

The following two heatmaps illustrate the per-class "Name the Dataset" test accuracy of ResNet18 trained on dataset 03,04 across different probe locations. The x-axis corresponds to the semantic class indices (class 0–class26), while the y-axis represents the probe positions along the network. Each cell is annotated with the exact accuracy value (in %), and the colour intensity reflects the magnitude according to the side colormap. Darker shades indicate higher classification accuracy. The heatmap reveals variations in bias-exploiting capability across probes and classes, highlighting that certain layers exhibit markedly higher per-class accuracies, potentially indicating stronger sensitivity to dataset-specific shortcuts.

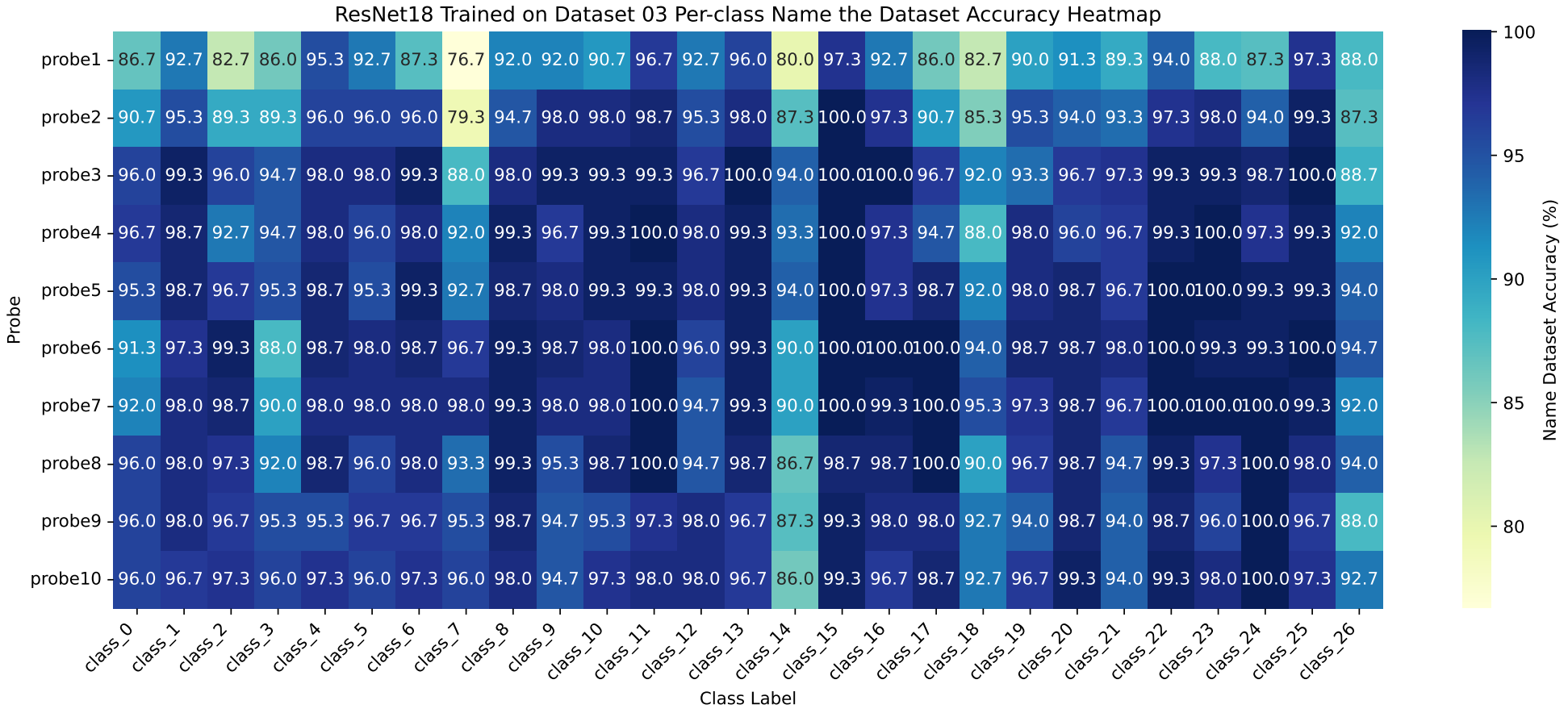


FIGURE C.1: Cross Layer Linear Probing Name Dataset Per-Class(27 semantically overlapping classes (labels)) Test Accuracies of ResNet-18 Pre-Trained with Protocol A (Section 3.5.2) and then Linear Probed with Protocol C (Section 3.5.4) on Dataset 03

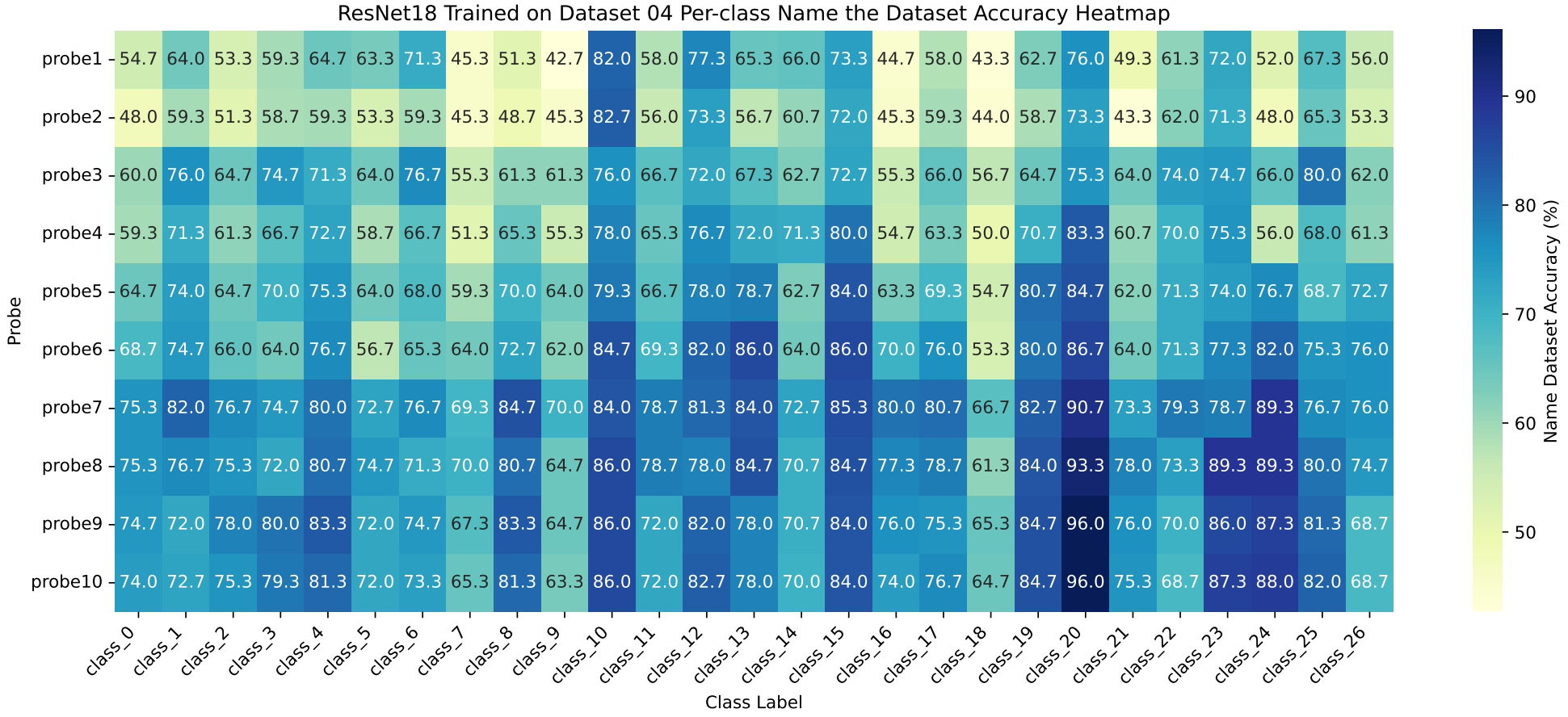


FIGURE C.2: Cross Layer Linear Probing Name Dataset Per-Class(27 semantically overlapping classes (labels)) Test Accuracies of ResNet-18 Pre-Trained with Protocol A (Section 3.5.2) and then Linear Probed with Protocol C (Section 3.5.4) on Dataset 04

ID	Class Name	TinyImageNet (ID / Name)	CIFAR-100 (ID / Name)
0	goldfish	n01443537 / goldfish	1 / aquarium_fish
1	brown bear	n02132136 / brown bear	3 / bear
2	bee	n02206856 / bee	6 / bee
3	lady beetle	n02165456 / ladybug	7 / beetle
4	pop bottle	n03983396 / pop bottle	9 / bottle
5	suspension bridge	n04366367 / suspension bridge	12 / bridge
6	school bus	n04146614 / school bus	13 / bus
7	monarch butterfly	n02279972 / monarch	14 / butterfly
8	Arabian camel	n02437312 / Arabian camel	15 / camel
9	ox	n02403003 / ox	19 / cattle
10	rocking chair	n04099969 / rocking chair	20 / chair
11	chimpanzee	n02481823 / chimpanzee	21 / chimpanzee
12	cockroach	n02233338 / cockroach	24 / cockroach
13	African elephant	n02504458 / African elephant	31 / elephant
14	computer keyboard	n03085013 / computer keyboard	39 / keyboard
15	lawn mower	n03649909 / lawn mower	41 / lawn_mower
16	lion	n02129165 / lion	43 / lion
17	American lobster	n01983481 / American lobster	45 / lobster
18	mushroom	n07734744 / mushroom	51 / mushroom
19	orange	n07747607 / orange	53 / orange
20	plate	n07579787 / plate	61 / plate
21	snail	n01944390 / snail	77 / snail
22	dining table	n03201208 / dining table	84 / table
23	pay-phone	n03902125 / pay-phone	86 / telephone
24	tractor	n04465501 / tractor	89 / tractor
25	bullet train	n02917067 / bullet train	90 / train
26	bell pepper	n07720875 / bell pepper	83 / sweet_pepper

TABLE C.1: List of the 27 semantic labels (classes) used in the experiments, with their TinyImageNet and CIFAR-100 mappings.

Bibliography

Alain, G. and Bengio, Y. (2016). Understanding intermediate layers using linear classifier probes. *arXiv preprint arXiv:1610.01644*.

Attanasio, G., Nozza, D., Hovy, D., and Baralis, E. (2022). Entropy-based attention regularization frees unintended bias mitigation from lists. *arXiv preprint arXiv:2203.09192*.

Bardes, A., Ponce, J., and LeCun, Y. (2021). Vicreg: Variance-invariance-covariance regularization for self-supervised learning. *arXiv preprint arXiv:2105.04906*.

Bassi, P. R. A. S., Cavalli, A., and Decherchi, S. (2024). Explanation is all you need in distillation: Mitigating bias and shortcut learning. *arXiv preprint*. Under review.

Bhatt, G., Das, D., Sigal, L., and N Balasubramanian, V. (2023). Mitigating the effect of incidental correlations on part-based learning. *Advances in Neural Information Processing Systems*, 36:63738–63757.

Celis, L. E., Keswani, V., and Vishnoi, N. K. (2020). Data preprocessing to mitigate bias: A maximum entropy based approach. In *International Conference on Machine Learning*, pages 1395–1405. PMLR.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al. (2020). An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*.

d'Ascoli, S., Touvron, H., Leavitt, M. L., Morcos, A. S., Biroli, G., and Sagun, L. (2021). Convit: Improving vision transformers with soft convolutional inductive biases. In *International conference on machine learning*, pages 2286–2296. PMLR.

Fabbrizzi, S., Papadopoulos, S., Ntoutsi, E., and Kompatsiaris, I. (2022). A survey on bias in visual datasets. *Computer Vision and Image Understanding*, 223:103552.

Fang, C., Xu, Y., and Rockmore, D. N. (2013). Unbiased metric learning: On the utilization of multiple datasets and web images for softening bias. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*.

Ganin, Y., Ustinova, E., Ajakan, H., Germain, P., Larochelle, H., Laviolette, F., March, M., and Lempitsky, V. (2016). Domain-adversarial training of neural networks. *Journal of machine learning research*, 17(59):1–35.

Gat, O., Shalev-Schwartz, S., Angelov, P., and Chechik, G. (2021). Removing bias in multi-modal classifiers by regularizing functional entropies. In *Advances in Neural Information Processing Systems*.

Ge, S., Mishra, S., Li, C.-L., Wang, H., and Jacobs, D. (2021). Robust contrastive learning using negative samples with diminished semantics. *Advances in Neural Information Processing Systems*, 34:27356–27368.

Geirhos, R., Jacobsen, J.-H., Michaelis, C., Zemel, R., Brendel, W., Bethge, M., and Wichmann, F. A. (2020). Shortcut learning in deep neural networks. *Nature Machine Intelligence*, 2(11):665–673.

Geirhos, R., Rubisch, P., Michaelis, C., Bethge, M., Wichmann, F. A., and Brendel, W. (2019). Imagenet-trained cnns are biased towards texture; increasing shape bias improves accuracy and robustness. In *International Conference on Learning Representations (ICLR)*.

Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep Learning*. MIT Press.

Hall, M., van der Maaten, L., Gustafson, L., Jones, M., and Adcock, A. (2022). A systematic study of bias amplification. *arXiv preprint arXiv:2201.11706*.

He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

Hermann, K., Chen, T., and Kornblith, S. (2020). The origins and prevalence of texture bias in convolutional neural networks. In *Neural Information Processing Systems (NeurIPS)*.

Herranz, L., Jiang, S., and Li, X. (2016). Scene recognition with cnns: Objects, scales and dataset bias. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Hosseini, H. and Poovendran, R. (2018). Semantic adversarial examples. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 1614–1619.

Ilyas, A., Santurkar, S., Tsipras, D., Engstrom, L., Tran, B., and Madry, A. (2019). Adversarial examples are not bugs, they are features. *Advances in neural information processing systems*, 32.

Jackson, P. T., Bonner, S., Jia, N., Holder, C., Stonehouse, J., and Obara, B. (2021). Camera bias in a fine grained classification task. In *2021 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE.

Jaipuria, N., Zhang, X., Bhasin, R., Arafa, M., Chakravarty, P., Shrivastava, S., Manglani, S., and Murali, V. N. (2020). Deflating dataset bias using synthetic data augmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 824–825.

Jiang, D., Wang, H., Zhang, L., Wei, W., Dai, G., Wang, M., Wang, J., and Zhang, Y. (2024). Low-biased general annotated dataset generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Joshi, A., Mukherjee, A., Sarkar, S., and Hegde, C. (2019). Semantic adversarial attacks: Parametric transformations that fool deep classifiers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 4773–4783.

Keys, R. (2003). Cubic convolution interpolation for digital image processing. *IEEE transactions on acoustics, speech, and signal processing*, 29(6):1153–1160.

Khosla, A., Zhou, T., Malisiewicz, T., Efros, A. A., and Torralba, A. (2012). Undoing the damage of dataset bias. In *Proceedings of the European Conference on Computer Vision (ECCV)*.

Kim, B., Kim, H., Kim, K., Kim, S., and Kim, J. (2019). Learning not to learn: Training deep neural networks with biased data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Kingma, D. P. and Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Krizhevsky, A. and Hinton, G. (2009). Learning multiple layers of features from tiny images. Technical Report TR-2009, University of Toronto, Toronto, ON, Canada.

Li, D., Yang, Y., Song, Y.-Z., and Hospedales, T. M. (2017). Deeper, broader and artier domain generalization. In *Proceedings of the IEEE international conference on computer vision*, pages 5542–5550.

Li, F.-F., Karpathy, A., and Johnson, J. (2015). Tiny imagenet visual recognition challenge. https://tinyimagenet.com/. Accessed: 2025-08-10.

Li, Y. and Vasconcelos, N. (2019). Repair: Removing representation bias by dataset resampling. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9572–9581.

Liu, J., Shen, Z., He, Y., Zhang, X., Xu, R., Yu, H., and Cui, P. (2021). Towards out-of-distribution generalization: A survey. *arXiv preprint arXiv:2108.13624*.

Liu, Z. and He, K. (2025). A decade's battle on dataset bias: Are we there yet? *Meta AI Research, FAIR*.

Loshchilov, I. and Hutter, F. (2016). Sgdr: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983*.

Loshchilov, I. and Hutter, F. (2017). Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

Luo, W., Li, Y., Urtasun, R., and Zemel, R. (2016). Understanding the effective receptive field in deep convolutional neural networks. *Advances in neural information processing systems*, 29.

Mehrabi, N., Morstatter, F., Saxena, N., Lerman, K., and Galstyan, A. (2021). A survey on bias and fairness in machine learning. *ACM Computing Surveys*, 54(6):1–35.

Merler, M., Ratha, N., Ferrara, M., Willis, C., and Smith, J. R. (2019). Diversity in faces. *arXiv preprint arXiv:1901.10436*.

Panda, R. P., Palaniappan, K., and Berg, T. (2018). Contemplating visual emotions. In *Proceedings of the European Conference on Computer Vision*, pages 1–17.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al. (2019). Pytorch: An imperative style, high-performance deep learning library.

Prechelt, L. (1998). Early stopping – but when? In *Neural Networks: Tricks of the Trade*, pages 55–69. Springer.

Raghu, M., Unterthiner, T., Kornblith, S., Zhang, C., and Dosovitskiy, A. (2021). Do vision transformers see like convolutional neural networks? In *Advances in Neural Information Processing Systems (NeurIPS)*.

Sakaridis, C., Dai, D., and Gool, L. V. (2019). Guided curriculum model adaptation and uncertainty-aware evaluation for semantic nighttime image segmentation. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 7374–7383.

Sevetlidis, V., Pavlidis, G., Mouroutsos, S., and Gasteratos, A. (2022). Tackling dataset bias with an automated collection of real-world samples. *IEEE Access*, 10:126832–126844.

Talebpour, M. et al. (2025). Bias in language models: Interplay of architecture and data. In *Proceedings of the ACM SIGIR Conference*.

Tommasi, T., Patricia, N., Caputo, B., and Tuytelaars, T. (2017). A deeper look at dataset bias. In *Domain Adaptation in Computer Vision Applications*, pages 37–55. Springer, Cham.

Torralba, A. and Efros, A. A. (2011). Unbiased look at dataset bias. In *CVPR 2011*, pages 1521–1528. IEEE.

Turkheimer, E. (2000). Three laws of behavior genetics and what they mean. *Current directions in psychological science*, 9(5):160–164.

Tzeng, E., Hoffman, J., Zhang, N., Saenko, K., and Darrell, T. (2014). Deep domain confusion: Maximizing for domain invariance. *arXiv preprint arXiv:1412.3474*.

Wang, S., Veldhuis, R., Brune, C., and Strisciuglio, N. (2023). A survey on the robustness of computer vision models against common corruptions. *arXiv preprint arXiv:2305.06024*.

Yu, F., Chen, H., Wang, X., Xian, W., Chen, Y., Liu, F., Madhavan, V., and Darrell, T. (2020). Bdd100k: A diverse driving dataset for heterogeneous multitask learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2636–2645.

Zeng, B., Yin, Y., and Liu, Z. (2024). Understanding bias in large-scale visual datasets. *Advances in Neural Information Processing Systems*, 37:61839–61871.

Zhang, B. H., Lemoine, B., and Mitchell, M. (2018). Mitigating unwanted biases with adversarial learning. In *Proceedings of the 2018 AAAI/ACM Conference on AI, Ethics, and Society*, AIES '18. ACM.