Test-Time Data Augmentation using Latent Style Statistics for Enhanced Domain Generalization

Master Thesis

Master of Science in Computing in the Humanities

Hilke Maria Ahlers

September 15, 2025

Supervisor:

1st: Prof. Dr. Christian Ledig
2nd: Francesco Di Salvo, M.Sc.

Chair of Explainable Machine Learning
Faculty of Information Systems and Applied Computer Sciences
Otto-Friedrich-University Bamberg

Abstract

Domain generalization addresses the challenge of deploying machine learning models in environments where the data distribution differs from that seen during training.

This thesis introduces a new method to improve the robustness of deep neural networks when faced with shifts in data distribution by applying data augmentation at test time using latent style statistics. The approach extracts style-related feature statistics from intermediate layers of a pre-trained model and uses these statistics to generate style-transformed variants of samples at test time. The augmented inputs are then processed by the model and their predictions are aggregated, reducing sensitivity to domain-specific variations. Experiments on standard domain generalization benchmarks demonstrate that this test-time augmentation can improve accuracy on unseen target domains compared to models with training-time style-based augmentation.

The findings indicate that leveraging style variations at inference time effectively mitigates these shifts and enhances the model's generalization performance without requiring any modification to the trained network.

All code and experimental resources accompanying this thesis are accessible under the project repository hosted on GitHub[1].

[1] https://github.com/ahlershilke/latent-style-tta.git

Abstract

Domain Generalization befasst sich mit der Herausforderung, Machine-Learning-Modelle in Umgebungen einzusetzen, in denen die Datenverteilung von der während des Trainings beobachteten abweicht.

Diese Arbeit stellt eine neue Methode vor, um die Robustheit tiefer neuronaler Netze bei Verschiebungen in der Datenverteilung zu verbessern, indem zum Testzeitpunkt eine Datenaugmentierung unter Verwendung latenter Stilstatistiken durchgeführt wird. Der Ansatz extrahiert stilbezogene Merkmalsstatistiken aus den Zwischenschichten eines vortrainierten Modells und verwendet diese, um stiltransformierte Varianten jeder Testeingabe zu generieren. Die augmentierten Eingaben werden dann vom Modell verarbeitet und ihre Vorhersagen aggregiert, wodurch die Empfindlichkeit gegenüber domänenspezifischen Schwankungen verringert wird. Experimente mit standard Datensätzen zu Domain Generalization zeigen, dass diese Augmentierung zum Testzeitpunkt die Genauigkeit auf unbekannten Zieldomänen im Vergleich zu Modellen mit stilbasierter Augmentierung zur Trainingszeit verbessern kann.

Die Ergebnisse deuten darauf hin, dass die Nutzung von Stilvariationen zum Zeitpunkt der Inferenz diese Verschiebungen wirksam abmildert und die allgemeine Leistung des Modells verbessert, ohne dass Änderungen am trainierten Netzwerk erforderlich sind.

Der gesamte Code und alle experimentellen Ressourcen zu dieser Arbeit sind im Projekt-Repository auf GitHub[2] verfügbar.

[2]https://github.com/ahlershilke/latent-style-tta.git

Acknowledgements

I am grateful to Prof. Dr. Ledig for giving me the opportunity to write my thesis at his chair. I also want to acknowledge my second supervisor, Francesco Di Salvo, for his continuous support and very helpful pep talks.

I would like to thank my family and friends for their kind words of encouragement and care throughout this process.
Special thanks go to my support group D.R.O.P.O.U.T.[3], who made sharing the burdens about theses a group effort − you guys are the best.

And finally, I want to thank myself for pushing through and getting this thesis done. Cheers!

[3]Doing Research. Overwhelmed. Procrastinating. Overcaffeinated. Underfunded. Terrified.

Contents

List of Figures

List of Tables

List of Acronyms

AdaIN	Adaptive Instance Normalization
AUAD	Area Under Accuracy Drop
AUC	Area Under Curve
CNN	Convolutional Neural Network
DA	Domain Adaptation
DG	Domain Generalization
ERM	Empirical Risk Minimization
IID	Independent and Identically Distributed
LODO	Leave One Domain Out
ML	Machine Learning
OOD	Out of Distribution
PACS	Photo, Art Painting, Cartoon, Sketch
ResNet	Residual Network
ROC	Receiver Operating Characteristic Curve
t-SNE	t-distributed Stochastic Neighbor Embedding
TTA	Test-Time Augmentation
VLCS	VOC2007, LabelMe, Caltech101, SUN09

1 Introduction

1.1 Motivation

Deep learning has become a central technology in computer vision, powering applications from autonomous driving to medical imaging. Neural networks typically assume that the training and test data are drawn from the same distribution (Zhou et al., 2022a). In practice, this assumption rarely holds due to variations in visual style, lighting or other domain-specific factors. For example, a model trained on clear daytime images may perform poorly on images taken at night or in foggy conditions (Zhou et al., 2022a). Domain Generalization (DG) addresses this problem by learning representations that remain robust across distribution shifts, using one or multiple source domains for training to enable generalization to previously unseen target domains (Zhou et al., 2022a). In DG, the model has access only to the source domains during training, which makes large domain shifts particularly challenging.

Many recent works focus on image style as a way to characterize domain differences. In convolutional neural networks (CNN), the channel-wise means and variances of feature maps, which are often called latent style statistics, encode the low-level appearance of an image (Park et al., 2023). Observations by Huang and Belongie (2017) and others show that domain characteristics strongly correlate with these feature statistics. For example, approaches by Zhou et al. (2021) and Li et al. (2021) explicitly mix the style statistics of different images during training to generate new domain variants, which has been shown to improve generalization. However, even with these methods, an unseen target domain may lie far outside the range of generated styles, and a model trained only on source data may still be affected by a large style gap.

A promising direction for addressing domain shift is test-time adaptation and augmentation. Instead of relying exclusively on training-time strategies, these methods adapt the model or its inputs at inference. Test-time data augmentation, in particular, leverages the idea of exposing a model to transformed versions of each test sample to improve prediction robustness (Shanmugam et al., 2020). Recent work has shown that augmenting test data with style-related variations can reduce sensitivity to unseen domains. Park et al. (2023) propose shifting a test sample's style to match the nearest training domain using Adaptive Instance Normalization (AdaIN) normalization.

Inspired by Park et al. (2023), we focus on transforming the test data itself by augmenting it with source-domain style statistics from multiple intermediate convolutional layers. By manipulating these statistics at inference, we align test samples with the known source-domain characteristics. In doing so, we aim to improve robustness without altering the trained model's parameters to offer a lightweight and flexible approach for DG.

1.2 Focus and Scope

The central focus of this thesis is to evaluate whether inserting source-domain feature statistics into selected layers of a network during inference can enhance DG. This approach falls under the umbrella of Test-Time Augmentation (TTA), but it differs in that it operates within the feature space of a pre-trained model and selectively shifts the latent style of test samples toward training domains.

The scope of this work are classification tasks. We use convolutional residual networks as the backbone architecture. Two benchmark datasets are employed, PACS (Li et al., 2017) and VLCS (Fang et al., 2013), which are widely adopted in DG literature (PACS stands for *Photo, Art Painting, Cartoon, Sketch*, VLCS is short for *VOC2007, LabelMe, Caltech101, SUN09*; both are acronyms for their respective domains). Within this setup, we systematically compare different insertion points in the network and focus on how layer depth influences the trade-off between style alignment and semantic preservation.

By narrowing the scope to this setting, we aim to provide insights into the interplay between feature statistics, TTA and classification performance. These insights are intended to inform future extensions to more complex architectures, tasks and domains.

1.3 Relevance of Research

The relevance of this work lies in its position at the intersection of three active research areas: domain generalization, style transfer and test-time augmentation. Prior approaches such as MixStyle (Zhou et al., 2021) have demonstrated the effectiveness of blending feature statistics during training to create synthetic domains. At the same time, test-time augmentation shows that adjusting models or inputs at inference can mitigate unexpected domain shifts (Park et al., 2023).

This thesis aims to contribute to this landscape by exploring a hybrid perspective. Instead of retraining the model or relying only on training-time mixing, we directly manipulate test samples in feature space during inference. The novelty lies in evaluating where in the network this manipulation is most effective, and whether it provides measurable robustness improvements across benchmarks.

By situating the work within this broader literature, the thesis both validates and extends the idea that style statistics are useful for managing domain shifts. It provides insight on the conditions under which test-time style augmentation succeeds and where its limitations become apparent.

1.4 Contributions

This thesis aims to make the following contributions:

1. Test-Time Latent Style Augmentation: We propose a test-time data augmentation framework that modifies test features by injecting source-domain style statistics. This approach improves the model's robustness to unseen target domains without any model fine-tuning.

2. Analysis of Extraction Strategies: We systematically evaluate how the layer at which style augmentation is applied affects domain generalization. Our results show that aligning low-level statistics often yields the largest improvements in bridging domain gaps, while perturbing higher-level features tends to hurt semantic content.

3. Augmentation Techniques: We explore multiple methods for blending or substituting feature statistics during inference. For example, we test simple substitution versus mixing of channel means and standard deviations. We find strategies that effectively draw test samples towards the source-domain manifold while preserving class identity, and we quantify their impact on accuracy.

4. Uncertainty-Accuracy Study: We investigate the relationship between predictive accuracy and model uncertainty under different strategies. Our analysis reveals whether more accurate configurations also tend to produce more confident predictions. This has implications for using uncertainty estimates to gauge reliability in domain-shift scenarios.

5. Empirical Validation: We demonstrate our approach on the PACS and VLCS benchmarks. On challenging domains, our approach substantially improves classification accuracy by reducing style discrepancies. The results support our hypotheses about layer choice and style injection, and provide practical insights for future DG research.

Thesis Organization The remainder of the thesis is organized as follows. Chapter 2 reviews the theoretical background on domain generalization and related concepts. Chapter 3 describes our methodology in detail, including the test-time augmentation pipeline and feature extraction strategies. Chapter 4 presents the experimental setup and results, analyzing the influence of different strategies and validating the research questions. Finally, Chapter 5 concludes with a discussion of the findings, limitations and possible future directions.

2 Theoretical Background

2.1 Domain Generalization

Domain Generalization refers to a machine learning (ML) paradigm that aims to overcome the fundamental challenge of distributional shift between training and deployment environments by learning models capable of generalizing to out-of-distribution (OOD) target domains without access to any target domain data during training (Mai et al., 2025; Zhou et al., 2022a).

Formally, let the input space be denoted by $\mathcal{X}$ and the label space by $\mathcal{Y}$. A domain is defined as a joint probability distribution P_{XY} over $\mathcal{X} \times \mathcal{Y}$. For a particular domain P_{XY}, we denote P_X as the marginal distribution over inputs, $P_{Y|X}$ as the posterior distribution and $P_{X|Y}$ as the class-conditional distribution.

In DG, we are given access to K related but distinct source domains

$$\mathcal{S} = \{\mathcal{S}_k = \{(x_i^{(k)}, y_i^{(k)})\}_{i=1}^{N_k}\}_{k=1}^{K} \tag{1}$$

where each domain S_k is sampled from a distinct joint distribution $P_{XY}^{(k)}$. It holds that $P_{XY}^{(k)} \neq P_{XY}^{(k')}$ for all $k \neq k'$. The goal is to learn a predictor $f : \mathcal{X} \to \mathcal{Y}$ based solely on source domains such that the prediction error is minimized on an unseen target domain $\mathcal{T} = \{x_i^T\}_{i=1}^{N_T}$ with its own distribution P_{XY}^T, where $P_{XY}^T \neq P_{XY}^{(k)} \forall k \in \{1, \ldots, K\}$ (Mai et al., 2025; Zhou et al., 2022a).

The DG task was first formalized as a learning problem by Blanchard et al. (2011), while the term 'domain generalization' was later introduced by Muandet et al. (2013). Unlike Domain Adaptation (DA) or transfer learning, DG assumes that no samples, not even unlabeled ones, from the target domain are available at training time, which makes it a more challenging yet practical setting for real-world applications (Gulrajani and Lopez-Paz, 2020).

Multi-Source vs. Single-Source Domain Generalization DG has been most extensively studied under the multi-source setting, where data is drawn from multiple distinct but related source domains ($K > 1$). This setting is particularly powerful because it enables models to discover statistically invariant patterns across domains, which are more likely to hold in novel target domains (Zhou et al., 2022a). The original motivation for DG, as outlined by Blanchard et al. (2011), was to exploit these multiple sources to learn robust predictors without relying on target domain data. In contrast, the single-source DG setting assumes all training data originates from a single distribution ($K = 1$). While more restrictive, this setting is relevant to research on robustness to OOD shifts, such as image corruptions or changes in environment. Many techniques originally designed for multi-source DG are also applicable in the single-source setting and vice versa, as they generally focus on enhancing the model's robustness to domain shifts in a more general sense (Gulrajani and Lopez-Paz, 2020).

Domain Shift and the IID Assumption A core challenge in DG is the domain shift problem, which refers to a mismatch between the distributions of training and test data. Formally, if $\mathcal{D}_{train} \sim P(train(X, Y)$ and $\mathcal{D}_{test} \sim P(test)(X, Y)$ with $P_{train} \neq P_{test}$, we have a distribution shift that violates the standard IID assumption of training and test samples being independently and identically distributed (Zhou et al., 2022a). Such distributional discrepancies are ubiquitous in real-world applications and often lead to significant drops in model performance (Mai et al., 2025). For example, Torralba and Efros (2011) demonstrated that an object detector trained on one image dataset failed to generalize to another dataset due to dataset-specific biases. Likewise, modern deep learning systems such as self-driving car vision models can 'crash' when tested in conditions unlike those in training with changes in lighting, weather or sensor perspective degrading their performance (Gulrajani and Lopez-Paz, 2020). These failures occur because models trained via empirical risk minimization (ERM) on $\mathcal{D}_{train}$ tend to overfit to domain-specific patterns by exploiting incidental or spurious correlations in the source data that do not hold in the new domain (Gulrajani and Lopez-Paz, 2020). Deep neural networks in particular are notorious for latching onto superficial cues that can differ across domains (Gulrajani and Lopez-Paz, 2020). As a result, even minor perturbations in the data-generating process can cause significant drops in accuracy when models are evaluated OOD (Zhou et al., 2022a). This underscores the inherent fragility of current ML systems under distribution shift and highlights the need for robust generalization, i.e. models that maintain high performance despite changes in data distribution. Designing algorithms to achieve this kind of robustness is therefore of importance in DG research.

Most existing DG benchmarks assume a relatively homogeneous domain shift, where the differences between source and target domains are of a similar nature across all domains (Zhou et al., 2022a). Under such setups, the source to source shifts encountered during training are highly correlated with the source to target shifts at test time. However, real-world scenarios often exhibit heterogeneous domain shifts, where the target domain differs in a fundamentally new way not observed in any source domain (Zhou et al., 2022a). In these cases, the shifts between training domains provide poor predictors of the shift to an unseen test domain. For example, sources might consist of photos, paintings or sketches of objects, while an unforeseen target domain contains images captured from novel viewpoints (Zhou et al., 2022a). Another example would be digit images under various rotations as source domains with the target domain containing digits with a different texture or background (Zhou et al., 2022a). Such heterogeneous shifts are a critical challenge for DG, and they are underrepresented in current benchmarks (Zhou et al., 2022a).

Formal Learning Objective in DG The supervised learning objective in standard settings is to learn a function $f : \mathcal{X} \to \mathcal{Y}$ that minimizes the expected loss over the true data distribution:

$$\mathbb{E}_{(x,y)\sim P}[\ell(f(x), y)]. \tag{2}$$

In practice, due to limited access to the real distribution P, this is approximated via ERM on a training dataset $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{n} \sim P$:

$$\hat{f} = \arg\min_{f} \frac{1}{n} \sum_{i=1}^{n} \ell(f(x_i), y_i). \tag{3}$$

In DG, this objective is redefined to account for distributional variability across multiple training domains with the goal of minimizing the expected loss on unseen domains:

$$\hat{f} = \arg\min_{f} \max_{P \in P_{test}} \mathbb{E}_{(x,y)\sim P}[\ell(f(x), y)], \tag{4}$$

where $\mathcal{P}_{test}$ is the family of test distributions that are unknown during training but share certain invariances with the source domains (Arjovsky et al., 2020; Gulrajani and Lopez-Paz, 2020). Various Domain Generalization algorithms differ in how they model and enforce invariances, for example by aligning causal mechanisms through regularization based on average causal effects (e.g., Contrastive ACE by Wang et al. (2023)) or by learning transformations that minimize distributional divergence across domains to extract domain-invariant features (e.g., Domain-Invariant Component Analysis by Sheth et al. (2022)).

Methods and Applications DG has found applications across domains, particularly in computer vision, speech recognition and medical imaging.

In computer vision, DG is crucial for tasks like object recognition and detection under varying conditions. Models are often evaluated on datasets such as PACS (Li et al., 2017), VLCS (Fang et al., 2013) or DomainNet (Peng et al., 2019), which simulate shifts in image style or capture environment (Gulrajani and Lopez-Paz, 2020). Here residual networks (ResNet) (He et al., 2016) serve as strong baselines, and methods ranging from style transfer data augmentation to domain-adversarial training are used to learn representations invariant to image style. The efficacy of these approaches is evidenced by robust performance across drastically different visual domains in object recognition tasks (Gulrajani and Lopez-Paz, 2020). Another vision application is in autonomous driving, where models trained in one city or weather condition need to generalize to new cities or weather conditions without additional tuning, a scenario where DG algorithms can improve safety by handling unseen conditions. The computer vision field has driven much of development in DG, with algorithms stress tested on heterogeneous image domains and even challenges like image corruption robustness under the single-source DG setting.

In speech recognition, DG addresses variability in speakers, accents, languages and acoustic environments. A notable study by Narayanan et al. (2018) trained an end-to-end ASR system on six distinct source domains like voice search queries, dictated speech or telephone audio and tested it on unseen domain data of telephony speech. The multi-source trained model achieved better accuracy on the unseen domain than any single-source model, demonstrating that diverse training data leads to a more domain-agnostic speech recognizer.

In medical imaging, DG is particularly critical due to the variability across hospitals, imaging devices and patient populations (Yoon et al., 2024). A model trained on scans from one hospital may fail when applied to another hospital's data because of shifts in image distributions caused by different scanner manufacturers, protocols or demographic differences. In this direction, DG is often addressed through data augmentation to mimic the induced variability. Zhang et al. (2020) proposed an algorithm with a stacked transformation pipeline for combining intensity, appearance and spatial changes, which significantly improved cross-site MRI and ultrasound segmentation without target data. Ouyang et al. (2022) introduced a causality-inspired augmentation framework that generates diverse appearance shifts and breaks spurious correlations to achieve robust gains across cross-modality and cross-site benchmarks. Similarly, Zhou et al. (2022b) used style-based intensity transformations and dual normalization to enhance cross-modality generalization in brain and cardiac imaging.

A key enabler for DG has been the advent of deep architectures like ResNets. Introduced by He et al. (2016), ResNet uses skip connections (residual shortcuts) to combat the vanishing gradient problem to allow the training of ultra-deep CNNs without degradation in performance. These residual networks achieved state-of-the-art results on ImageNet (Deng et al., 2009) and demonstrated excellent generalization to other recognition tasks. ResNets (especially ResNet-50) have become a backbone in many DG studies. Recent evidence suggests that a strong backbone coupled with proper training strategies can substantially narrow the gap between vanilla ERM and specialized DG algorithms. Gulrajani and Lopez-Paz (2020) found that a deep ResNet with heavy data augmentation performed on par with numerous DG-specific methods across vision benchmarks. Likewise, an extensive study by Angarano et al. (2024) showed that simply adopting a more powerful feature extractor together with effective augmentation allows plain ERM to outperform many dedicated DG solutions, even reaching state-of-the-art accuracy on standard benchmarks. These findings underscore the importance of network architecture in DG and suggest that continuing advances in backbone designs directly translate to stronger OOD performance.

To address the challenges of DG, multiple algorithmic strategies have been developed. These include domain-invariant representation learning, where distributions are aligned in feature space (Muandet et al., 2013), meta-learning, where models are trained on simulated domain shifts to improve their adaptability (Zhou et al., 2022a) and data augmentation, which introduces variability in the training data to enhance model robustness (Mai et al., 2025). Data augmentation methods in particular have been proven effective for DG.

2.2 Augmentation

Data Augmentation is a key strategy for improving model robustness under domain shift in DG. By generating diverse training examples, augmentations encourage models to learn invariant features that generalize to unseen domains. Augmentation methods are often categorized by where they act in the pipeline. Image-level (input-space) augmentations operate directly on raw images and apply transformations or perturbations that mimic cross-domain variability (Mai et al., 2025). In contrast, feature-level augmentations act on intermediate neural representations and inject variability into the latent feature space. Style-transfer–based augmentations have emerged as a complementary approach by synthesizing novel image styles while preserving semantic content to expose models to a broader range of appearance variability.

2.2.1 Image-level Augmentation

At the image level, augmentations manipulate the raw input space $\mathcal{X}$ to simulate inter-domain variability. Traditional geometric and photometric augmentations such as random flips, crops, color jitter and scaling are simple yet effective techniques to enlarge the training set. For example, random horizontal flips and crops encourage invariance to object viewpoint and positioning, while brightness and contrast adjustments or Gaussian noise mimic camera or sensor variability (Mai et al., 2025; Schwonberg et al., 2023). These classic augmentations preserve the semantic label of an image (e.g., an object remains the same despite a flip) and have proven beneficial in many DG settings. In particular, device-related shifts in medical imaging (e.g., changes in scanner settings or patient conditions) can often be mitigated by basic augmentations (Mai et al., 2025; Zhou et al., 2022a). However, it is important that augmentations do not distort class semantics. For instance, flipping a handwritten digit may alter its label and extreme color changes can render an object unrecognizable. Thus, practitioners typically choose augmentations that are semantically appropriate for the task and domain. When well-chosen, even standard augmentations can significantly improve generalization without adding complexity to the model.

Beyond naive transformations, advanced image space strategies can generate more powerful domain variations. One notable example is Fourier-based Amplitude Mixing (Xu et al., 2021). This method decomposes an image into its Fourier amplitude (low-level texture) and phase (high-level structure). To synthesize an augmented image, Xu et al. (2021) linearly interpolate the amplitude of two images from different domains while keeping the phase fixed. After mixing amplitudes, the inverse Fourier transform produces the new image. Because phase encodes the object's semantic layout, preserving it ensures that the augmented image depicts the same content (e.g., a 'horse' remains recognizable) while its texture and color statistics are blended. In practice, an image reconstructed from only phase retains clear structure, whereas an amplitude-only reconstruction loses detail (Xu et al., 2021). By

perturbing the entire amplitude spectrum, the Amplitude Mixing strategy injects aggressive yet label-preserving variations. The authors show that this forces the model to focus on phase (semantic) information and become invariant to amplitude variations. In other words, Fourier mixing helps the network ignore domain-specific texture differences and concentrate on the core content, which enhances OOD performance.

Another sophisticated augmentation is Stochastic Feature Augmentation (Li et al., 2021), which although applied in feature space, can be viewed as an image-level regularizer. This approach injects Gaussian noise into the latent embedding of each training image during forward passes. In its simplest form, a feature vector is scaled and shifted by random factors sampled from normal distributions. This perturbs features in a data-independent way and creates slight variations of each example as if drawn from a continuous domain. The authors further propose an adaptive variant by maintaining running estimates of the class-conditional covariance matrices of features and sampling a noise vector to add to features of a class. Intuitively, this adds larger perturbations along directions of high inter-domain variability while preserving intra-class structure. Although Stochastic Feature Augmentation modifies feature vectors, it has a comparable effect to image augmentation. Li et al. (2021) demonstrate that this simple plug-in augmentation substantially improves DG performance.

In addition, sample mixing augmentations like MixUp (Zhang et al., 2018) and AugMix (Hendrycks et al., 2020) have been used to synthesize novel images. In MixUp, two images are combined, with their one-hot labels similarly mixed. This creates training samples 'between' domains and encourages linear label interpolation. AugMix takes a different approach: It applies multiple random simple augmentations from a predefined set (like rotation or contrast change) in sequence and averages the results with the original. Both techniques generate diverse yet realistic augmentations that help models generalize. While they were not specifically designed for DG, they are complementary tools for creating domain-bridging samples.

It is important to exercise caution when applying augmentations. Wen et al. (2021) show that MixUp can adversely interact with model ensembling. They find that the soft-labeling inherent in MixUp leads to ensembles that are systematically underconfident. Intuitively, mixing labels across classes spreads probability mass, so when averaged over ensemble members, predictions gravitate towards a low-confidence baseline. This 'compounding under-confidence' degrades calibration (reliable uncertainty estimation). Wen et al. (2021) address this by proposing CAMixUp, which adapts the mix ratio λ on a per-class basis according to the accuracy-confidence mismatch. By doing so, they partially recover confidence levels without sacrificing the accuracy gains from MixUp. So complex augmentations, especially those which soften labels, must be integrated carefully, since naive combinations can yield unforeseen calibration issues (Wen et al., 2021). Beyond calibration, overly aggressive augmentations may introduce artifacts or unrealistic samples that confuse the model. Thus in practice, one typically applies a balanced suite of augmentations by combining several mild transformations such as blur, color shift and slight geometric

distortions, which often yield the best results. Schwonberg et al. (2023) find that blending multiple simple augmentations can nearly match complex DG methods in semantic segmentation. The authors observe that chains of basic augmentations outperform single operations and achieve competitive mean Intersection over Union values on challenging synthetic-to-real benchmarks.

In summary, image-level augmentation spans a spectrum from standard color transforms to novel spectral and mixing techniques. Each tool has trade-offs: while geometric and photometric edits are cheap and interpretable (Mai et al., 2025), advanced mixes create stronger domain shifts at the risk of calibration or semantic distortion (Wen et al., 2021; Xu et al., 2021). Selecting and tuning augmentations remains task-dependent, but an effective augmentation pipeline typically includes a variety of both low-level and learned transformations.

2.2.2 Feature-level Augmentation

Unlike image-level augmentations, feature-level augmentations operate on the hidden representations (the latent features) $\phi(x)$ within the neural network rather than on pixels. In a deep CNN, latent space refers to the intermediate feature maps or activation vectors produced by each layer, essentially a high-dimensional encoding of the input image at a level of abstraction (Islam et al., 2023). These latent representations capture salient information about the input. Lower-layer CNN features tend to encode style or texture, while deeper layers capture more content or structure related to object identity (Zhou et al., 2021). By manipulating these internal features directly, variability to simulate new domain conditions without altering the pixel space can be introduced. Because augmentations in latent space do not require rendering of an image, they are often more flexible and computationally cheaper (Mai et al., 2025). Many feature-space augmentation modules are built for easy insertion as an extra layer in the network without requiring major architectural changes (Mai et al., 2025; Zhou et al., 2021). This makes them a practical option for improving domain robustness.

Latent feature augmentation methods generally fall into two broad categories, feature perturbation and feature stylization. These are also known as adversarial and non-adversarial feature augmentations (Mai et al., 2025). Feature perturbation techniques introduce noise or other random perturbations into the intermediate features to simulate domain shifts. For instance, Simple Feature Augmentation by Li et al. (2021) adds Gaussian noise to feature activations by randomly scaling and shifting the feature map values. They sample random affine parameters from a Gaussian distribution and apply them to the feature tensor, which injects controlled noise into the network's representation. Such perturbations imitate real-world variation in the latent space. By training with these perturbed features, the model learns to ignore domain-specific activation patterns and focuses instead on the essential signals that generalize across domains. Some approaches craft these perturbations adversarially, for example by adding feature noise that maximally confuses a domain classifier (Mai

et al., 2025), to explicitly force the learned representation toward domain-invariant directions.

Feature stylization methods alter the distributional statistics of feature maps to create novel styles in the latent space. A hallmark feature-level method is MixStyle (Zhou et al., 2021). Intuitively inspired by style transfer techniques such as AdaIN, MixStyle randomly exchanges the channel-wise feature statistics between two samples. Given two feature embeddings f_i and f_j (e.g., from the first CNN block), their per-channel means and standard deviations are computed (μ_i, σ_i) and (μ_j, σ_j), then a value $\lambda \sim \text{Beta}(\alpha, \alpha)$ is sampled and mixed statistics $\mu_{\text{mix}} = \lambda\mu_i + (1-\lambda)\mu_j$, $\sigma_{\text{mix}} = \lambda\sigma_i + (1-\lambda)\sigma_j$ are formed. Finally, these mixed features are applied to normalize and re-scale f_i:

$$f_{\text{mix}} = \sigma_{\text{mix}} \odot \frac{f_i - \mu_i}{\sigma_i} + \mu_{\text{mix}}. \tag{5}$$

This operation swaps the 'style' or texture encoded in f_i with that of f_j while leaving the relative content structure intact. Intuitively, mixing feature statistics in this way synthesizes new domain appearances within the feature space so the network is exposed to a continuum of interpolated styles beyond those present in the original training data Zhou et al. (2021). This is done implicitly during training without needing to generate any actual images to keep it efficient. By randomizing feature distributions on the fly, the model is encouraged to learn latent representations that discount style variations as irrelevant and focus on more stable, domain-agnostic features. Zhou et al. (2021) observe that visual domains form distinct clusters in the feature-style space. By continually mixing these statistics during training, MixStyle creates a continuum of novel styles. This creation of synthetic distributions effectively augments the feature set with unseen domain styles without changing the class labels. Empirically, feature stylization methods like MixStyle have shown substantial gains in DG performance across vision benchmarks (Mai et al., 2025; Zhou et al., 2021).

Several extensions and variants build on the MixStyle idea. Jeon et al. (2021) make feature augmentation aware of class semantics through contrastive learning to further improve domain robustness. The authors combine feature stylization with a domain-aware contrastive loss by decomposing a feature map into high-frequency (detailed) and low-frequency (coarse) components. Then they stylize only the low-frequency part by mixing in statistics from other samples which preserves the high-frequency shape cues. The augmented features are merged back and fed to the network. They introduce a supervised contrastive objective that pulls together stylized features of the same class across different simulated styles while pushing apart different classes. This encourages the model to learn features that are invariant to the artificial style perturbations yet still discriminative by class.

Yamashita and Hotta (2024) extend MixStyle into a framework for adaptation at inference. They propose MixStyle-based Contrastive Test-Time Adaptation, where the network learns with two objectives, standard classification and a MixStyle-driven contrastive task. At test-time, they apply MixStyle-based contrastive updates to

adapt the feature extractor to the incoming unlabeled domain. This two-phase scheme leads to state-of-the-art DG accuracy on benchmarks like PACS and Office-Home and surpasses even specialized Test-Time Augmentation methods. The key insight is that the same style-mixing that diversifies domains during training can be used to align the model to a new domain at inference, all without needing target labels.

Other feature-level augmentations have been explored, such as Huang et al. (2025), who introduce a generative feature-style augmentation targeted at medical image segmentation. They train a style generator to produce plausible variations of feature 'style' that mimic different imaging devices. By feeding these synthetic style-perturbed features to the segmentation network during training, they increase robustness to cross-device shifts.

Park et al. (2023) focused on inference-time stylization. They compute a test image's style vector and align it to the nearest source domain style centroid via feature normalization. This style shifting at test time requires no gradient updates and effectively induces the model into processing the input as if it came from a known source domain, which reduces performance drops due to style discrepancy.

While the above methods differ in implementation, the unifying theme is leveraging the latent feature space to induce variability that covers potential domain shifts. Augmenting the hidden representations expands the support of the training data in a conceptually richer way than pixel-level augmentation alone. The model learns to stabilize its intermediate representations against these perturbations, which yields features that are more invariant to domain-specific quirks. In practice, feature-level augmentations often complement image-level ones, and using both in tandem can substantially enrich the training distribution. That said, designing effective latent-space augmentations remains an active search area. It requires balancing diversity and realism of the perturbations such that they help rather than hurt learning (Mai et al., 2025). The continued refinement of these approaches is crucial to improving robustness. The consensus in recent literature is that perturbing feature representations markedly improves a model's ability to generalize to new domains (Mai et al., 2025; Wen et al., 2021; Zhou et al., 2021).

2.2.3 Style Transfer

Style-transfer augmentation refers to techniques that transform the visual style (e.g., textures, color statistics, contrast) of input images while preserving their high-level semantic content, to simulate domain shifts in the training data and thus improve OOD robustness. It operates in pixel space like image-level methods but shares the objective of promoting style invariance typically targeted by feature-level approaches.

Recent work has extended this idea in several ways. For example, the approach by Yamashita et al. (2021) replaces low-level texture features of histopathology images with styles sampled from non-medical image sources (e.g., paintings or artistic styles), while keeping cellular or tissue structures intact. This encourages the model

to rely less on superficial texture cues and more on content and shape (Yamashita et al., 2021).

In medical imaging, DG methods using style transfer have also been applied to cross-modality segmentation, where images from different imaging modalities differ in contrast and intensity distributions. The recent work by Zhou et al. (2022b) uses Bezier-curve transformations to generate 'source-similar' styles, which are close to the original domain, and 'source-dissimilar' styles, which show more extreme transformations, to augment the source data. A dual-normalization module helps the network adapt to both types of styles and generalize to new modalities (Zhou et al., 2022b).

Further, in histopathological image classification, style-augmented feature domain mixing that uses AdaIN to perturb the style statistics of feature maps has been shown to improve generalization substantially, often with lower computational overhead than full image-space transformations (Khamankar et al., 2023).

Empirical results across these works show that style-transfer augmentations tend to outperform or complement traditional image-level augmentations, especially when style shifts are a major factor in domain difference (Khamankar et al., 2023; Yamashita et al., 2021; Zhou et al., 2022b). They help widen the support of training data in appearance space, reduce overfitting to source-style texture biases, and improve the feature invariance to style (Khamankar et al., 2023; Yamashita et al., 2021). The efficacy of style-transfer augmentation depends on preserving semantic content during transformation (Yamashita et al., 2021). The style variations applied must be diverse and realistic, but should not compromise label integrity (Zhou et al., 2022b). Additionally, these augmentations should be carefully balanced and integrated with other data augmentation strategies to maintain a representative and stable training distribution (Khamankar et al., 2023).

2.3 Test-Time Augmentation

Test-Time Augmentation (TTA) refers to the technique of applying data augmentation during inference to improve a model's predictions on new inputs. In practice, multiple transformed versions of each test example are passed through the model and their predictions are aggregated, typically by averaging, to yield the final prediction (Kimura, 2021; Shanmugam et al., 2021). This simple ensemble-style approach can substantially improve robustness: for instance, Kimura (2021) describes TTA as "a very powerful heuristic" that "takes advantage of data augmentation during testing to produce averaged output". Similarly, Shanmugam et al. (2020) note that TTA is a common practice in image classification which often yields net accuracy gains.

In essence, TTA leverages the same intuition as training-time augmentation (increasing data diversity) and model ensembling by exposing the model to multiple views of an input, which enables smoothing out of idiosyncratic errors and produces a more reliable estimate. Unlike training-time augmentation and test-time adapta-

tion, which modify the model or its parameters during training or inference, test-time augmentation leaves the model fixed and instead augments the test samples, aggregating predictions from their transformed versions (Kimura, 2021; Shanmugam et al., 2020).

From a theoretical perspective, researchers have shown that TTA can strictly reduce expected error under certain assumptions (Kimura, 2021). Intuitively, if individual augmentations produce somewhat independent prediction errors, then averaging can cancel out random mistakes (much like an ensemble) and lower overall error (Kimura, 2021; Shanmugam et al., 2020). Experiments confirm that simple TTA often improves accuracy on computer-vision tasks (Kimura, 2021; Shanmugam et al., 2020). However, Shanmugam et al. (2020, 2021) emphasize that naive TTA can also corrupt some predictions: Even when the average accuracy increases, "it can change many correct predictions into incorrect predictions" (Shanmugam et al., 2020). That work analyzes when and why TTA succeeds or fails, noting that the choice of augmentations and aggregation method matters. Building on these insights, Shanmugam et al. (2021) propose learnable aggregation weights rather than simple averaging, which consistently outperforms naive TTA.

TTA is closely related to DA and DG because it actively uses unlabeled test data to adjust the model's output. As Zhou et al. (2021) note, test-time training (or adaptation) "blurs the boundary between [domain] adaptation and domain generalization". In both cases, the model must cope with domain shift from training (source) to testing (target) distributions. TTA effectively assumes that small, unlabeled test batches are available to the model at inference time. Like DA, it tailors the model to the test distribution (albeit without labels), but unlike typical adaptation it usually makes no architectural changes and requires no supervision. Zhou et al. (2021) argue that TTA is thus related to source-free DA (no access to source data at test time) while also resembling generalization approaches that assume target data will be encountered during deployment. In practice, TTA is often evaluated on the same corrupted-image benchmarks used in domain-shift research and tends to yield larger gains than generic DG methods because it exploits actual test samples (Zhou et al., 2021).

Style-Specific Test-Time Augmentation Certain TTA methods explicitly target style differences between domains. Park et al. (2023) observe that in DG scenarios, a target domain may have very different visual style statistics (e.g., texture or color distributions) from all source domains. To address this, they propose Test-Time Style Shifting, as introduced in section 2.2.2, a technique that modifies the style of each test sample before inference without any gradient updates. Concretely, the method finds the nearest source-domain style (in terms of batch normalization statistics) and shifts the test image's style toward that domain's style distribution (Park et al., 2023). For an incoming feature map $f(t)$ extracted from an early convolutional block, they compute its style vector

$$\Phi(f(t)) = [\mu_c(f(t)), \sigma_c(f(t))]_{c=1}^{C} \tag{6}$$

where μ_c and σ_c are the per-channel mean and standard deviation across spatial locations. They precompute and store the style centroids Φ_{S_k} for each source domain S_k. At test-time, they measure the Euclidean distance to each centroid and identify the nearest domain $k^* = \arg\min_k d_k$. If $d_{k^*} > \alpha$, which is a small threshold tuned on source validation data, they perform Adaptive Instance Normalization (AdaIN) by normalizing $f(t)$ using its own μ, σ and then re-scaling and shifting with $\mu_{S_{k^*}}, \sigma_{S_{k^*}}$:

$$\hat{f}(t) = \sigma_{S_{k^*}} \frac{f(t) - \mu(f(t))}{\sigma(f(t))} + \mu_{S_{k^*}}. \tag{7}$$

This style shift forces the network to process the test sample as if it came from a familiar source domain, thereby reducing texture- or color-based domain shift. During training, they further introduce a lightweight style balancing module that periodically mixes style statistics among under-represented domain-class pairs to ensure the stored centroids reflect balanced class proportions and prevent bias in the nearest-centroid selection. Together, these components require no model updates or additional parameters at inference, yet consistently improve OOD classification accuracy on benchmarks with large style gaps.

Conceptually, this approach is a hybrid of augmentation and adaptation. It does not update the model but transforms the input representation so that its style matches known source domains. This makes the method particularly useful when a test image's style is extreme or anomalous, which allows models to "handle any target domains with arbitrary style statistics, without additional model update at test time" (Park et al., 2023).

Advanced Test-Time Augmentation More recent work has focused on improving the basic TTA framework by making the augmentation process more adaptive and uncertainty-aware. For example, Chen et al. (2023) point out two key pitfalls in standard TTA: selecting an appropriate auxiliary loss for adaptation and deciding which model parameters to update with test data. They propose an improved Test-Time Augmentation method that addresses both issues. First, instead of using a fixed self-supervised loss (like entropy minimization) to tune the model on test samples, this method learns a small set of adaptive consistency parameters within a loss function so that the auxiliary task (e.g., a consistency loss between different augmentations) becomes 'aligned' with the main classification objective (Chen et al., 2023). Second, their approach augments the model with a few new parameters added after each layer that are the only weights tuned at test time, which leaves the rest of the network fixed. This avoids forgetting or instability when updating on a single test batch. In practice, this method achieves state-of-the-art results on challenging DG benchmarks (Chen et al., 2023).

Another recent advance is to make TTA uncertainty-aware. Sherkatghanad et al. (2024) introduce BayTTA, which integrates Bayesian model averaging into the TTA pipeline. The standard TTA pipeline produces multiple predictions by augmenting the input and averaging; BayTTA instead treats each augmented version's prediction as coming from a different 'model' and uses Bayesian Model Averaging (Fragoso

et al., 2018) to combine them. Concretely, BayTTA generates a list of predictions for each input (one per augmentation) and then weights them according to their posterior probabilities under a Bayesian model. This means predictions from more 'likely' augmentations (or ensemble members) have more influence and the process naturally captures model uncertainty. Sherkatghanad et al. (2024) report that their approach significantly outperforms simple averaging TTA, especially on medical imaging tasks with high uncertainty. By explicitly accounting for uncertainty via Bayesian weighting, BayTTa reduces the risk of spurious errors that a naïve average might commit.

In summary, the theoretical foundations outlined above establish the key challenges of domain shift and motivate the use of augmentation and test-time techniques as central strategies, which directly influences the methodological choices in this work.

3 Methodology

3.1 Research Objectives

This project is motivated by the observation that two complementary strands of robustness research offer a promising opportunity to explore. Feature-statistic mixing methods like MixStyle (Zhou et al., 2021) that operate in deep feature space to encourage style-invariance and TTA approaches that adapt predictions at inference by exposing the model to plausible variations both aim to improve generalization under distribution shift, yet they are rarely studied together within a unified framework. This thesis wants to test if training-domain feature statistics can be used to systematically move test features at inference toward the known domains at test time, similarly to style transfer, and if so, how the way of extraction influences the performance of such a model.

The central research objectives driving this project are therefore closely tied to the experimental design:

1. How does the choice of feature extraction strategy affect Domain Generalization performance measured by prediction accuracy when these statistics are used within a TTA pipeline for classification?

2. What methods augment test features toward training domains without erasing class-discriminative information?

3. Is there a correlation between predictive accuracy and model uncertainty across extraction strategies?

Underpinning these objectives are several hypotheses:

1. We hypothesize that the choice of extraction strategy is crucial because different residual blocks encode varying mixtures of semantic content and domain style. While earlier blocks capture lower-level texture and color statistics that are closely tied to appearance, later blocks encode higher-level, class-specific representations (Yosinski et al., 2014; Zeiler and Fergus, 2014). Accordingly, shifting statistics from early blocks is expected to provide stronger alignment style benefits, whereas modifying higher-level representations may be less effective at bridging appearance gaps.

2. We further hypothesize that domain-specific performance patterns observed during training are preserved under TTA. Specifically, domains that achieve the highest accuracy in the training setting are expected to remain the strongest under TTA, while domains that perform poorly in training are likewise anticipated to perform worst under adaptation.

3. Finally, we hypothesize that augmenting test samples from domains with lower baseline performance using feature statistics derived from stronger-performing domains will improve predictive accuracy.

The thesis will operationalize these research objectives and evaluate them through controlled experiments across two datasets with a specialized approach to test the domain-specific statistics on for the model unseen domains. Its contribution lies in demonstrating that feature extraction combined with a style augmentation at inference can further improve domain generalization by identifying which layers are most effective for style shifting. Collectively, these research objectives aim to bridge conceptual gaps between style-mixing augmentation in feature space and TTA.

3.2 Problem Formulation

We consider the problem of Domain Generalization with Test-Time Augmentation in image classification.

Let

- $\mathcal{D} = \{\mathcal{D}_1, \ldots, \mathcal{D}_N\}$ denote the set of source domains,
- where each domain $\mathcal{D}_i$ consisting of samples (x, y) with inputs $x \in \mathbb{R}^{C \times H \times W}$ and class labels $y \in \{1, \ldots, K\}$,
- with N being the number of source domains and K the number of semantic classes.

To simulate real-world generalization to unseen domains, we adopt a Leave-One-Domain-Out (LODO) evaluation protocol:

- In each fold, one domain $\mathcal{D}_t$ is selected as the target domain,
- while the remaining $N - 1$ domains $\{\mathcal{D}_i\}_{i \neq t}$ serve as source domains.

A model f_θ is trained exclusively on labeled data from the union of source domains

$$\mathcal{D}_{src} = \bigcup_{i \neq t} \mathcal{D}_i, \tag{8}$$

and is evaluated on the held-out target domain $\mathcal{D}_t$.

During both training and TTA, we represent the style of an activation map

$$x \in \mathbb{R}^{B \times C \times H \times W} \tag{9}$$

by its per-instance, per-channel statistics:

$$\mu(x)_{b,c} = \frac{1}{HW} \sum_{h=1}^{H} \sum_{w=1}^{W} x_{b,c,h,w}, \tag{10}$$

$$\sigma(x)_{b,c} = \sqrt{\frac{1}{HW}\sum_{h=1}^{H}\sum_{w=1}^{W}(x_{b,c,h,w} - \mu_{b,c}(x))^2}\,. \tag{11}$$

These statistics ($\mu,\ \sigma$) are commonly used to represent style information of a domain, while the normalized activations

$$\hat{x}_{b,c,h,w} = \frac{x_{b,c,h,w} - \mu(x)_{b,c}}{\sigma(x)_{b,c} + \epsilon} \tag{12}$$

encode content features that are more domain-invariant.

The target domain $\mathcal{D}_t$ remains unseen throughout training. To mitigate the domain shift, we employ TTA as follows. First, style statistics from all source domains are pre-computed and stored during training. Second, at inference, unlabeled samples from $\mathcal{D}_i$ are forwarded through the model while domain-aware hooks replace their feature statistics with those of candidate source domains. This process produces a set of augmented predictions $\{f_\theta^{(j)}(x)\}_{j=1}^{N-1}$, one per source domain style. Finally, the predictions are aggregated across these source-domain hypotheses by averaging to obtain the final output for each sample. The learning and inference objectives are defined as follows:

$$\text{Training (per fold):} \quad \min_\theta \frac{1}{N-1}\sum_{i\neq t}\mathbb{E}_{(x,y)\sim\mathcal{D}_i}\ell(f_\theta(x), y), \tag{13}$$

$$\text{Test-time augmentation (Inference on } \mathcal{D}_t\text{):} \quad \hat{y}(x) = \mathcal{A}\left(\left\{f_\theta^{(j)}(x)\right\}_{j=1}^{N-1}\right), \tag{14}$$

where ℓ is the cross-entropy loss and $\mathcal{A}(\cdot)$ is the aggregation operator, e.g. the average of softmax probabilities.

This procedure enables the model to leverage style variability from source domains in order to adapt on-the-fly to unseen domains without requiring target labels or retraining.

3.3 Setup

All experiments were implemented in PyTorch 2.5.1 with torchvision 0.20.1 and executed on a dedicated workstation provided by the Chair. The system was equipped with an NVIDIA RTX A5000 GPU (24 GB VRAM), a 24-core CPU and 63.5 GB RAM. For storage, the workstation offered a 512 GB SSD and an 8 TB HDD, providing sufficient capacity. The software environment was based on Ubuntu 22.04.4 LTS with CUDA 11.5 and cuDNN 9.0.1. To ensure reproducibility, random seeds were fixed across PyTorch, NumPy and the Python random library. For a complete specification of the Python environment, including all package dependencies, please refer to the provided GitHub repository[4].

[4]https://github.com/ahlershilke/latent-style-tta.git

Baseline Model Architecture The baseline model of this work employs a ResNet-50 architecture, which was chosen for its proven effectiveness in visual representation learning and DG tasks. The residual connections of a ResNet enable the training of deeper networks while mitigating vanishing gradient problems (He et al., 2016). This property makes it particularly suitable for learning robust, transferable representations across diverse domains. The model is initialized with weights pretrained on ImageNet (Deng et al., 2009). By using the pretrained version, the early layers of the model are already able to capture generic low-level features such as edges or textures, thereby bootstrapping feature extraction capabilities.

Input images are resized to 224×224 pixels and normalized using standard ImageNet statistics (mean [0.485, 0.456, 0.406], standard deviation [0.229, 0.224, 0.225]). No additional data augmentation is applied to the models used for feature extraction. MixStyle (Zhou et al., 2021) is only applied to the model that serves as the comparative baseline for the evaluation of our method.

The ResNet-50 backbone consists of four residual blocks (layer1 - layer4), where each contains multiple bottleneck layers that progressively transform the input into higher-level semantic features. During training, the outputs of each residual block are captured using forward hooks. This yields hierarchical feature representations at different levels of abstraction (Yosinski et al., 2014; Zeiler and Fergus, 2014):

- **layer1**: low-level features such as textures and edges, with high spatial resolution
- **layer2 - layer3**: mid-level features such as shapes and patterns, at reduced spatial dimensions
- **layer4**: high-level semantic features such as object parts or global context, at the highest abstraction.

Two modes of feature utilization are supported. In end-to-end fine-tuning, all layers are trainable. This allows adaptation of the backbone to the target task. Batch normalization statistics are updated during training to align with the target domain distribution. In frozen feature extraction, the pretrained backbone remains fixed, and only task-specific heads (e.g., classifiers) are trained.

For DG, the system further extracts style statistics (mean μ and standard deviation σ) from the activations of each residual block using the hooks. The statistics are computed spatially over the height and width dimensions `[H, W]` for each channel, which results in descriptors of shape `[B, C, 1, 1]` where B is the batch size and C the channel dimension. A dedicated `StyleStatistics` module aggregates these statistics per domain using exponential moving average (EMA), which ensures stable updates while adapting to domain-specific distributions. These statistics capture domain-specific characteristics while remaining invariant to spatial arrangement, which makes them particularly valuable for domain analysis and adaptation.

LODO Setup To study TTA driven by feature statistics, the deployment data must come from a previously unseen domain. Classical random train/validation/test splits intermingle images from every domain in the used dataset(s) and therefore can not measure performance under genuine domain shift properly. For a proper evaluation of this approach, an evaluation protocol is needed that isolates an entire domain during training, permits learning only from the remaining source domains and tests on the unseen domain for an attempt at imitating domain shift in one dataset. The LODO protocol is able to fulfill these criteria. It ensures that model performance is measured on truly unseen domains and prevents data leakage that could otherwise bias the results. This separation is particularly important for our TTA mechanism. Since it adapts features at inference time, mixing target-domain images into training would compromise the integrity of the style statistics.

Given D domains, the used LODO setup creates D folds. In fold i, domain D_i is fully excluded from training and used only for testing. The other $D - 1$ domains are used to form the training validation set by a 80/20 split, balancing the need for a large and diverse training set with a sufficiently representative validation set for reliable model selection. This ensures the prevalence of every training domain in both train and validation splits. The held-out domain is rotated across folds so that each domain serves once as the target. This yields unseen-domain evaluation, per-domain diagnostics and enables the aggregation of statistics across folds to quantify DG. In our experiments, this setup creates four folds per used dataset.

LODO is integrated into our approach in multiple areas.

- **Data splitting**. Multi-domain datasets are divided into LODO folds.
- **Global Hyperparameters.** Results are aggregated across folds to estimate hyperparameters that generalize robustly. The global configuration is then re-evaluated to confirm domain-agnostic performance and used for model training.
- **Training & statistics collection.** The backbone is trained on the D-1 source domains while collecting layer-wise style statistics per domain. The held-out domain remains completely unseen.
- **Model saving & stats export.** For each fold the best model is saved and its corresponding style statistics are extracted per source domain and per extraction mode.
- **TTA evaluation.** At test time, the model trained without domain D_i is evaluated on that domain using source domain statistics applied through forward hooks.

Hyperparameter Tuning Framework The hyperparameter optimization pipeline employs Optuna (Akiba et al., 2019) to systematically explore the model's

hyperparameter space, utilizing a tree-structured Parzen Estimator sampler for efficient search. Hyperparameters are tuned per fold using only the source domains of the LODO cross-validation setup, after which a global configuration is computed across folds. This two-stage procedure avoids tailoring the model to any single domain, produces parameters that transfer across domains and establishes a consistent configuration for training and evaluation. The optimization process is implemented in a custom class which manages the full lifecycle of trials, logging, and model saving.

The search space spans both architectural and optimization parameters, including:

- Learning rates (1e-5 to 1e-2 in log scale)
- Batch sizes (8, 16, 32, 64)
- Weight decay values (1e-5 to 1e-3 in log scale)
- Optimizer configurations (Adam, AdamW, SGD with momentum)
- Learning rate schedulers (StepLR, CosineAnnealing, ReduceLROnPlateau)
- Dropout rates (0.0 to 0.5)

Each trial's configuration and validation accuracy are recorded to ensure full reproducibility. The system also preserves the top five configurations per domain fold, including their trained weights and optimizer states.

Hyperparameter configurations are evaluated within each LODO fold using only source domains. Key elements of the evaluation procedure include:

- Early Stopping with patience=3 and $\delta = 0.001$
- Median Pruning after 5 warm-up trials
- Evaluation for each configuration over 20 epochs
- Tracking of validation accuracy as primary metric
- Maintenance of separate trial records per domain fold

After fold-wise optimization, results are aggregated to produce a robust global configuration. Performance patterns across folds are analyzed to identify parameter values that generalize consistently across domains. These consolidated results are stored in a YAML configuration file as a standardized reference for Model Training. This automation eliminates the need for manual transfer of parameters between experimentation and deployment phases, reducing overhead and enhancing reproducibility.

Domain relationships are preserved during the tuning process through index management, which ensures that domain shifts are correctly accounted for. Stratified sampling maintains class balance when splitting data into training and validation

sets to prevent bias towards dominant classes. In addition, domain-specific performance metrics are tracked across folds to enable analysis of how different hyperparameter combinations generalize across heterogeneous data distributions. The framework supports both models with MixStyle layers and models that rely on feature extraction for style augmentation.

Training Framework The model training pipeline is implemented using PyTorch and operates over domain-diverse datasets such as PACS or VLCS, supporting both architectures with and without MixStyle augmentation. Its primary training objective is to learn representations that generalize effectively across unseen visual domains, which is simulated again through the LODO cross-validation setup. Training relies on the globally optimized hyperparameters identified by the tuning pipeline, which are applied uniformly across domain folds to ensure experimental consistency. For the used hyperparameter configurations in the experiments, refer to Tables 9 and 10 in the appendix. The respective configurations were used for model training of the baseline with and without MixStyle.

For reproducibility, random seed control is enforced across multiple runs (seeds: 42, 7, 0). Using multiple seeds allows us to test the robustness of the models against random initialization and provides the foundation for later statistical significance analysis. All experiments are logged using TensorBoard for performance tracking (Abadi et al., 2015).

Each training run is based on a domain-aware dataset split into N folds, with N being the number of domains. In fold i, domain i is excluded for testing, while the remaining $N - 1$ domains are used for training and validation. Results are aggregated across folds to compute mean accuracy and standard deviation per run.

Training proceeds for 50 epochs, with evaluation after each epoch. Each epoch consists of forward and backward passes over batches, with each batch containing images, their class labels and respective domain indices. Due to our experimental setup, proper handling of the domain indices is critical. Thus, the data loaders use a custom collate function to handle the indices alongside their respective images and labels in a seamless integration with the model's forward pass. Validation and test sets are evaluated at the end of each epoch to assess generalization. The best performing model per fold is saved based on its validation accuracy. The framework tracks accuracy and loss across training, validation and test sets as well as per-domain performance. The TensorBoard integration logs scalar metrics for inspection (Abadi et al., 2015).

The framework includes a Visualizer class:

- Training Dynamics: Learning curves for training/validation loss and accuracy are plotted per fold to diagnose overfitting or optimization issues.
- Feature Space Analysis: t-SNE projections of ResNet block outputs reveal domain clustering and style invariance.

- Seed-overarching Results: Comparative plots for results across random seed runs.

The framework's modular design allows easy integration of new datasets (e.g., OfficeHome).

3.4 Feature Extraction

This section provides an analysis of the provided implementation for collecting, extracting, storing and applying per-domain feature statistics inside a ResNet by using PyTorch.

The code implements three cooperating components:

- DomainAwareHook
  This is a lightweight forward hook that transforms feature maps during inference by normalizing them per-channel per-sample and re-scales them with target domain statistics μ and σ. It can obtain those domain statistics either directly from a `StyleStatistics` object in-memory or by loading a `pth` file.

- StyleStatistics
  This is the module responsible for collecting, maintaining and exposing per-domain per-layer style statistics, which consist of channel-wise mean and standard deviation. It supports several collection methods and uses EMA with warm-up behavior to update the stored statistics.

- StyleExtractorManager
  A manager class that creates multiple `StyleStatistics` extractors with different modes. It attaches hooks to the model to collect statistics for specific domains by performing forward passes with dummy inputs, transfers statistics between extractor instances and persists statistics to disk.

The dominant concept is the extraction and management of per-domain channel statistics for feature maps. These statistics are used to adapt features at inference time.

In DA, style transfer and robustness experiments, it is often helpful to capture statistics of intermediate feature maps for different data domains (Huang and Belongie, 2017). The module implemented here computes the per-channel mean and standard deviation by averaging over the spatial dimensions of intermediate feature maps. It aggregates these statistics per domain and per layer, with optional selection for specific layers of the architecture and stores them in disk-friendly formats, to be re-applicable to feature maps during forward passes via hooks, which enables inference-time style adaptation. The extraction is happening after the finished model training, so the internal weights and other activations are already fine-tuned to the dataset used.

DomainAwareHook The hook encapsulates the runtime behavior needed to transform intermediate feature maps so that their per-channel statistics match a chosen target domain. It is intended to be attached to a ResNet-50 block and executed automatically during the model's forward pass. The hook can obtain target-domain statistics either directly from an in-memory `StyleStatistics` object or by loading a record from disk. This flexibility supports both interactive experiments and deployment scenarios where only files are available.

Given an input feature map of shape `[B, C, H, W]` the hook computes the sample-specific per-channel mean and standard deviation across spatial locations. It normalizes the features using these instance statistics and then re-scales and re-centers the normalized features using the stored statistics. This is the standard AdaIN-style transform applied at the selected network location to impose the target domain's 'style' on the incoming features while preserving per-sample content information.

The hook includes mechanisms to cope with differences between stored statistics and the current layer's channel dimensionality. If necessary it will up or down-sample the stored vectors so that broadcasting to `[1, C, 1, 1]` is possible. This allows the same persisted statistics to be re-used across slightly different model variants or extraction modes.

It is also used at inference time to replace features at particular ResNet layers with domain-specific statistics collected post-training.

StyleStatistics This module is the authoritative store and update engine for per-domain, per-layer channel statistics. It provides logic to initialize storage for different network layers, update running statistics based on batches of activations, retrieve aggregated statistics according to different strategies and (de-)serializes the state.

For each ResNet layer of interest the module holds a compact representation of per-domain statistics. Each stored vector represents the channel-wise mean or standard deviation aggregated over time and batches. The storage is structured so there is a separate row per domain per layer. The class supports multiple strategies for producing a domain-level statistic that can be applied later during TTA.

- Single-layer mode
  This mode provides the most granular style representation by isolating the statistical signature from a single network layer. All four ResNet blocks are available for extraction. In this mode, the model exclusively tracks and updates statistics for the chosen layer, disregarding the others. This approach is optimal for analyzing the distinct stylistic contribution of a specific hierarchical level of features.

- Selective mode
  The selective mode offers a multi-layer perspective by collecting statistics from a defined subset of layers. For each layer in the subset, the statistics are tracked and updated independently, but stored in a single tensor with a dedicated layer dimension to ensure efficient organization and access. This mode enables the

creation of a tailored style profile that balances specific feature hierarchies by combining the different structural patterns of different layers.

- Average mode
  This mode generates the most comprehensive and generalized domain signature by incorporating statistics from all available layers of the architecture. Independent features are tracked and maintained for every layer. Then follows the interpolation of each layer's feature vector to a common reference length, which are 256 channels in this case, before they are element-wise averaged to form a single feature vector. This results in a holistic style vector that is supposed to encapsulate the domain's characteristic patterns across all feature scales and semantic levels.

These modes allow a choice whether to apply a narrowly scoped style or a broader, multi-layer style signature.

During training, the style statistics of the domains and layers are updated with each forward pass. In this pipeline, the updates are performed using EMA with a configurable momentum. This works by reducing per-batch computed μ and σ values to summaries and blending them into the already stored statistics. The `StyleStatistics` module tracks the update counts per domain and layer so the system knows how many observations contributed to each stored value. This approach yields a compromise between reactivity to new observations and long-term stability of the stored statistics. The implementation also supports a warm-up period during which the effective momentum ramps up over the first N updates, which lets early observations shape the stored statistics more strongly. If a batch contains samples from multiple domains, the module can partition the batch by domain and perform per-domain updates in a vectorized way. This capability enables efficient collection of domain-specific statistics during a multi-domain training or evaluation pass. If incoming statistics have a different channel dimensionality than the existing storage for a layer, the module will initialize or reinitialize storage for the new dimensionality. This behavior makes the system robust to variations in the feature extractor.

In the *average* mode which combines statistics across layers, the module includes an alignment step that interpolates per-layer channel vectors to a common reference length before averaging. This is a pragmatic choice to produce a single, comparable signature from heterogeneous layer outputs as an approximation that trades some fidelity for interoperability between layers.

The module implements loader routines that are designed to accept slightly different on-disk layouts for storage. Statistics are saved in both human-readable files (in `JSON`-format) and tensor-preserving files that can be reloaded without loss of shape (`pth`-format). The module also can reconstruct the in-memory storage appropriately and map tensors to the desired device.

StyleExtractorManager The manager automates the end-to-end process of creating multiple `StyleStatistics` extractors, one per chosen aggregation mode, by

running inference passes to populate their storage from a trained ResNet-50 model and saving the resulting artifacts. This module provides a experiment-level interface so that multiple extraction strategies can be executed and persisted in a single controlled workflow.

The manager constructs a set of the `StyleStatistic` extractors, where each is independently configurable and can maintain its own training-domain mask which controls which domains it should collect statistics for. This is useful for training on multi-domain datasets with a LODO setup. The manager can load a saved ResNet-50 checkpoint and restore both the model weights and any embedded style statistic state saved with the checkpoint. During loading it separates model weights from style storage so that the latter can be inspected, transferred or reinitialized as needed.

For each domain and each extractor the manager attaches the domain-aware hooks to the targeted ResNet blocks, typically the last block of each layer. Then it performs a forward pass with a dummy input that is expected to trigger the model's internal statistics collection. Dummy inputs are enough in this case since the objective is to elicit typical activations of per-channel statistics in an already trained model. After the forward pass the hooks are removed. This controlled attach-execute-detach cycle ensures the statistics are gathered at the intended network locations and for the specific domain label. After gathering statistics from the model's own `StyleStatistics` instance or the live activations, the manager can transfer per-layer vectors into the per-mode extractors. This step consolidates raw collected values into the formats required by the different aggregation styles and update counters so downstream users are aware of how much data underlies each stored vector.

Feature Statistics As mentioned above, the statistics are obtained by using hooks in the internal model architecture. The per-sample per-channel feature mean is computed as the mean over the spatial axes for each channel, for each sample in the batch. The feature standard deviation is computed across the spatial axes. These operations yield tensors shaped `[B, C, 1, 1]`. To avoid division by zero in low-variance channels, there is an epsilon ($1e-6$) applied in the normalization step. This is the standard instance statistics procedure used by AdaIN and related methods by summarizing the per-sample style of a channel across spatial support (Huang and Belongie, 2017).

The aggregation of the feature statistics is handled by the `StyleStatistics` module. For a batch of samples from domain D and layer l the incoming μ and σ are averaged over the batch axis to produce the mean values of these statistics for the current update ($\mathtt{s}_{\mathtt{mean}}$). The EMA update uses these average statistics to calculate the new stored statistic $\mathtt{s}_{new}$; the same formula applies to both μ and σ:

$$\mathtt{s}_{\mathtt{new}} = \text{momentum} * \mathtt{s}_{\mathtt{old}} + (1 - \text{momentum}) * \mathtt{s}_{\mathtt{mean}}. \tag{15}$$

The warm-up updating logic scales the momentum linearly during early updates, which makes the early statistic rapidly adapt to incoming samples due to low momentum and becomes more stable later on when the momentum approaches the

configured value (0.9 in this pipeline).
The use of EMA for maintaining running statistics avoids recomputing exact statistics over the entire training set, which would be computationally expensive and memory-intensive. Instead, EMA provides a smoothed and memory-efficient approximation of the underlying distribution by gradually integrating information from new batches while retaining long-term statistics. This principle is similar to the running mean and variance updates used in Batch Normalization (Ioffe and Szegedy, 2015).

The `StyleStatistics` module supports multiple retrieval modes for obtaining stored feature statistics. For each domain d and layer $\ell \in \{0, 1, 2, 3\}$, the module maintains per-channel mean and standard deviation vectors

$$\mathbf{mean}_d^{(\ell)} \in \mathbb{R}^{C_\ell}, \qquad \mathbf{std}_d^{(\ell)} \in \mathbb{R}^{C_\ell},$$

where C_ℓ is the number of channels in layer ℓ. At runtime, the retrieval mode determines how these layer-wise vectors are combined into the final statistics (μ, σ) that are used to normalize and re-style features in the forward pass.
The single-layer mode retrieves statistics from a single, specified layer L without any modification:

$$\mu = \mathbf{mean}_d^{(L)}, \qquad \sigma = \mathbf{std}_d^{(L)}. \tag{16}$$

This is appropriate when the style is assumed to be localized to one network stage or when analyzing layer-specific domain characteristics.
The selective-layer mode operates on a predefined subset of layers $\mathcal{S} \subseteq \{0, 1, 2, 3\}$ but treats them independently. For each selected layer $\ell \in \mathcal{S}$, the corresponding stored statistics are applied when that layer's forward hook is triggered:

$$\mu = \mathbf{mean}_d^{(\ell)}, \qquad \sigma = \mathbf{std}_d^{(\ell)}. \tag{17}$$

This approach enables capturing style statistics that are distributed across specific regions of the network, rather than relying solely on a single layer.
The *average* mode aggregates information from all monitored layers $\mathcal{L} = \{0, 1, 2, 3\}$ to form a single global style signature. Because the layers have different channel dimensions C_ℓ, each vector is first interpolated to a common dimensionality C_{tgt}:

$$\widehat{\mathbf{mean}}_d^{(\ell)} = \text{Interp}\left(\mathbf{mean}_d^{(\ell)} \to C_{\text{tgt}}\right), \qquad \widehat{\mathbf{std}}_d^{(\ell)} = \text{Interp}\left(\mathbf{std}_d^{(\ell)} \to C_{\text{tgt}}\right). \tag{18}$$

The aligned vectors are then averaged element-wise across layers:

$$\mu = \frac{1}{|\mathcal{L}|} \sum_{\ell \in \mathcal{L}} \widehat{\mathbf{mean}}_d^{(\ell)}, \qquad \sigma = \frac{1}{|\mathcal{L}|} \sum_{\ell \in \mathcal{L}} \widehat{\mathbf{std}}_d^{(\ell)}. \tag{19}$$

This provides a holistic domain representation that smooths over layer-specific variations and captures network-wide activation patterns.

The choice of retrieval mode reflects a trade-off between preserving layer-specific information and capturing a generalized style signature. The single-layer mode

offers a high specificity by preserving the unique characteristics of a single network stage but potentially misses important style information from other layers. The selective-layer mode balances specificity and generality in theory, which should allow targeting style information from multiple layers without averaging over the entire network. The full-layer average mode maximizes generality by producing a unified style vector that captures overall domain characteristics at the cost of losing fine-grained layer distinctions.

3.5 Augmentation at Test-Time

This section provides a description of the TTA experiment framework for feature augmentation in the supplied code. The aim is to give a clear mental model of how the system is organized and how the pieces cooperate to carry out LODO TTA experiments.

The code implements an experiment pipeline for evaluating TTA strategies on multi-domain image classification tasks. It was designed to load pre-trained models and per-domain style statistics to apply domain-aware modifications to model activations at test time via forward hooks to mimic style transformations. The inference is run over one held-out test domain while adapting features from other domains as the augmentations, with the aggregation and computation of performance and uncertainty metrics.

Two cooperating classes structure the pipeline. The `TTAClassifier` encapsulates the model-level augmentation and prediction logic while `TTAExperiment` organizes the experiments across multiple test domains, modes and random seeds. Utility helpers for seed management and result serialization support reproducibility and result handling.

The `TTAExperiment` class is responsible for running experiments across the full set of domains of the used datasets, the configured modes and configured seeds. Its primary entry points are a method that iterates over the test domains, modes and seeds, calls another method for completion of the logic and logs per-seed and per-mode summaries by writing a timestamped text file for enabling of reproducibility for recorded metrics.

The pipeline centralizes seed setting in the `SeedManager` helper. This class sets seeds for the Python `random` module, NumPy and PyTorch, which provides a deterministic execution environment to the external libraries permit, and is invoked at both experiment initialization and per-seed runs to isolate seed effects.

The system assumes a pre-trained ResNet model and checkpoint files organized by seed and test domain, as made available by the training process and the style extraction functionality. The style statistics for individual domains and layers are stored as mentioned above as PyTorch objects, which can be loaded and used in the `DomainAwareHooks` for the style adaptation.

The `TTAClassifier` wraps an existing model and provides the mechanisms required for TTA and prediction aggregation. Its main responsibilities include the loading

and integration of the extracted style statistics for training the source domains with respect to a held-out test domain; the registration of forward hooks to capture the intermediate activations and dynamic installation of domain-aware hooks during per-domain inference. The classifier performs inference across the specific test set while applying the domain adaptation hooks and collecting as well as aggregating predictions and metrics for analysis.

The classifier accepts a mode argument (introduced in section 3.3) that configures which layers will be used for the TTA process. The supplied model is moved to the configured device, all parameters are frozen and the evaluation mode is called. The classifier creates a new linear head where its feature dimensions are estimated by forwarding a dummy tensor through the model to receive a feature vector. The forward hooks are registered on every 2D convolutional layer in the model to capture the output activation tensors.

The main logic is orchestrated by the `predict()` method. It creates a list of all available source domains, excluding the configured test domain. These source domains are the target directions for the augmentation process, in which the images of the excluded domain are adapted towards. For each batch of the dataloader containing the test samples, it computes an original prediction without augmentation and then dynamically constructs and registers `DomainAwareHook` instances for each source domain targeted at the specified layers for the experiment. It then performs a forward pass to obtain the domain-adapted logits and probabilities. The hooks are removed after obtaining the domain-adapted outputs. Across the source domains it computes the prediction accuracy and per-sample variance of per-class probabilities. These metrics are introduced in section 4.2.1. For each source domain used as an adaptation target, the logic stores the logits, probabilities and predictions and then computes accuracy as well as the additional measures. The metrics are used to quantify how sensitive the model's beliefs are to the style transformation space induced by the training domains. A non-augmented 'original' forward pass over the test set provides the baseline predictions and class variance for comparison. For each test domain and mode, the pipeline records per-seed metrics for every augmentation target. All results are returned in a structured dictionary and written to disk.

Finally, `TTAExperiment` orchestrates the LODO protocol by looping over all test domains, all configured modes and the specified random seeds. It restores the appropriate checkpointed ResNet-50 for each seed/test-domain pair, constructs the test loader and instantiates the `TTAClassifier`. Deterministic behavior is enforced as far as the libraries allow via the centralized `SeedManager` to ensure that results are repeatable across execution runs and seed effects can be reported isolated.

3.6 Workflow

The overall workflow of the proposed system integrates the previously described components into a coherent pipeline for DG and TTA. The process can be divided into two main phases: Model training with style extraction and evaluation with style augmentation (see Fig. 1).

In the training phase, images from the source domains are processed by the ResNet-50 backbone to extract hierarchical feature representations. Forward hooks are attached to selected residual blocks to enable the collection of per-channel statistics (mean and standard deviation) of intermediate activations. The aggregated statistics serve two purposes. First, they capture domain-specific style signatures that can later be re-applied at inference and second, they provide a basis for analyzing inter-domain variability. Simultaneously, the extracted feature embeddings are fed into a task-specific classifier, which is trained to minimize cross-entropy loss on the source domains. The LODO protocol ensures the full exclusion of one domain during training, which allows the system to simulate realistic domain shift.

In the evaluation phase, the pipeline performs TTA by reintroducing the style statistics collected during training. For each test image from the unseen domain, the frozen ResNet backbone is traversed while forward hooks dynamically apply domain-aware transformations. Specifically, the `DomainAwareHook` normalizes activations per sample and re-scales them with style statistics from one of the source domains, which projects the test sample into the style space of that domain. This procedure is repeated across all available source domains, resulting in multiple domain-adapted feature representations for the same test input. The classifier collects predictions from these augmented passes for aggregation and computes the additional uncertainty metric like variance. The final prediction is derived by combining the augmented outputs with the unmodified baseline prediction.

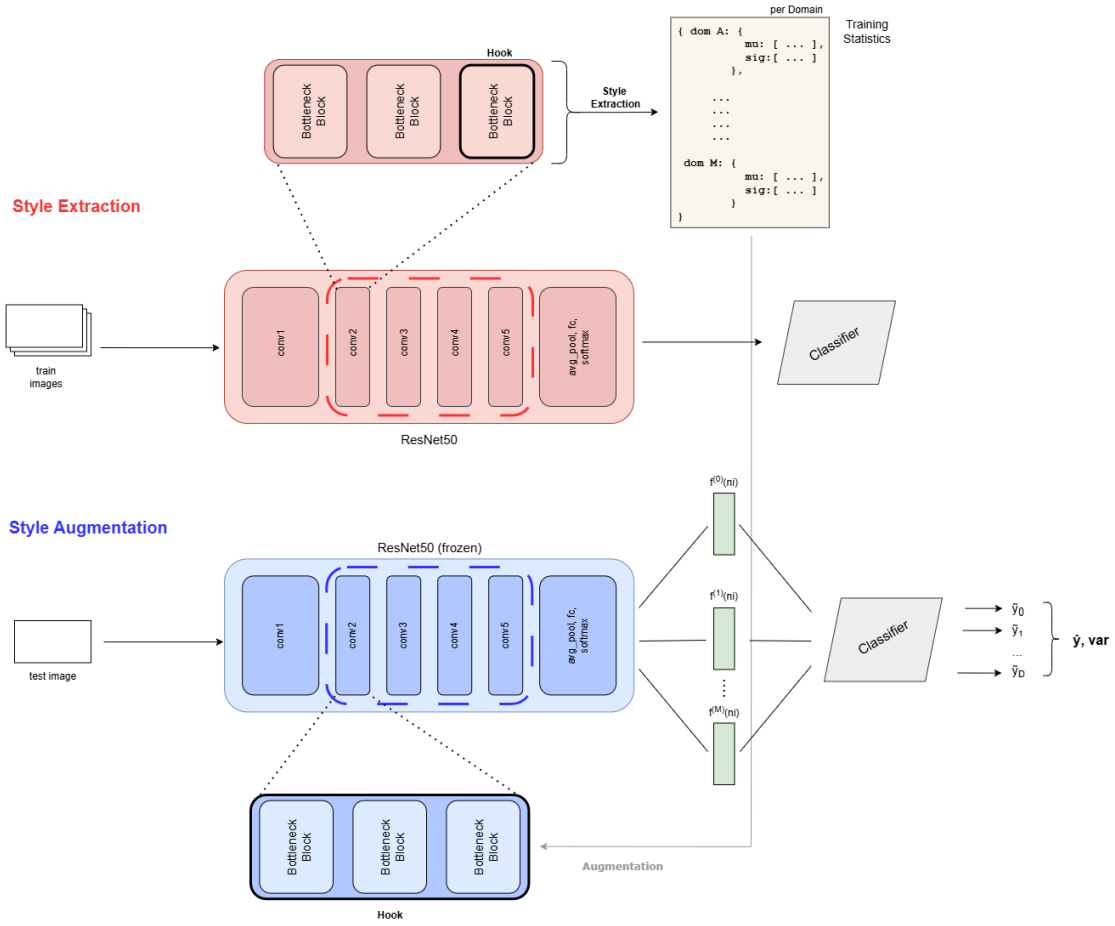


Figure 1: Stylized Pipeline

This pipeline establishes a process where domain-specific statistics are first extracted during training and subsequently re-applied during evaluation to enable style-driven TTA. This design ensures both reproducibility and generalization across unseen domains as well as provides interpretable measures of robustness under domain shift.

4 Experiments & Results

4.1 Datasets

This section describes the datasets used in the experiments.

4.1.1 PACS

The PACS dataset is a widely adopted benchmark for evaluating DG algorithms, designed to test model robustness under cross-domain distribution shifts (Li et al., 2017).

It consists of four distinct stylistic domains. The *Photo* domain uses real-world photographs with natural textures and lighting, whereas the *Art_painting* domain includes artistic renditions such as paintings or illustrations with varied brushstrokes and color palettes. The less realistic domains *Cartoon* and *Sketch* consist of stylized drawings with bold outlines and flat shading as well as black-and-white line drawings with minimal texture respectively. This diversity requires models to learn style-invariant features, as domain shifts are primarily stylistic rather than semantic (Zhou et al., 2022a). Rather unequally distributed across these domains are the seven classes of the dataset. PACS includes these object categories: *dog, elephant, giraffe, guitar, horse, house* and *person*. It is worth noting that these classes are semantically consistent across domains but exhibit stylistic divergence within each class. E.g., a *guitar* appears differently in a photo than in a sketch but shares the same characteristics in both domains.

The dataset contains 9991 images in total, with an uneven distribution across domains. While the *Sketch* domain contains 3929 images, *Photo* only consists of 1670 images. *Cartoon* and *Art_painting* contain 2344 and 2048 images respectively.
Figure 2 shows examples of every class in every domain of the dataset.

4.1.2 VLCS

The VLCS dataset is another widely used benchmark for evaluating DG algorithms, first introduced by Fang et al. (2013). It combines image data from four distinct sources, *VOC2007, LabelMe, Caltech101* and *SUN09*, each representing a separate domain. While all domains consist of real-world photographs, they vary in composition, background complexity and perspective. For example, *Caltech101* typically includes object-centric images with minimal background clutter, whereas the others contain more complex scenes with variable lighting and context (Fang et al., 2013).

Across all domains, the dataset shares five object categories: *bird, car, chair, dog* and *person*. These classes remain semantically consistent throughout the domains but differ stylistically. For instance, while a *car* appears in each domain as a real object, the backgrounds, occlusions and angles differ, which forces models to learn representations that are invariant to these domain-specific features.

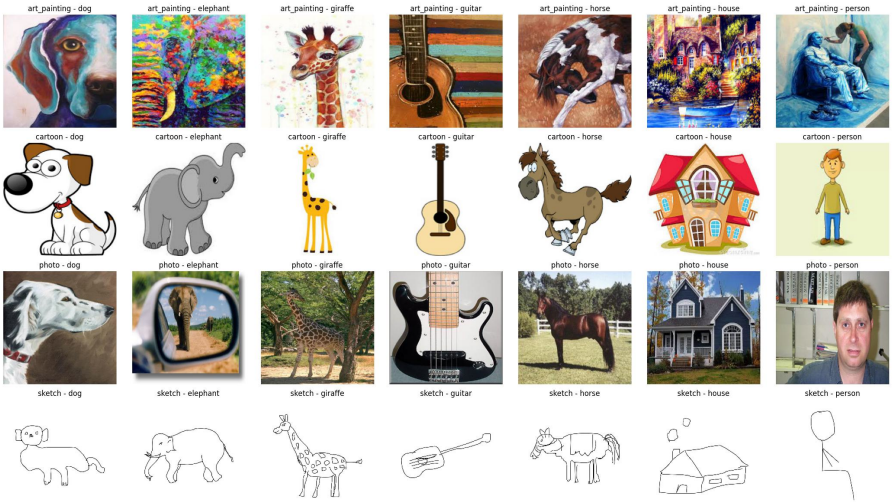


Figure 2: Exemplary images from the PACS dataset, containing images from the classes *dog, elephant, giraffe, guitar, horse, house* and *person*, each from the four domains *Art_painting, Cartoon, Photo* and *Sketch*.

The dataset contains a total of 10729 images with an imbalanced distribution across domains: while both *VOC2007* and *SUN09* consist of more than 3000 examples (3376 and 3282 images respectively), *LabelMe* and *Caltech101* contain significantly less data (2656 and 1415 images respectively). These differences in domain size introduce additional challenges during training and evaluation, as the models must generalize under data scarcity and imbalance.

During dataset preparation, four images from the VLCS dataset were identified as corrupted during the download process and consequently excluded from the experimental setup. This exclusion resulted in a final curated dataset comprising 10725 valid images for analysis. The removal of these corrupted samples ensured data integrity while maintaining a statistically robust sample size for all subsequent experiments.

Figure 3 shows examples from every class of every domain in the dataset.

4.1.3 Domain and Class Structure Analysis

Both the PACS and VLCS datasets were selected for this study due to their complementary strengths in evaluating DG algorithms. Their defined domain boundaries allow for precise assessment of model robustness against distribution shifts, where PACS emphasizes stylistic variation across artistic modalities and VLCS highlights real-world diversity arising from different data sources.

As established community benchmarks, both datasets facilitate meaningful comparison with prior DG approaches. Moreover, their inherent class and domain imbalances reflect challenges commonly found in practical settings, such as heterogeneous

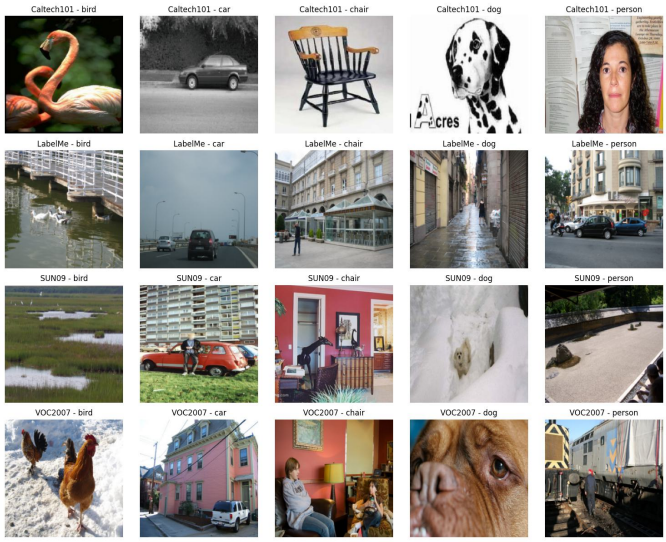


Figure 3: Exemplary images from the VLCS dataset, containing images from the classes *bird, car, chair, dog* and *person*, each from the four domains *Caltech-101, LabelMe, SUN09* and *VOC2007*.

data distributions and uneven sample availability (Choi et al., 2024; Gulrajani and Lopez-Paz, 2020).

To gain an intuition for the structure of the raw datasets, we apply t-distributed Stochastic Neighbor Embedding (t-SNE), a nonlinear dimensionality reduction technique that maps high-dimensional data into a two-dimensional space while preserving local neighborhood relationships (van der Maaten and Hinton, 2008). Figure 4 and figure 5 show the resulting embeddings when color-coded by domain on the left and by class on the right.

The domain-based visualization reveals a clear separation between the stylistic groups for the PACS dataset in figure 4. Particularly, the *Sketch* domain forms a distinct cluster, while *Photo* and *Art painting* exhibit partial overlap but remain distinguishable. In contrast, the class-based visualization shows considerable overlap across categories and highlights that stylistic variance dominates the embedding structure more strongly than semantic variance.

In contrast to PACS, where stylistic differences dominate the structure, the visualization for the VLCS dataset (see Fig. 5) shows a substantial overlap between the four source domains. The only limited separation visible is for the subset *Caltech101*, which forms more compact clusters. The class-wise visualization shows, like with PACS, a high entanglement across domains without clear boundaries in the embedding space.

These observations are highly relevant for TTA strategies that augment with feature statistics. If style drives the primary shifts in representation, then adapting feature distributions to better align with unseen target domains may mitigate domain gaps without harming class semantics. The overlap for VLCS suggests that the TTA might have to adapt to subtler shifts in distribution caused by scene complexity.

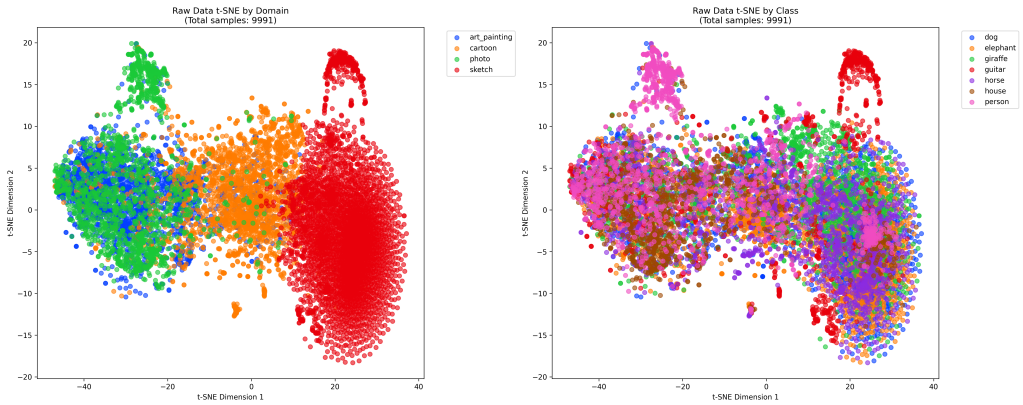


Figure 4: t-SNE visualization of the raw PACS image tensors. Left: samples color-coded by domain. Right: samples color-coded by class.

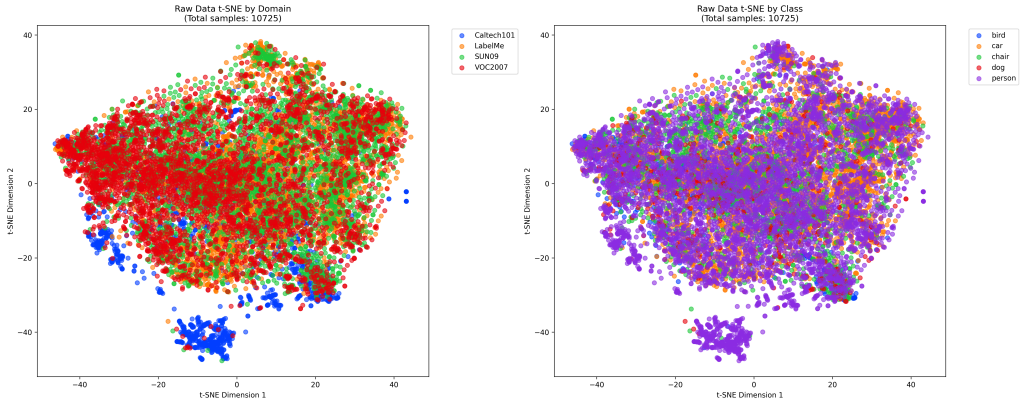


Figure 5: t-SNE visualization of the raw VLCS image tensors. Left: samples color-coded by domain. Right: samples color-coded by class.

4.2 Metrics & Tests

This section describes the metrics and statistical means used for evaluation in the project.

4.2.1 Metrics

TTA alters a trained model's behavior at inference to reduce the impact of domain shift between training and deployment. Because TTA changes both the model's decisions and its confidence under transformed inputs, the evaluation of such a system cannot rely solely on accuracy, which quantifies task correctness. Accuracy does not reveal whether a model's predictions are stable under the applied test-time transformations or whether the model's confidence is reliable.

To characterize the effects of TTA we therefore use variance as a complementary stability metric alongside accuracy to measure how the predicted probability distribution fluctuates across augmentations. Variance exposes changes in model confi-

dence that can indicate calibration or robustness issues even when the top-1 level remains constant.

Accuracy Let N denote the number of evaluated test samples and let $y_i \in \{1, \ldots, K\}$ be the ground truth label of sample i. Let $\hat{y}_i$ denote the predicted label for sample i (a single label per sample, such as the model's top-1 prediction after any aggregation). The accuracy over the dataset is defined as

$$\text{Accuracy} = \frac{1}{N} \sum_{i=1}^{N} 1\{\hat{y}_i = y_i\}, \tag{20}$$

where $1\{\cdot\}$ is the indicator function that equals 1 if the argument is true and 0 otherwise.

The TTA pipeline introduced in this work computes an original prediction for each input and stores these as $\hat{y}_i^{orig}$. This equals the accuracy for the model forward pass without augmenting towards the target domains. The pipeline also computes domain-adapted predictions via domain-aware forward passes with one forward per target domain acting as an augmentation, which yields domain-specific predicted labels $\hat{y}_i^{(a)}$ for augmentation a. The implementation concatenates predicted labels and ground-truth labels across batches and computes accuracy by taking the mean of equality tests (`(preds == labels).mean()`). When experiments are repeated with multiple random seeds, the experiment runner aggregates final accuracies across seeds and reports summary statistics for the sample mean and standard deviation of accuracy across seeds.

Accuracy measures task correctness, the fraction of examples for which the model's chosen label equals the ground truth. The comparison between the original accuracy for images without feature augmentation and TTA-derived accuracy indicates whether the chosen method is beneficial for predictive performance on the test domain. An increase in accuracy after TTA indicates an improved task performance, whereas a decrease suggests a less accurate classifier decision, e.g. by misaligned features or over-correction.

Per-sample variance Let $p_{i,y_i}^{(a)}$ denote the predicted probability assigned to the true class y_i of sample i under augmentation a where $a = 1, \ldots, A$. For a given sample i, the augmentation-wise mean probability of the true class is defined as:

$$\mu_{i,y_i} = \frac{1}{A} \sum_{a=1}^{A} p_{i,y_i}^{(a)}. \tag{21}$$

The TTA pipeline extracts the probability associated with the correct class y_i for each sample and augmentation, computes its variance across domain adaptations and then averages across all samples for class variance.

The Variance measure here therefore measures confidence instability. It quantifies how much the entire predicted distribution changes across augmentations, not just

the top-1 label. Low variance indicates consistent confidence, which shows a stable probability mass of the model across augmentations. High variance shows a strong probability fluctuation with the augmentations, suggesting a sensitivity of the model to the TTA transformations. Large mean variance across the dataset suggests poor robustness of confidence estimates to domain-adaptive transform; this motivates methods that either stabilize probabilities or use aggregation strategies that reduce variance.

Evaluating TTA requires measuring not only whether the model predicts the correct label, but also how stable those predictions and their associated confidences are under the applied transformations. The presented metrics are able to give a multi-dimensional view of model behavior.

Accuracy as the primary performance metric can be misleading in TTA contexts when used alone. It does not indicate whether the model's predictions are consistent across the different augmentations that the TTA mechanism produces.

Beyond point estimates of variance, we quantify how well per-sample variance acts as a usable uncertainty metric signal by studying accuracy drop curves based on uncertainty ranking and by reporting three scalar summaries, Area Under Accuracy Drop (AUAD), Gain and Area Under Curve (AUC). Let u_i denote the uncertainty score for sample i, derived from the augmentation-wise variability of its predicted probabilities. Unless stated otherwise, we use the above defined per-sample variance.

For the accuracy drop curves, let $\pi \in [0, 1]$ be a removal rate. Sort the evaluation set by u_i in descending order and remove the top π fraction, which represent the most uncertain samples. The retained accuracy is

$$\mathrm{Acc}_{\mathrm{unc}}(\pi) = \frac{1}{|\mathcal{I}_\pi|} \sum_{i \in \mathcal{I}_\pi} 1[\hat{y} = y], \quad \mathcal{I}_\pi = \{i : i \text{ not in the top } \pi \text{ fraction by } u_i\}. \tag{22}$$

We compare the curve to a random removal baseline $\mathrm{Acc}_{\mathrm{rand}}(\pi)$ obtained by repeatedly (e.g., 10^3 trials) removing a fraction of π samples uniformly at random and averaging the resulting accuracies.

We summarize $\mathrm{Acc}_{\mathrm{unc}}(\pi)$ by its area

$$\mathrm{AUAD}_{\mathrm{unc}} = \int_0^1 \mathrm{Acc}_{\mathrm{unc}}(d\pi), \quad \mathrm{AUAD}_{\mathrm{rand}} = \int_0^1 \mathrm{Acc}_{\mathrm{rand}}(d\pi), \tag{23}$$

approximated via the trapezoidal rule on a fixed grid $\pi \in \{0,\ 0.05,\ \ldots\ ,\ 0.95\}$. Larger $\mathrm{AUAD}_{\mathrm{unc}}$ indicates a steeper improvement when abstaining on high-variance samples.

To quantify the added value of variance over uninformed abstention, we report

$$\mathrm{Gain}_{\mathrm{abs}} = \mathrm{AUAD}_{\mathrm{unc}} - \mathrm{AUAD}_{\mathrm{rand}}, \quad \mathrm{Gain}_{\mathrm{rel}} = \frac{\mathrm{AUAD}_{\mathrm{unc}} - \mathrm{AUAD}_{\mathrm{rand}}}{\mathrm{AUAD}_{\mathrm{rand}}}. \tag{24}$$

$\mathrm{Gain}_{\mathrm{abs}}$ measures absolute improvement in the area, $\mathrm{Gain}_{\mathrm{rel}}$ contextualizes it as a percentage over the random baseline.

We additionally treat u_i as a score for identifying errors and compute the Area Under the Receiver Operating Characteristic Curve (ROC), which is denoted in this work as AUC, between u_i and the error indicator $e_i = \mathbb{1}[\hat{y}_i \neq i]$:

$$\text{AUC} = \text{ROC} - \text{AUC}(\{(u_i, e_i)\}_i).$$

An AUC close to 1 indicates that higher variance tends to occur on misclassified samples, whereas 0.5 corresponds to chance. All these quantities are computed per test domain and TTA mode.

4.2.2 Tests

To evaluate the performance of different adaptation strategies, we conducted a series of statistical analyses based on linear mixed-effects models (Jiang, 2007). The aim of this evaluation was to determine whether various TTA modes provide a measurable improvement in accuracy compared to the baseline method, while accounting for the variability introduced by different domains and random seeds, as claimed as a research objective in section 3.1

While paired t-tests could in principle be used to compare accuracies across modes, they are limited to analyzing one contrast at a time and assume independence of observations (Yu et al., 2022). In this setting, multiple sources of variability exist simultaneously: there are differences across domains and random seeds as well as repeated measures within domain-seed combinations. A linear mixed-effects model is therefore more appropriate in this context, as it allows the inclusion of fixed effects, which are the systematic differences between modes, and the modeling of random effects, which are present by uncontrolled variation due to different domains and seeds (Jiang, 2007). It also makes the simultaneous estimation of multiple contrasts within one unified framework possible. This approach increases the statistical power and provides a more reliable quantification of uncertainty compared to running a large number of separate paired tests.

Before using the linear mixed-effects model, we defined categorical variables for the experimental factors (approach, mode) and introduced a combined `domain_seed` identifier. This ensured that the subsequent models could properly account for repeated measures within the same domain and seed.

Two main model specifications were estimated:

- **TTA-only analysis**. The results were restricted to runs under the TTA approach. A mixed-effects model was fitted with mode as a fixed effect, random intercepts for domains and additional random variability at the `domain_seed` level. This estimates whether different TTA modes significantly differ in accuracy.
- **Baseline comparison**. The MixStyle baseline (base) was compared against selected TTA modes. By using base as the reference category, the model estimated the expected difference in accuracy between each TTA mode and the baseline, while controlling for domain and random seed.

Coefficients and their confidence intervals were extracted to provide interpretable estimates of the expected accuracy differences. Positive coefficients indicate improvements relative to the baseline. Confidence intervals that exclude zero suggest statistically significant differences at the 95% level.

Beyond the global models, paired differences per domain and seed were computed. For each domain, accuracy differences between a TTA mode and the baseline were calculated per seed. From these differences, mean improvements, standard deviations, standard errors, and t-based 95% confidence intervals were derived. This domain-level analysis illustrates whether improvements hold consistently across individual domains and seeds or whether gains are specific to certain cases. Narrow confidence intervals suggest stable effects while wide intervals reflect variability. With wide confidence intervals the effects of the respective augmentation style appears to be unstable and may depend heavily on which seed is used.

4.3 Experiments

The experimental evaluation investigates whether domain-aware TTA can improve model robustness under distribution shifts. We use the community benchmarks PACS and VLCS, as introduced in section 4.1, which differ in the nature of their domain shifts. Whereas PACS emphasizes stylistic variation across artistic modalities, VLCS highlights real-world diversity across different sources. For each dataset, we follow the LODO protocol (see section 3.3), training on all but one domain and treating the remaining domain as unseen test environment.

During inference, we apply domain-aware feature transformations based on stored style statistics. These consist of channel-wise means and standard deviations and are extracted from the training domains at multiple ResNet layers using hooks. At test time, they are injected into the forward pass to simulate different domain styles, producing multiple augmented predictions per input. We evaluate several augmentation strategies, namely *single layer*, *selective multi-layer* and *averaging* modes, which differ in how layer-level statistics are selected and combined (see section 3.4). We use statistics from all four ResNet-50 convolutional blocks for the modes.

Each experiment is repeated with three random seeds to account for initialization variability. For every test domain, we report mean accuracy as well as other values like AUAD, Gain and AUC values, which are derived from our uncertainty measure per-sample variance. While mean accuracy is reported for all TTA modes in order to provide a complete picture of classification performance, the evaluation of uncertainty quality is restricted to the three best-performing modes per dataset. This choice avoids redundancy and highlights the configurations most relevant for selective prediction.

Statistical analyses are performed using linear mixed-effects models to compare TTA modes with the baseline model without style adaptation, and with usage of the MixStyle approach (Zhou et al., 2021), while controlling for variability introduced by domains and seeds. This comprehensive setup allows the assessment of both task

performance and prediction stability for a thorough evaluation of TTA effectiveness across heterogeneous distribution shifts.

4.4 Results

4.4.1 TTA Approach

The evaluation of the proposed TTA approach compared to the ResNet-50 baseline trained with MixStyle (Zhou et al., 2021) shows dataset- and mode-dependent outcomes. Tables 1 and 3 show mean accuracy results for the datasets used in this project. The reported values were obtained by averaging the results of TTA toward the respective training domains across three random seeds. The baseline values were also calculated by averaging the test accuracies for the respective models in the LODO split.

PACS For the PACS dataset (see Tab. 1), several configurations achieve accuracies that are higher than the baseline. The *single_1* mode provides the most consistent improvements, with mean accuracies of 87.47% for *Art Painting*, 79.57% for *Cartoon* and 78.8% for *Sketch*, each exceeding the corresponding baseline results of 85.75%, 77.7% and 73.84%. In the *Photo* domain, *single_1* remains close to the baseline (96.86% vs. 96.99%). The *single_0* configuration also reaches competitive results, e.g., yielding 97.11% in *Photo*, which is slightly higher than the baseline. In contrast, modes involving *single_3* and their selective combinations result in considerably lower accuracies across domains, often in the range of $12-16\%$. The extended breakdown across target domains (see Tab. $11-$ 14 in the appendix) illustrates this variation. In the *Cartoon* domain, mode *selective_0_2* reaches 80.63% accuracy compared to the baseline of 77.7% (see Tab. 12) while in the *Sketch* domain, *single_1* obtains 79.18% relative to the baseline of 73.84% (see Tab. 14), both when augmented towards the *Photo* domain. These observations indicate that improvements are attainable, but outcomes depend strongly on the specific configuration and test target domain.

In addition to accuracy, the quality of the uncertainty estimate variance is evaluated using AUAD, Gain and AUC measures (see Tab. 2). The AUAD values are consistently high, with all domains except *Sketch* exceeding 90%, and *Sketch* still achieving values above 80%. This indicates that the proposed method consistently produces accuracy-drop curves that remain well above the baseline and that discarding the samples with the highest estimated sample variance substantially improves the accuracy of the remaining predictions. This behavior can be clearly observed in the drop curve plots (Figs. $6-9$), where the blue variance-based curve rises much more steeply than the orange random baseline as the most uncertain samples are discarded.

The observed gains quantify this improvement over random dropping. For domain *Art Painting*, the gains reach about 9.5%, while in *Cartoon* and *Sketch* are slightly

Table 1: Mean accuracy results in % for the PACS dataset in comparison to the Baseline results of a ResNet-50 model trained with MixStyle (Zhou et al., 2021)

Mode	**Mean accuracy across seeds for Test Domain**			
	Art Painting	Cartoon	Photo	Sketch
single_0	86.83 ± 0.82	76.46 ± 2.2	$\mathbf{97.11 \pm 0.46}$	72.74 ± 5.09
single_1	$\mathbf{87.47 \pm 1.09}$	$\mathbf{79.57 \pm 1.81}$	96.86 ± 0.45	$\mathbf{78.8 \pm 2.41}$
single_2	83.06 ± 2.23	79.1 ± 1.24	95.93 ± 0.71	71.49 ± 4.51
single_3	13.51 ± 4.79	15.95 ± 2.08	15.04 ± 6.05	11.98 ± 7.65
selective_0_1	86.94 ± 1.26	78.71 ± 1.99	96.69 ± 0.43	77.16 ± 3.42
selective_0_2	83.45 ± 2.49	79.41 ± 1.26	95.48 ± 0.8	72.84 ± 3.11
selective_0_3	13.51 ± 4.79	15.95 ± 2.08	15.04 ± 6.05	11.98 ± 7.65
selective_1_2	82.77 ± 2.47	79.45 ± 1.34	94.98 ± 1.09	73.19 ± 4.76
selective_1_3	13.51 ± 4.79	15.95 ± 2.08	15.04 ± 6.05	11.98 ± 7.65
selective_2_3	13.51 ± 4.79	15.95 ± 2.08	15.04 ± 6.05	11.98 ± 7.65
average	16.49 ± 2.85	16.6 ± 0.0	11.26 ± 0.35	9.26 ± 7.34
Baseline with MixStyle	85.75 ± 1.77	77.7 ± 2.53	96.99 ± 0.69	73.84 ± 4.12

Table 2: Summary of results per domain and top-3 modes on PACS. Accuracy, AUAD, and Gain are reported in % across seeds, AUC is reported in [0,1].

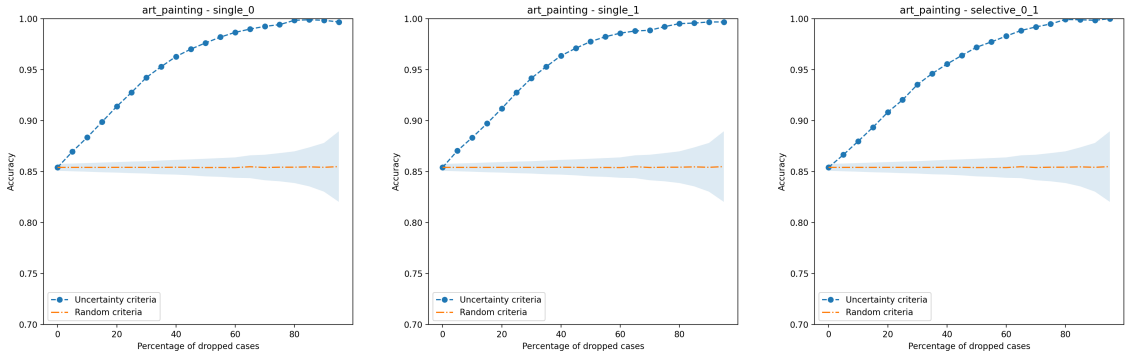
Domain	**Mode**	**Accuracy**	**AUAD**	**Gain**	**AUC**
Art Painting	single_0	86.83 ± 0.82	90.82 ± 0.05	9.67 ± 0.30	0.824 ± 0.006
	single_1	87.47 ± 1.09	90.74 ± 0.14	9.58 ± 0.45	0.822 ± 0.011
	sel_0_1	86.94 ± 1.26	90.50 ± 0.08	9.35 ± 0.36	0.806 ± 0.008
Cartoon	single_0	76.46 ± 2.20	84.02 ± 0.66	11.00 ± 0.52	0.721 ± 0.003
	single_1	79.57 ± 1.81	86.02 ± 0.52	13.00 ± 0.81	0.785 ± 0.011
	sel_0_1	78.71 ± 1.99	85.46 ± 0.44	12.43 ± 0.87	0.768 ± 0.012
Photo	single_0	97.11 ± 0.46	94.78 ± 0.03	2.13 ± 0.07	0.932 ± 0.008
	single_1	96.86 ± 0.45	94.76 ± 0.04	2.11 ± 0.09	0.929 ± 0.007
	sel_0_1	96.69 ± 0.43	94.77 ± 0.02	2.13 ± 0.08	0.929 ± 0.004
Sketch	single_0	72.74 ± 5.09	78.83 ± 3.23	8.24 ± 3.04	0.640 ± 0.062
	single_1	78.80 ± 2.41	82.89 ± 2.41	12.30 ± 0.63	0.740 ± 0.023
	sel_0_1	77.16 ± 3.42	81.04 ± 3.46	10.45 ± 1.54	0.694 ± 0.047

sel_0_1 = selective_0_1

higher at $11 - 13\%$ (Tab. 2). These values match the visibly larger gap between the uncertainty and random curves in Figs 7 and 9. In contrast, the *Photo* domain,

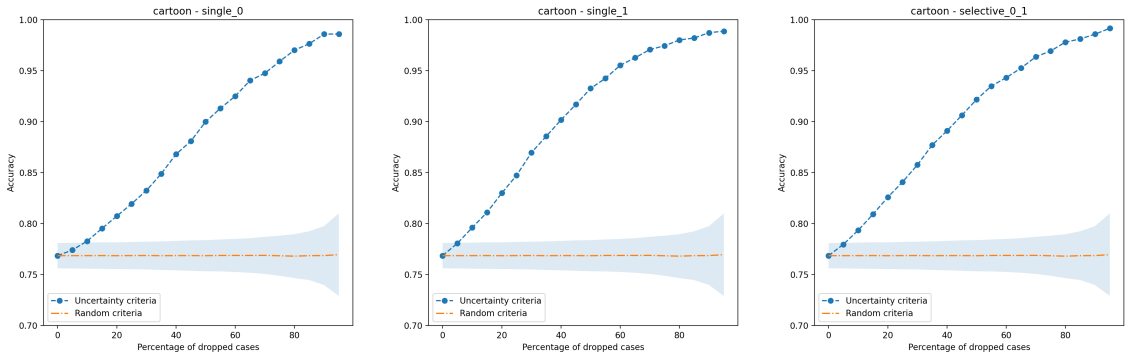
where accuracies are already close to 97%, shows only marginal gains of around 2%, as there is little room left for selective prediction to improve performance (see Fig. 8).

The AUC values provide a complementary view by directly assessing the ability of our second metric variance to discriminate between correct and incorrect predictions. In *Art Painting* and *Cartoon*, AUC values of about 0.82 and 0.78 confirm that variance is consistently informative (see Tab. 2). For the *Sketch* domain, AUC values around $0.64 - 0.74$ indicate a moderate discriminative ability. The AUC values for *Photo* peak at around 0.93, showing that variance is strongly aligned with prediction correctness in this domain.



(a) *single_0* (b) *single_1* (c) *selective_0_1*

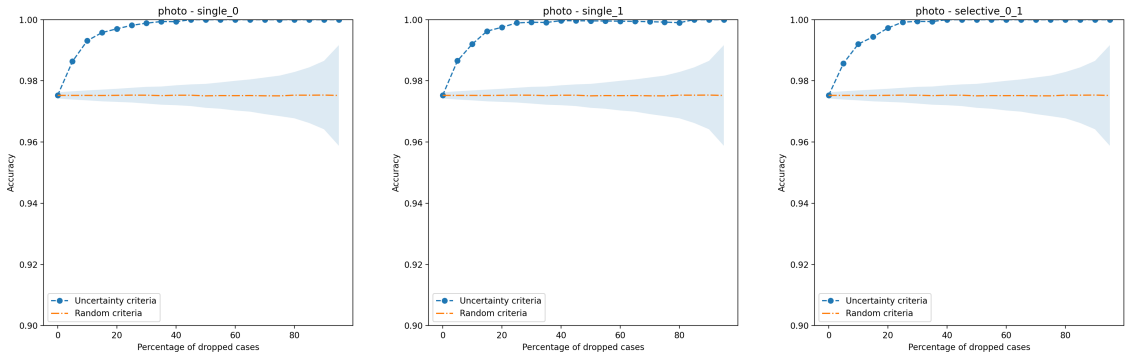
Figure 6: Dataset PACS, Test Domain *Art Painting*



(a) *single_0* (b) *single_1* (c) *selective_0_1*

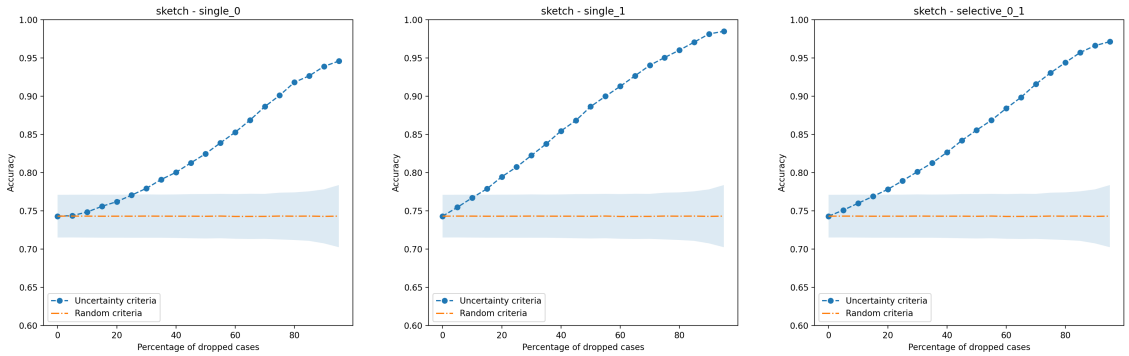
Figure 7: Dataset PACS, Test Domain *Cartoon*

VLCS For the VLCS dataset (Tab. 3), the baseline generally attains higher or comparable accuracies across domains. For *Caltech101*, the baseline reaches 97.97%, with *single_0* and *single_1* producing similar values of 97.6% and 96.51%, respectively. In *VOC2007*, mode *single_1* achieves 77.08%, which is slightly above the baseline of 76.02%. For *LabelMe* and *SUN09*, however, the TTA modes remain



(a) *single_0* (b) *single_1* (c) *selective_0_1*

Figure 8: Dataset PACS, Test Domain *Photo*



(a) *single_0* (b) *single_1* (c) *selective_0_1*

Figure 9: Dataset PACS, Test Domain *Sketch*

below the baseline: the best performing configurations reach 63.79% and 70.28%, while the baseline achieves 64.91% and 72.37%, respectively. The more detailed per-domain results (see Tab. 15−18 in the appendix) provide further insight into these patterns. While certain modes approach or slightly exceed the baseline in individual cases, other configurations yield substantially lower values, with accuracies around $35 - 46\%$ in some instances.

The VLCS dataset presents a more heterogeneous picture in the evaluation of AUAD, Gain and AUC (see Tab. 4). For *Caltech101*, AUAD values approach 95%, with small positive gains of around 1.6%. This is also visible in the drop curve plots (Fig. 10), where the uncertainty-based curve consistently outperforms the random baseline, but the gap remains narrow due to the already very high base accuracy ($\approx 97\%$). The corresponding AUC values between 0.88 and 0.91 underline that variance is a reliable error signal in this domain. By contrast, *LabelMe* shows clear weaknesses. AUAD values are close to 58% and the gains are negative (-3 to -4%). In the drop curves (Fig. 11), the uncertainty-based line falls below the random baseline, illustrating that variance fails to guide selective prediction and

Table 3: Mean accuracy results in % for the VLCS dataset in comparison to the Baseline results of a ResNet-50 model trained with MixStyle (Zhou et al., 2021)

Mode	**Mean accuracy across seeds for Test Domain**			
	Caltech101	LabelMe	SUN09	VOC2007
single_0	97.6 ± 0.44	63.28 ± 0.68	70.28 ± 1.23	77.02 ± 0.69
single_1	96.51 ± 1.04	63.79 ± 0.88	70.13 ± 0.86	$\mathbf{77.08 \pm 0.86}$
single_2	90.15 ± 3.86	62.73 ± 2.06	61.2 ± 4.42	70.06 ± 4.47
single_3	55.62 ± 16.6	46.57 ± 0.0	35.14 ± 4.77	39.14 ± 9.85
selective_0_1	95.78 ± 1.27	63.22 ± 0.98	69.6 ± 0.77	76.45 ± 0.87
selective_0_2	88.21 ± 3.97	61.21 ± 2.37	58.94 ± 4.19	67.76 ± 4.34
selective_0_3	55.62 ± 16.6	46.57 ± 0.0	35.14 ± 4.77	39.14 ± 9.85
selective_1_2	86.23 ± 4.67	61.67 ± 2.64	58.0 ± 4.42	66.68 ± 4.3
selective_1_3	55.62 ± 16.6	46.57 ± 0.0	35.14 ± 4.77	39.14 ± 9.85
selective_2_3	55.62 ± 16.6	46.57 ± 0.0	35.14 ± 4.77	39.14 ± 9.85
average	61.48 ± 0.0	46.57 ± 0.0	38.51 ± 0.0	44.4 ± 0.0
Baseline with MixStyle	$\mathbf{97.97 \pm 0.73}$	$\mathbf{64.91 \pm 0.36}$	$\mathbf{72.37 \pm 1.18}$	76.02 ± 0.49

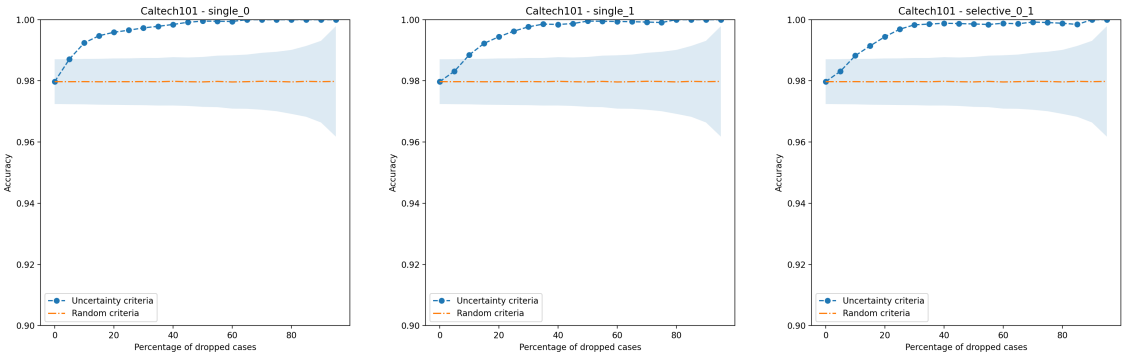
Table 4: Summary of results per domain and top-3 modes on VLCS. Accuracy, AUAD, and Gain are reported in % across seeds, AUC is reported in [0,1].

Domain	**Mode**	**Accuracy**	**AUAD**	**Gain**	**AUC**
Caltech101	single_0	97.60 ± 0.44	94.74 ± 0.18	1.66 ± 0.52	0.912 ± 0.032
	single_1	96.51 ± 1.04	94.67 ± 0.22	1.59 ± 0.47	0.880 ± 0.029
	sel_0_1	95.78 ± 1.27	94.64 ± 0.25	1.57 ± 0.44	0.879 ± 0.034
LabelMe	single_0	63.28 ± 0.68	57.88 ± 0.56	−3.79± 0.69	0.481 ± 0.009
	single_1	63.79 ± 0.88	58.28 ± 0.74	−3.40± 1.04	0.497 ± 0.012
	sel_0_1	63.22 ± 0.98	57.90 ± 0.90	−3.77± 1.20	0.487 ± 0.012
SUN09	single_0	70.28 ± 1.23	75.80 ± 1.11	7.05 ± 0.23	0.616 ± 0.005
	single_1	70.13 ± 0.86	75.80 ± 0.69	7.05 ± 0.46	0.620 ± 0.004
	sel_0_1	69.60 ± 0.77	75.53 ± 0.58	6.78 ± 0.54	0.612 ± 0.005
VOC2007	single_0	77.02 ± 0.69	79.84 ± 1.19	7.60 ± 0.91	0.663 ± 0.023
	single_1	77.08 ± 0.86	79.27 ± 1.40	7.03 ± 1.01	0.646 ± 0.027
	sel_0_1	76.45 ± 0.87	78.96 ± 1.23	6.72 ± 0.89	0.641 ± 0.025

sel_0_1 = selective_0_1

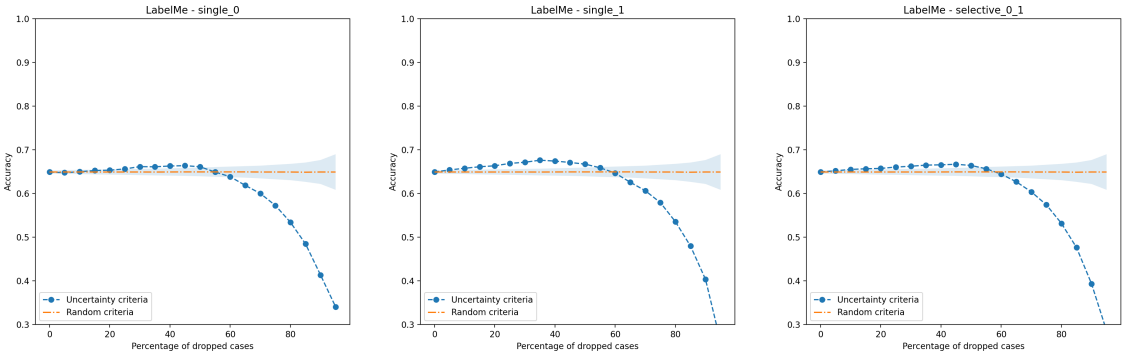
can degrade performance. The AUC results mirror this, with values near 0.5, which is equivalent to random discrimination. *SUN09* and *VOC2007* show intermediate behaviour. In *SUN09*, AUAD values of about 76% and gains of roughly 7% indicate that variance is moderately useful for filtering and is consistent with the clear separation visible between the uncertainty and random curves in Figure 12. The AUC values, around 0.61, suggest modest discriminative ability. A similar pattern

is observed in domain *VOC2007* (Fig. 13), where AUAD values close to 79% and gains of 7% align with AUC scores ≈ 0.65. Overall, the results highlight a marked variability across domains.



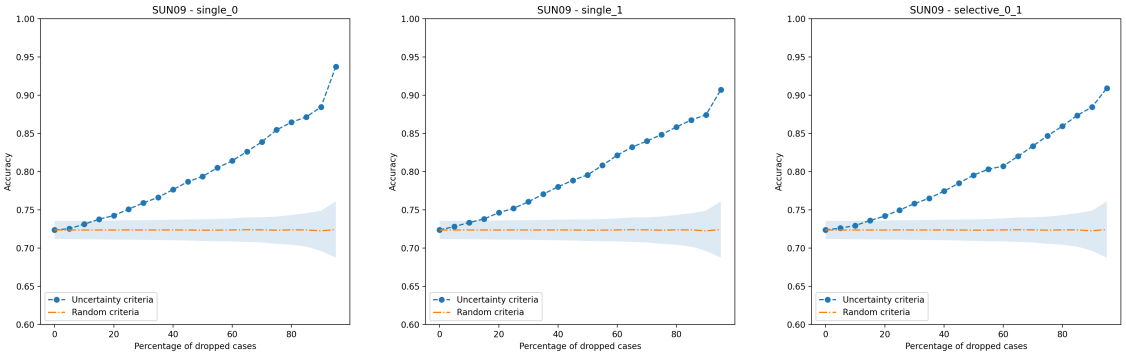
(a) *single_0* (b) *single_1* (c) *selective_0_1*

Figure 10: Dataset VLCS, Test Domain *Caltech101*



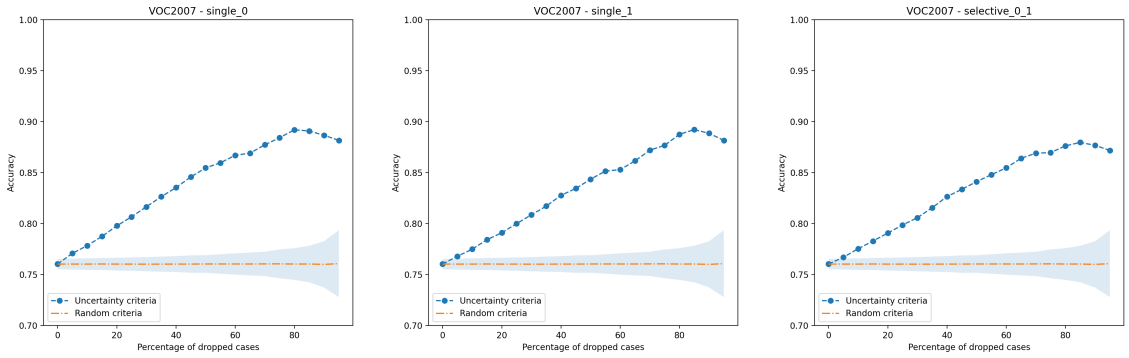
(a) *single_0* (b) *single_1* (c) *selective_0_1*

Figure 11: Dataset VLCS, Test Domain *LabelMe*



(a) *single_0* (b) *single_1* (c) *selective_0_1*

Figure 12: Dataset VLCS, Test Domain *SUN09*



(a) *single_0* (b) *single_1* (c) *selective_0_1*

Figure 13: Dataset VLCS, Test Domain *VOC2007*

4.4.2 Statistical Analysis

The statistical analysis reveals differences between the accuracy for TTA modes and the MixStyle baseline, with notable variation across datasets and domains.
In the regression tables, the intercept represents the estimated baseline accuracy for the reference condition, which is either the mode *average* or the MixStyle baseline, depending on the model. Each coefficient indicates the difference in performance between a specific TTA mode and this reference, with positive values denoting higher accuracy and negative values indicating lower accuracy. The associated p values report whether these differences are statistically significant, while the confidence intervals provide the estimated range within which the true effect is likely to fail. Thus, modes with positive coefficients and statistically significant p values can be interpreted as outperforming the reference, whereas non-significant or negative coefficients suggest comparable or inferior performance.

PACS The mixed linear model regression for PACS (see Tab. 5) shows clear differences in the effectiveness of TTA modes across domains. Several configurations exhibit strongly positive and statistically significant coefficients relative to the *average* mode reference. The most effective mode is *single_1* ($\beta = 0.723$, $p < 0.001$, CI $[0.686, 0.759]$), closely followed by *selective_0_1* ($\beta = 0.715$, $p < 0.001$, CI $[0.678, 0.752]$) and *single_0* ($\beta = 0.699$, $p < 0.001$, CI $[0.662, 0.736]$). *Single_2* and *selective_0_2* also significant positive coefficients ($\beta = 0.69$ and 0.692, respectively, both $p < 0.001$), albeit slightly lower. By contrast, *single_3* and its associated selective modes (*selective_{0,1,2}_3*) return coefficients close to zero (≈ 0.007) with non-significant p values ($p = 0.702$), statistically confirming their poor predictive performance in addition to the observations from the TTA results. These findings indicate that while several TTA modes provide a significant accuracy advantage in relation to the augmentation with averaged feature statistics, their effectiveness is highly mode-dependent.

When directly comparing the three best TTA modes against the MixStyle baseline (see Tab. 6), results become more nuanced. The baseline intercept is estimated

Table 5: Mixed Linear Model Regression Results for PACS (TTA-only: mode effects)

	Coef.	Std.Err.	z	P > |z|	[0.025	0.975]
Intercept [T.average]	0.134	0.055	2.450	0.014	0.027	0.241
C(mode)[T.selective_0_1]	0.715	0.019	38.083	0.000	0.678	0.752
C(mode)[T.selective_0_2]	0.694	0.019	36.972	0.000	0.657	0.731
C(mode)[T.selective_0_3]	0.007	0.019	0.382	0.702	-0.030	0.044
C(mode)[T.selective_1_2]	0.692	0.019	36.867	0.000	0.655	0.729
C(mode)[T.selective_1_3]	0.007	0.019	0.382	0.702	-0.030	0.044
C(mode)[T.selective_2_3]	0.007	0.019	0.382	0.702	-0.030	0.044
C(mode)[T.single_0]	0.699	0.019	37.234	0.000	0.662	0.736
C(mode)[T.single_1]	0.723	0.019	38.508	0.000	0.686	0.759
C(mode)[T.single_2]	0.690	0.019	36.758	0.000	0.653	0.727
C(mode)[T.single_3]	0.007	0.019	0.382	0.702	-0.030	0.044
seed Var	0.034					

Model: MixedLM Dependent Variable: accuracy
No. Observations: 132 Method: ML
No. Groups: 4 Scale: 0.0021
Min. group size: 33 Max. group size: 33 Mean group size: 33.0
Log-Likelihood: 188.1686 Converged: Yes

at 0.835 (CI [0.786,0.883]), which serves as the reference performance level. Here, only mode *single_1* achieves a statistically significant improvement ($\beta = 0.022$, $p = 0.002$, CI $[0.008, 0.036]$), confirming its superiority over the baseline across seeds. The other modes *single_0* ($\beta = -0.002$, $p = 0.781$, CI $[-0.016, 0.012]$) and *selective_0_1* ($\beta = 0.014$, $p = 0.047$, CI $[0.000, 0.028]$) are statistically indistinguishable from the baseline. The fixed effects estimates (see Tab. 19 and 20 in the appendix) confirm these patterns by providing narrow confidence intervals for the strong modes, which indicate that their effect sizes are estimated with high precision. This reinforces the conclusion that the observed improvements are both statistically reliable and robust across seeds.

Domain-specific improvements over the baseline (see Tab. 21 in the appendix) provide further insights. In *Art Painting*, both *single_0* and *single_1* show small positive mean differences (mean $\Delta = 0.0145$ and $\Delta = 0.0209$, respectively), though their confidence intervals cross zero, indicating non-significance. For domain *Cartoon*, mode *single_1* again records a positive mean improvement (mean $\Delta = 0.0187$), with *selective_0_2* and *selective_1_2* showing similar trends, though none are statistically conclusive. In the *Photo* domain, nearly all TTA modes underperform relative to the baseline, with *selective_1_2* producing a significant negative effect (mean $\Delta = -0.0202$, CI $[-0.0362, -0.0042]$). The most notable improvements occur in the *Sketch* domain, where *single_1* shows a highly robust positive effect (mean $\Delta = 0.0495$, CI [0.0428,0.0562]), representing the clearest and most consistent domain-specific gain across the dataset with the suggested approach.

Table 6: Mixed Linear Model Regression Results for PACS (MixStyle (base) vs. best TTA modes)

	Coef.	Std.Err.	z	P > |z|	[0.025	0.975]
Intercept (base)	0.835	0.025	33.678	0.000	0.786	0.883
single_0	-0.002	0.007	-0.278	0.781	-0.016	0.012
single_1	0.022	0.007	3.119	0.002	0.008	0.036
selective_0_1	0.014	0.007	1.986	0.047	0.000	0.028
seed Var	0.007	0.196				

Model: MixedLM Dependent Variable: accuracy
No. Observations: 48 Method: ML
No. Groups: 4 Scale: 0.0003
Min. group size: 12 Max. group size: 12 Mean group size: 12.0
Log-Likelihood: 99.4076 Converged: Yes

single_k = C(mode, Treatment(reference='base'))[T.single_k]
selective_i_j = C(mode, Treatment(reference='base'))[T.selective_i_j]

Thus, the statistical analysis for PACS indicates that while several TTA modes outperform chance levels, *single_1* is the only configuration that provides a consistent, statistically significant improvement over the MixStyle baseline, with its advantage being most pronounced in the *Sketch* domain.

VLCS The regression results for the VLCS dataset present a markedly different picture. In the TTA-only analysis (see Tab. 7), several modes demonstrate significant positive coefficients relative to the *average* intercept (0.477, CI [0.417,0.538]). Specifically, *single_0* ($\beta = 0.293,\ p < 0.001$), *single_1* ($\beta = 0.291,\ p < 0.001$) and *single_2* ($\beta = 0.233,\ p < 0.001$) all show strong positive effects. Selective modes such as *selective_0_1* ($\beta = 0.284,\ p < 0.001$) and *selective_1_2* ($\beta = 0.204,\ p < 0.001$) also achieve positive effects in comparison to the intercept. However, modes involving *single_3* (*selective_{0,1,2}_3*) yield small negative coefficients (-0.036) that are only marginally significant ($p = 0.057$). When the best three modes are directly compared against the MixStyle baseline (see Tab. 8), the positive impression shifts. The baseline intercept is estimated at 0.777 (CI [0.708,0.845]), higher than in PACS, and several TTA modes fall short of this level. The best modes *single_0* ($\beta = -0.006,\ p = 0.031,\ \text{CI}\ [-0.012, 0.001]$), *single_1* ($\beta = -0.008,\ p = 0.006,\ \text{CI}\ [-0.013, 0.008]$) and *selective_0_1* ($\beta = -0.014,\ p = 0.000,\ \text{CI}\ [-0.020, 0.008]$) are not significantly different from the baseline, which indicates comparable but not superior performance. The fixed effects estimates (see Tab. 22 and 23 in the appendix) reinforce these findings, with consistently negative coefficients and narrow confidence intervals for mode *single_2* and the selective combinations.

The domain-by-domain analysis (see Tab. 24 in the appendix) highlights further variability. In *Caltech101*, nearly all modes underperform relative to the MixStyle baseline, with *single_2* (mean $\Delta = -0.0757$, CI $[-0.1496, -0.0018]$) and *selective_1_2*

Table 7: Mixed Linear Model Regression Results for VLCS (TTA-only: mode effects)

	Coef.	Std.Err.	z	P > |z|	[0.025	0.975]
Intercept [average]	0.477	0.031	15.456	0.000	0.417	0.538
sel_0_1	0.284	0.020	14.520	0.000	0.245	0.322
sel_0_2	0.213	0.019	11.167	0.000	0.176	0.250
sel_0_3	-0.036	0.019	-1.902	0.057	-0.074	0.001
sel_1_2	0.204	0.019	10.703	0.000	0.167	0.241
sel_1_3	-0.036	0.019	-1.902	0.057	-0.074	0.001
sel_2_3	-0.036	0.019	-1.902	0.057	-0.074	0.001
single_0	0.293	0.019	15.369	0.000	0.256	0.330
single_1	0.291	0.019	15.281	0.000	0.254	0.329
single_2	0.233	0.019	12.217	0.000	0.196	0.270
single_3	-0.036	0.019	-1.902	0.057	-0.074	0.001
seed Var	0.009	0.087				

Model: MixedLM Dependent Variable: accuracy
No. Observations: 131 Method: ML
No. Groups: 4 Scale: 0.0022
Min. group size: 32 Max. group size: 33 Mean group size: 32.8
Log-Likelihood: 192.3458 Converged: Yes

single_k = C(mode)[T.single_k] sel_i_j = C(mode)[T.selective_i_j]

Table 8: Mixed Linear Model Regression Results for VLCS (MixStyle (base) vs. TTA modes)

	Coef.	Std.Err.	z	P> |z|	[0.025	0.975]
Intercept [base]	0.777	0.036	21.689	0.000	0.706	0.847
single_0	-0.006	0.003	-2.16	0.031	-0.012	0.001
single_1	-0.008	0.003	-2.752	0.006	-0.013	0.008
sel_0_1	-0.014	0.003	-4.9	0.0	-0.02	0.008
seed Var	0.015	1.035				

Model: MixedLM Dependent Variable: accuracy
No. Observations: 48 Method: ML
No. Groups: 4 Scale: 0.0000
Min. group size: 12 Max. group size: 12 Mean group size: 12.0
Log-Likelihood: 127.3225 Converged: Yes

single_k = C(mode, Treatment(reference='base'))[T.single_k]
sel_i_j = C(mode, Treatment(reference='base'))[T.selective_i_j]

(mean $\Delta = -0.1149$, CI $[-0.2085, -0.0212]$) showing significant declines. In *LabelMe*, even the stronger modes exhibit negative mean differences, such as *single_1* (mean $\Delta = -0.0108$, CI $[-0.0145, -0.0072]$), indicating that the baseline remains superior. *SUN09* displays the sharpest contrast, with large and significant declines

for mode *single_2* (mean $\Delta = -0.1091$, CI $[-0.1893, -0.029]$) and the selective combinations (up to -0.1411). Only in domain *VOC2007* do the modes *single_0* and *single_1* show small positive differences (0.0108 and 0.0113), though their confidence intervals include zero, which renders them statistically insignificant.

Overall, the VLCS analysis reveals that the MixStyle baseline consistently outperforms most TTA configurations of this approach, with only two modes approaching comparable levels of performance in isolated domains.

5 Discussion

5.1 Experimental Findings

The results of this study hint at the strong dependence of TTA's effectiveness on both the dataset and the chosen mode. While PACS demonstrates consistent improvements under certain TTA configurations, VLCS largely favors the MixStyle baseline. We first analyze PACS, then VLCS, and finally synthesize results across datasets and relate them to prior work.

PACS The PACS experiments show in which cases domain-aware TTA can help and when it harms. Injecting augmentations at the shallow or intermediary layers is beneficial, while deeper interventions are detrimental.

This benchmark offers a revealing test case for domain-aware TTA. The descriptive results show that the effectiveness of the method depends strongly on which layers are targeted for adaptation. Configurations that intervene in shallow and mid-level layers, especially *single_1*, consistently achieve the best results. This mode improves over the MixStyle baseline in three out of four target domains, with mean accuracies of 87.47% in *Art Painting*, 79.57% in *Cartoon*, and 78.8% in *Sketch*, compared to 85.75%, 77.7% and 73.84% for the baseline. In the *Photo* domain, where the baseline already reaches 96.99%, *single_1* remains almost identical at 96.86%. The *single_0* configuration also performs well, slightly outperforming the baseline in Photo (97.11%) while remaining competitive in the other domains. In contrast, adapting at the deepest block, as in *single_3*, or stacking it with other layers like in the selective variants, produces collapsing accuracies to the low teens ($\approx 12 - 16\%$) across all domains. The selective modes reflect the same depth-dependent trend. *Selective_0_1*, which combines two shallow blocks, closely mirrors the strong performance of *single_1*, while combinations that include the deepest block replicate the breakdown of *single_3*. Finally, the *average* mode, which injects pooled statistics across all domains, performs worst of all. Across the dataset, it reduces accuracies to $9-16\%$, well below both baseline and any targeted configuration.

Uncertainty analysis provides a complementary perspective. Using variance across style augmentations as an uncertainty measure reveals a consistent ability to separate easy from difficult cases. Accuracy-drop curves demonstrate high AUAD values in *Art Painting, Cartoon* and *Photo* (around $90-95\%$) and somewhat lower but still useful values in *Sketch* ($\approx 83\%$). Gains over random removal quantify this advantage: improvements of $9-13\%$ in *Art Painting, Cartoon* and *Sketch* show that discarding high-variance samples leads to substantial boots in retained accuracy. While *Photo* achieves only $\approx 2\%$ gain, this reflects the fact that its baseline is already near ceiling. AUC values reinforce this interpretation. In *Art Painting* and *Cartoon*, variance is a strong predictor of error (≈ 0.82 and 0.78). In *Sketch*, the values are lower ($\approx 0.64 - 0.74$), indicating moderate but not perfect discrimination. In *Photo*, variance is especially reliable, with AUC ≈ 0.93, which shows

that even small fluctuations in confidence correlate with misclassifications. Importantly, these uncertainty patterns align with the most effective adaptation modes. The same shallow/mid-level configurations that achieve the highest accuracies also yield the clearest separation between correct and incorrect predictions, which makes per-sample variance not only a diagnostic signal but also a useful tool for selective prediction for this dataset.

Statistical analyses confirm and sharpen these observations. When using *average* as the regression reference, all shallow and mid-layer modes achieve large, positive and highly significant coefficients (e.g., $\beta \approx 0.723$ for *single_1*, $p < 0.001$), while the deep-layer modes remain statistically indistinguishable from zero, which formally establishes the depth effect seen in the descriptive results. When compared directly to the MixStyle baseline, results are more conservative. *Single_1* emerges as the only configuration with a statistically robust improvement ($\beta \approx 0.022, p \approx 0.002$). *Single_0* and *selective_0_1* are statistically comparable to the baseline, showing that while they may match its performance, they do not significantly exceed it. Domain-specific contrasts localize these effects even further. *Sketch* shows the clearest and most reliable benefit, with *single_1* achieving a mean improvement of $\approx +0.05$ over the baseline and narrow confidence intervals that exclude zero. In *Cartoon* and *Art Painting*, the gains are smaller and not always statistically conclusive, while in *Photo* most TTA modes remain neutral or slightly negative, consistent with its already high baseline accuracy. The domain-level contrasts (see Tab. 21 in the appendix) further clarify where these effects originate. In *Sketch*, *single_1* produces a highly reliable mean improvement of roughly +0.05 over the baseline, with confidence intervals that are both narrow and strictly positive. This underscores that gains in this domain are consistent across seeds and represents the clearest case where shallow-layer adaptation provides tangible benefits. In *Cartoon*, positive differences are also observed, especially for *single_1* (mean $\Delta \approx 0.019$) and certain selective variants, though here the confidence intervals overlap zero, which renders the improvements suggestive but not conclusive. For *Art Painting*, the effects are even smaller. Shallow adaptations trend slightly positive, but none reach statistical significance, which aligns with the descriptive finding of only moderate performance shifts. By contrast, *Photo* shows no meaningful improvements; most modes remain neutral or slightly negative relative to the baseline, consistent with the domain's already high baseline accuracy that leaves little headroom for further gains. Taken together, the regression results confirm that while the advantage of shallow test-time adaptation is robust at the global level, it is concentrated almost entirely in the *Sketch* domain, with *Cartoon* and *Art Painting* showing weaker and less stable improvements, and *Photo* remaining essentially unaffected. This domain-specific breakdown highlights that the utility of domain-aware adaptation is not evenly distributed, but most pronounced where the stylistic mismatch with training data is largest.

VLCS Below, we discuss the VLCS findings. Overall, the descriptive picture is clear: our approach does not surpass the MixStyle baseline. The baseline already

achieves strong accuracies, such as 97.97% on *Caltech101* and 76.02% on *VOC2007*, with moderate values like 64.91% on *LabelMe* and 72.37% on *SUN09*. Shallow or mid-layer TTA modes sometimes come close, but rarely exceed these levels. For example, *single_0* and *single_1* yield 97.6% and 96.51% on *Caltech101*, which is essentially at parity but not exceeding the baseline. In *LabelMe*, the best modes reach 63.79% versus 64.91% for the baseline; and in SUN09, they hover around 70%, still below 72.37%. The one domain where our domain-aware TTA approach edges ahead is *VOC2007*, where mode *single_1* obtains 77.08% and *single_0* 77.02%, both nominally above the baseline of 76.02%. However, this edge is small and, as we show below, fragile when subjected to statistical controls. The *selective* modes largely mirror the shallow single-layer variants. *Selective_0_1* reaches competitive but not superior accuracies (e.g. 95.8% on *Caltech101* vs. 97.9% baseline; 76.5% on *VOC2007* vs. 76.0%), while combinations involving the deepest block collapse to poor performance ($\sim 35 - 46\%$). The *average* mode performs the worst across domains. While single and some selective modes hover close to the baseline, averaging style statistics collapses accuracy to mid-40s or lower (e.g., 61.5% on *Caltech101*, 46.6% on *LabelMe*, 38.5% on *SUN09*, 44.4% on *VOC2007*). This shows again that blending domain styles washes out discriminative structure instead of bridging shifts. These depth-dependent structures follow the findings on the PACS dataset.

Uncertainty and selective prediction provide a complementary lens to these results. On *Caltech101*, the AUC between variance and error lies high ($\approx 0.88 - 0.91$), and AUAD is $\approx 94.7\%$ with small positive gains of about 1.6%. This pattern of strong discrimination with only marginal selective-prediction improvement is expected near a performance ceiling, such that a good uncertainty signal has limited headroom to improve retained accuracy after abstention. The results for domain *LabelMe* sit on the opposite extreme. AUAD is low ($\sim 58\%$) and the gains are slightly negative (-3% to -4%). The AUC hovers around $\sim 0.48 - 0.5$ which equals random chance. Here, the per-sample variance fails to separate hard from easy examples better than random. The fluctuations induced by our style augmentations do not align with the true sources of error in this domain. *SUN09* and *VOC2007* occupy the middle ground. AUADs are $\sim 75-80\%$ with gains around 7%, with AUC results moderately above chance ($\sim 0.61 - 0.66$). In these two domains, variance is useful for triaging predictions, yet it still falls short of being as reliable as on PACS. The confidence instability we measure on VLCS is thus only partially linked to misclassification tasks.

Statistical modeling confirms and sharpens these patterns. In the TTA-only analysis that uses mode *average* as the reference, shallow and mid-layer modes show positive fixed effects. Coefficients around +0.29 for *single_0* and *single_1* as well as +0.23 for *single_2* with *selective_0_1* being similarly positive (+0.28). In contrast, any mode that touches the deepest block like *single_3* and its selective variations trends negative (≈ -0.036) and is at best marginal. This ranking is consistent with the findings for PACS.

When compared to the MixStyle baseline, the coefficients of the best TTA modes become small and negative: -0.006 for *single_0* ($p \approx 0.03$), -0.008 for *single_1*

($p \approx 0.006$) and -0.014 for *selective_0_1* ($p < 0.001$). Thus, once domain variability is controlled, the apparent descriptive ties or tiny wins largely diminish; across VLCS, the baseline is at least as good or better. Domain-level contrasts reinforce this point. In *Caltech101*, several TTA modes suffer significant drops (e.g., *single_2* at -0.0757; *selective_1_2* at -0.1149). In *LabelMe*, even nominally strong modes show small but consistent negatives (e.g., *single_1* at -0.0108). *SUN09* reveals the sharpest declines for deeper or *selective* modes (*single_2* at -0.1091; *selective* combinations up to -0.1411). Only in *VOC2007* do *single_0* and *single_1* show positive deltas ($\sim +0.011$), and even there the confidence intervals include zero, which indicates no statistical conclusive gains.

Interpretation and Implications Addressing our first research objective, our findings demonstrate that the choice of feature extraction strategy has a decisive impact on DG performance within a TTA pipeline. In line with our hypothesis that earlier residual blocks capture style information more effectively than deeper blocks, adaptations applied at shallow and mid-level layers like *single_0* and *single_1* improved or at least matched baseline accuracy, whereas interventions at the deepest block (*single_3*) consistently degraded performance. This pattern was most pronounced on PACS. Where the stylistic shifts dominated, the early-layer strategies delivered stable gains, particularly in the *Sketch* domain. On VLCS, shallow strategies achieved parity at best with the MixStyle baseline, again with later-layer interventions proving harmful. These outcomes confirm that the extraction layer is a critical determinant of TTA effectiveness, with shallow layers offering the strongest benefits under stylistic shifts, and they further support our hypothesis that domains which are strong during training tend to remain strong under TTA, while weaker domains continue to lag behind, e.g. with both the *Photo* and *Caltech101* domains being both accurate in training and during inference.

The depth-dependent behaviors observed in PACS and VLCS can be explained through the hierarchical organization of CNNs such as ResNet-50. Earlier layers predominantly capture low-level statistics such as edges, textures and color distributions, while progressively deeper layers encode higher-level semantic content like object structures (Yosinski et al., 2014; Zeiler and Fergus, 2014). Prior work in texture synthesis and style transfer has shown that style information is concentrated in shallow to mid layers of CNNs, where correlations of feature maps effectively reproduce textures and stylistic attributes (Gatys et al., 2015). More recent analyses confirmed that ImageNet-trained CNNs rely heavily on local texture clues for classification, which underscores the role of shallow/mid layers as the primary carriers of style features (Geirhos et al., 2018).

Within this framework, the results of both datasets align with these expectations: the strong performance of modes *single_0* and *single_1* can be attributed to their intervention at precisely those stages where style is most strongly represented. Injecting domain statistics here realigns superficial attributes such as texture or brushstroke patterns, which constitute the dominant axes of variation across PACS domains. Because these layers are early enough that semantic evidence is not yet

tightly bound to the activations, manipulating them does not destabilize category-level information. By contrast, deeper interventions such as *single_3* corrupt representations that already encode semantic distinctions by overwriting class-specific abstractions with mismatched style priors and thereby collapsing accuracy. *Selective* shallow combinations such as *selective_0_1* remain effective because they reinforce alignment at style-sensitive layers without disturbing semantics, though their benefits diminish once statistics have been shifted toward a plausible source style. The failure of the *average* mode is likewise consistent: pooling statistics across heterogeneous domains produces incoherent distributions that match no real style, which leads to representational collapse (Chang et al., 2019).

Addressing our second research objective, our results again show that the most effective methods are those that intervene at shallow or mid-levels without disturbing deeper semantic features. Specifically, *single_0*, *single_1* and *selective_0_1* consistently preserved class-discriminative information while improving or at least matching the baseline accuracy. Any mode involving the deeper layers led to severe performance breakdowns, confirming our hypothesis that interventions in later layers overwrite semantic representations and erase class information. On PACS, the shallow and mid-level strategies yielded clear improvements, especially in domains with strong stylistic mismatch, while mode *average* collapsed accuracy by blending incompatible style cues. On VLCS, the shame shallow strategies achieved parity but rarely surpassed the MixStyle baseline. These outcomes also support our hypothesis that augmenting 'weaker' domains with statistics from 'stronger' ones can provide measurable gains, as observed in PACS (*Sketch* improved most when aligned with statistics from *Cartoon* or *Photo*), though this effect was less pronounced in VLCS.

The contrasting outcomes on PACS and VLCS regarding the (mis-)success of our approach can be understood through the representational hierarchy of CNNs and the nature of the domain shifts in each benchmark. In PACS, the domain shifts are primarily stylistic. Images differ in textures, edges and color distributions across the four modalities that are being handled as domains. These variations align closely with what early and mid layers of a ResNet-50 represent (Yosinski et al., 2014; Zeiler and Fergus, 2014). VLCS represents a more content-driven domain shift. Due to its nature, the domains differ by object scale, background clutter, contextual co-occurrence and even annotation conventions. Such variation challenges deeper semantic representations rather than shallow style encodings. Consequently, the feature statstics injection of our approach appears to be poorly aligned with the sources of error for this dataset.

Beyond which layer to adopt, the direction of augmentation matters. For a fixed test domain, there seems to be a difference in which source domains statistics provide the best alignment for performance. On PACS, the answer is not always the 'nearest' domain and it seems to vary by target domain (see Tab. 11−14 in the appendix). For *Art Painting*, shifting towards Cartoon is best under the top mode *single_1*, where the mean accuracy reaches 88.14%, which beats augmenting toward both *Photo* (86.87%) and *Sketch* (87.4%). For *Cartoon*, the strongest configuration couples a shallow or mid intervention with a complementary source, where augmenting

with statistics from the *Sketch* domain yields the highest accuracy (81.64%). For the *Photo* domain, whose accuracy is already close to the ceiling, the best performing source domain is *Art Painting* (97.55% in *single_0*), which appears to be logical, given their stylistic similarities. However, shifting *Art Painting* towards *Photo* doesn't work as well, as seen above. Augmenting images of the *Photo* domain with statistics from worse performing domains would intuitively not work for a better performance, which seems to be confirmed here. For *Sketch*, the top mode *single_1* dominates regardless of source, but the top direction is the *Cartoon* domain (79.18%), closely followed by *Photo* (79.02%). Collectively, these patterns indicate that augmenting toward the nearest domain is not consistently optimal. Instead, the best source often provides a sufficiently large and stylistically informative shift. This supports the view that what matters is the magnitude and relevance of the style realignment rather than proximity per se.

On VLCS, directionality is clearer but yields smaller gains and seldom surpasses the MixStyle baseline even when split up for the target domains (see Tab. 15−18 in the appendix). For *Caltech101*, the strongest direction is *LabelMe* under mode *single_0* (97.74%) which still trails the baseline. Augmenting *LabelMe* test images towards *VOC2007* appears to be the best choice with mode *single_1* (64.72%). For *SUN09*, the best configuration is augmenting toward *VOC2007* in *single_0* (70.7%). The one case with a nominal edge over baseline is *VOC2007*, where augmenting towards the strongest domain *Caltech101* on *single_1* reaches 77.7% accuracy. As in PACS, the best directions appear to be often complementary rather than intuitively 'nearest'. But in this case, the content-centric nature of the dataset limits how much a style-only realignment can help. Taken together, the directional analysis supports our broader claim: effective augmentation directions are those that induce a sufficiently informative shift in shallow or mid-level style statistics. However, when the underlying domain shift is semantic, as in VLCS, even the best performing domain rarely closes the gap to the baseline.

A complementary view comes when visualizing the raw datasets with t-SNE (see Fig. 4 and 5). In PACS, the domains are mostly well separated, with three distinct clusters forming. *Sketch* shows the most compact group, while *Art Painting* and *Photo* are overlapping. This clear separation reflects the strong stylistic differences between the domains and explains why our approach proves as an effective layer for style alignment. By contrast, VLCS shows extensive overlap across the domains, with only *Caltech101* forming a few sub-clusters. This overlap highlights that the main source of variation is not stylistic texture but semantic and contextual content, which our approach cannot easily bridge. This visualization reinforces the interpretation of style-driven TTA needing a clear stylistic domain separation.

A further factor that helps explain the contrast between PACS and VLCS is the intrinsic structure of the datasets themselves. Within VLCS, performance varies strongly across domains, with *Caltech101* consistently yielding higher accuracies than *LabelMe, SUN09* or *VOC2007*. A plausible explanation lies in the photographic composition of the images. *Caltech101* typically presents objects in a centered, uncluttered manner, which reduces background interference and aligns well with the

spatial biases of CNN feature extractors. By contrast, the other domains often contain off-center objects, richer contextual co-occurrence, and distracting background material, introducing semantic variability that shallow style-based adaptation cannot resolve. This points to a more general limitation: Augmentations based on feature statistics, while effective at bridging stylistic shifts in PACS, are insufficient to capture the full extent of natural variation in VLCS, where errors stem more from structural, contextual, or annotation differences than from style alone. In this sense, the dataset characteristics themselves delimit the conditions under which TTA provides meaningful benefits.

Interpreting the uncertainty results reveals complementary but dataset-specific behaviors. By design, the method probes each test input under several plausible source styles. Samples whose predictions are stable across stylizations are likely to be both stylistically well-aligned and semantically unambiguous, while those that fluctuate tend to depend on style-sensitive features or lie near decision boundaries. Thus, per-sample variance acts as a functional estimate of sensitivity to stylistic perturbations, a form of epistemic uncertainty tied to the model's texture bias.

In PACS, this metric shows the strongest results in *Photo* (high AUC, small AUAD gains), a regime with few but well-identified hard cases, whereas in *Sketch*, values are noisier but still useful for ranking. Abstention thresholds should therefore be tuned per domain: conservative in *Sketch* to capture brittle cases, and tighter in *Photo* to catch rare but clear outliers. Importantly, these benefits are realized on top of MixStyle training (Zhou et al., 2021), which already improves style robustness. Our test-time method improves further, especially in shallow/mid-layer modes (*single_1*, *selective_0_1*), showing that it is complementary rather than redundant. MixStyle broadens the basin of attraction through average-case augmentation, while test-time alignment snaps inputs toward discrete, mastered style modes.

The findings on VLCS present a more nuanced picture. Because all four domains (*Caltech101, LabelMe, SUN09, VOC2007*) are photographic and differ more in scene layout, context, and annotation than in texture, variance reflects scene and label-context shifts rather than style. It remains highly predictive in *Caltech101* (high AUC, small gains), moderately useful in *SUN09* and *VOC2007* (mid AUC, mid gains), and non-informative in *LabelMe* (AUC ≈ 0.5, negative gains). Relative to PACS, complementarity with MixStyle is weaker: MixStyle's average-case robustness already covers most residual variability in VLCS, so snapping test inputs toward stored source styles rarely helps and can even hurt when the dominant shift is semantic rather than textural.

Addressing our third research objective, we find a clear relationship between predictive accuracy and our uncertainty measure, per-sample variance, across extraction strategies. Modes that improved accuracy, such as *single_0*, *single_1* and *selective_0_1*, also produced lower variance and stronger discriminative power for error detection, whereas modes that harmed accuracy (notably those involving deep-layer adaptation) generated unstable and less informative uncertainty estimates. On PACS, this coupling was especially strong with the best modes yielding steep accuracy-drop curves and high AUC values, showing that higher accuracy coincided

with more reliable uncertainty signals. On VLCS, the correlation was weaker but still present with shallow strategies showing moderate to high AUCs. The outlier in *LabelMe* could be ascribed to domain-specific properties.

While these dataset-specific patterns highlight when variance succeeds or fails as a predictor, it is also important to situate our approach in the broader landscape of uncertainty estimation. The metric employed in this approach, per-sample variance across style augmentations, differs conceptually from more common measures such as entropy or confidence scores. Because it is structurally tied to the TTA mechanism itself, it provides a particularly direct view on how predictions respond to plausible domain shifts. Our approach injects different source domain style statistics into each test sample's features, which yields a set of heterogeneous outputs that reflect how sensitive the model's classification is to stylistic shifts. If the model is truly confident and invariant to style, these augmented predictions will agree by showing low variance values, whereas high variance indicates the prediction flips under the domain shifts, which is a clear sign of uncertainty. The variance thus acts as a form of aleatoric uncertainty measurement by capturing the output variability due to input perturbations (Wang et al., 2019). One advantage of our method is that it is a data-centric ensemble at inference because we combine predictions from transformed versions of the same sample rather than from multiple models (Conde et al., 2024). This avoids the need to train an ensemble of networks while still reaping the benefits of ensemble uncertainty. Compared to model-centric approaches like Monte Carlo dropout, where dropout is kept active during inference and multiple forward passes are used to approximate the posterior predictive distribution (Gal and Ghahramani, 2016), our variance metric probes the data sensitivity of the model by checking consistency across input transformations. This enables our per-sample variance to complement traditional confidence measures by focusing on domain-related uncertainty. Whereas a single forward pass softmax score may be overconfident and mislead on OOD inputs, the variance across style augmented predictions provides an external consistency check to reveal when a high probability prediction may be fragile.

The per-sample variance proved to be a reliable indicator of prediction difficulty in our project. For instance, on the PACS benchmark we found that variance-based uncertainty yielded high area under ROC values for error detection (AUC ≈ 0.82 in the *Art Painting* domain, and up to ~ 0.93 in *Photo*), far above random chance (0.5). This suggests that the test examples whose predictions fluctuated the most across style augmentations where the ones likely to be misclassified. Exploiting this, we ranked test samples by their variance and removed the most uncertain fraction, which led to substantial accuracy boosts on the remaining data (e.g., a $9 - 13\%$ increase in accuracy in domains *Cartoon* and *Sketch* after filtering out the top 20% most uncertain samples), whereas removing the same number of samples at random yielded minimal improvement (see Fig. 6−13). These plots underline that our variance metric is capturing meaningful signals. If discarding a small percentage of high-variance predictions yields a much larger gain in reliability than a random removal, this indicates that the metric is pinpointing the difficult cases. Notably,

this aligns with observations in prior work that TTA-based uncertainty estimates can reduce overconfident errors and improve calibration compared to single-pass or dropout-based methods (Wang et al., 2019). Our use of style-focused TTA provides a targeted form of this effect by focusing on domain-shift-induced uncertainty. It differs from generic uncertainty metrics like softmax entropy or maximum probability by being sensitive to stylistic perturbations, which makes it especially effective on datasets such as PACS where inter-domain differences are largely texture and style based. Even on the more content-driven VLCS dataset, the variance offered moderate utility by achieving an average AUC around $0.6-0.7$ in domains like *SUN09* and *VOC2007*. Incorporating per-sample variance as an uncertainty measure was highly valuable for this project by providing deeper insight into our model's robustness.

Several alternative explanations may account for the outcomes reported above. Normalization interactions remain a potential confound: injecting foreign statistics into deeper blocks can mis-scale activations when BatchNorm moments mismatch per-sample distributions, especially in scene-heavy VLCS images, though this issue could be mitigated by recomputing normalization in strict evaluation mode or by adopting a Normalization strategy like GroupNorm or InstanceNorm. Another factor is source-statistic representativeness, since the style bank reflects variable intra-domain diversity. Statistics from 'safer' domains such as *Caltech101* or *Photo* may cause less harm, which aligns with their relatively strong AUC values, and an explicit analysis of domain balance and within-domain variability (as emphasized in style-balancing work from Park et al. (2023)) would help contextualize these effects. Architecture specificity also poses a limitation, as the optimal depth index identified for ResNet-50 (*single_1*) may not transfer cleanly to other backbones such as ConvNeXt or ViTs; while shallow and mid-level adaptation is likely to generalize, the exact layer choice may differ. In addition, result variability should not be overlooked: Despite averaging over seeds, some domains such as *Sketch* and *LabelMe* exhibit wide spreads in AUC and AUAD, particularly where variance approaches randomness. This means that although ranking across modes is consistent, statistical significance depends on the assumptions of the mixed-effects analysis.

The broader context of our findings highlights that TTA is not a one-size-fits-all remedy and its success is highly contingent on the nature of the domain shift and the model itself. Consistent with prior analyses, simple augmentations may fail to capture the full spectrum of natural variation in complex shifts (Shanmugam et al., 2020). For instance, our VLCS results underscore observations from Shanmugam et al. (2020) that when the distribution shift is more semantic or contextual rather than stylistic, conventional TTA strategies struggle to improve performance. Moreover, the impact of TTA appears to diminish with increasing model complexity. Kimura (2021) notes that simpler models often reap larger benefits from TTA compared to deeper, more capable networks, which could explain the gains from our results.

Another important consideration is when and how to apply the style-based adaptation. Our evidence and prior work advocate a selective, domain-aware use of this feature statistical TTA. If a target domain exhibits clear stylistic deviation as in

PACS, shallow-layer style alignment yields tangible gains; but if the shift is not primarily stylistic or is already well-handled by the base model (as in VLCS with the MixStyle baseline), aggressive test-time changes can be unnecessary or even detrimental. Recent advances like Park et al. (2023) offer an alternative perspective through their style balancing approach. By injecting a few style-transferred samples during training to cover underrepresented class-domain combinations, they address domain imbalance before deployment to ensure each source domain has sufficient examples per class. This pre-training approach incurs no test-time cost, which implies that if domain shifts are anticipated and broad enough, it is possible to handle them upfront and skip test-time processing altogether for faster inference. The direction of only engaging TTA when the domain shift exceeds a certain threshold, and otherwise relying on a well-trained, balanced model, represents a promising strategy for minimizing unnecessary inference-time computation while preserving accuracy on in-distribution data. The computational efficiency of such methods is inherently higher during deployment, since they avoid the per-sample overhead of online adaptation. In scenarios where test-time speed is crucial, this trade-off between pre-training augmentation and on-the-fly adaptation becomes especially relevant. In terms of computational efficiency, our approach and that of Park et al. (2023) make opposite trade-offs. By storing and injecting style statistics at inference, our method keeps training simple but introduces additional overhead during deployment, as each test sample requires extra processing for feature manipulation and uncertainty estimation. By contrast, the approach from Park et al. (2023) shifts the complexity into training: a subset of samples is style-shifted before learning to ensure a more balanced class disposition; then the distance to the nearest domain is calculated and only when this succeeds a certain threshold, is test-time evaluation executed as a standard forward pass. This means their method achieves lower latency and higher throughput at deployment, which is advantageous in real-time or resource-constrained scenarios. Our method offers the flexibility of adapting dynamically to unseen domains without requiring prior style balancing, at the cost of slower inference.

Our methodology itself yields insights about the interaction of training-time DG and TTA. In our case, the style statistics for adaptation are extracted from a model trained without using MixStyle (Zhou et al., 2021) to ensure that each domain's features remain distinct and suitable for targeted manipulation. This distinguishes our approach from the MixStyle baseline, where domain information is intentionally mixed during training. Whether it would be possible or even advisable to apply feature statistics TTA on top of a trained MixStyle model remains uncertain. Because MixStyle explicitly blends domain styles, the resulting representations may no longer preserve domain-specific feature distributions with the synthetic styles created during training. Injecting style statistics at test time in such a setting could reintroduce redundancy or even conflict with the already mixed representations. At the same time, it is conceivable that carefully designed variants might combine the broad robustness of MixStyle with the domain-targeted calibration of test-time style injection by careful analysis to avoid collapsing domain cues. This remains to be tested in practice.

Another practical consideration is our choice of using the global hyperparameter configuration across all folds and target domains for consistency. While this ensured fairness in comparison, it might not have been optimal for each domain. By allowing to train the models with their respective hyperparameter configurations determined during tuning, we might improve the quality of extracted statistics and thus the results themselves, albeit at the cost of additional complexity. This uniform approach could partially explain why some domains, especially those with unique characteristics like *LabelMe*, didn't see improvements. Nevertheless, the consistency of our hyperparameters makes our findings more directly comparable across domains, which reinforces the validity of the depth-related trends we observed.

Encouragingly, our findings resonate with and reinforce developments in the literature. The recent work of Yamashita and Hotta (2024) demonstrates that MixStyle-based contrastive TTA can surpass standard DG baselines to affirm that careful designed TTA can significantly boost robustness in unseen domains. Likewise, the layer-wise pattern we observed mirrors the conclusions of Wang et al. (2022). Their results reported the largest accuracy gain when injecting style randomization right after the first ResNet block, which parallels our findings of the best configurations (*single_0* and *single_1* on ResNet-50) for the approach presented in this thesis. Their approach of adding random noise on top of original feature statistics to diversify styles provides an interesting comparison: Whereas they added subtle noise to enrich style variation, our method takes a more extreme step by completely substituting feature statistics with those from other domains at inference. This full swap could be seen as pushing the style randomization to its limit, which in PACS yielded strong gains but in VLCS proved excessive, which again highlights the importance of matching the adaptation strategy to the nature of the shift. The success of other TTA techniques relative to our baseline with MixStyle reinforces the general efficacy of test-time adaptation for domain shift problems, and it motivates exploring hybrid methods.

In summary, these reflections emphasize that the benefit of TTA is nuanced. It works best when the source of domain shift aligns with what the augmentation can offer like texture differences, and when applied at the right representational level. Looking forward, incorporating insights such as those from Shanmugam et al. (2020), Kimura (2021) and Park et al. (2023) could guide more adaptive and efficient TTA protocols, which might dynamically decide if and how to apply augmentation at inference, possibly by first detecting the degree of domain shift or relying on uncertainty estimates (as we have explored) to trigger adaptation only when needed. This might preserve the strong baseline performance on easier or well-aligned domains and enable the deployment of style-shifting tools on harder, OOD cases to achieve a balance between robustness and efficiency.

5.2 Limitations

Our implementation introduces limitations beyond the dataset and design choices already discussed.

The pipeline is architecturally coupled to the ResNet hierarchy, where statistics and hooks are defined per block with fixed channel assumptions, so portability to non-ResNet backbones would require re-engineering. The hook-based test-time transformation itself is somewhat fragile in deployment by relying on correct internal naming and careful hook management. This increases maintenance risk and the chance of silent failures or memory leaks. Our aggregation methods for the style transfer are heuristic rather than learned; more principled approaches such as learned layer weighting could yield better calibration and robustness. The mode *average* is especially critical. In it, we reconcile layer heterogeneity by interpolating channel statistics to a common size, which distorts the underlying style signal as shown in section 4. Another limitation is the nontrivial overhead for storage and runtime. Because we persist per-mode, per-domain statistics in both JSON (for better readability) and tensor formats and perform repeated forward passes, scaling to larger dataset may reduce the computational efficiency even further.

Another limitation could be the exclusion of one domain from training, so that the amount of available training data per fold is reduced, which may increase variability. This effect is partly mitigated by repeating experiments with multiple random seeds and result aggregation. However, this reduction in training data is an inherent requirement of the LODO setup, which is essential for evaluating TTA and DG under truly unseen domains, and thus cannot be avoided without compromising the validity of the experimental design. This constraint can also be advantageous. Because our approach relies on extracting style statistics from a single domain, it is amenable to single-source DG scenarios, where the features from the single source could be used at test time to align samples more closely with the source domain distribution, albeit at the cost of losing the uncertainty measure that relies on diversity across domains.

The domains from our used datasets are imbalanced in difficulty, with some posing greater challenges than others, which may bias cross-domain comparisons. The external validity of our findings is restricted to the PACS and VLCS datasets. This raises the question of how well our approach generalizes to other benchmarks or real-world settings.

Additionally, in this work, style statistics are defined as the per-domain means and standard deviations of intermediate feature activations, and were maintained throughout training using an EMA update rule. This approach was directly motivated by the runnning statistics employed in Batch Normalization layers (Ioffe and Szegedy, 2015), where EMA provides an efficient method of approximating global dataset statistics. Instead of recomputing exact statistics from all training samples, EMA iteratively blends the current batch statistics with previously accumulated values, thereby yielding a smoothed and memory-efficient approximation of the underlying distribution.

The primary advantage of EMA is its ability to mitigate stochastic fluctuations arising from limited batch sizes or sampling variability (Ioffe, 2017; Cai et al., 2021). By placing greater weight on recent batches, the method adapts to changes in the learned feature space during training to ensure that the stored statistics remain aligned with the evolving representation. However, this temporal weighting also introduces a subtle bias towards later training phases, which may result in earlier domain characteristics being underrepresented. Whether this bias is beneficial or detrimental depends on the objective. It may improve stability by reflecting the most refined feature space, but it could also reduce fidelity to the full distribution of domain styles.

An alternative strategy is to compute the style statistics post hoc, after training has converged. In this setting, the final model would be used to extract statistics from the entire training set to ensure that each sample contributes equally and that the statistics correspond exactly to the converged feature representation. From a theoretical standpoint, this yields a more balanced and unbiased estimate of each domain's style. Such an approach may be particularly appealing in TTA scenarios, where accurate cross-domain style information is critical (Wang et al., 2022).

While our approach provides a smoothed approximation of the domain statistics, its necessity in our setup is debatable. Since we already perform a dedicated post-training extraction phase through an additional forward pass with a dummy input, EMA does not reduce the computational burden of style statistics collection but instead influences how these statistics evolve during training. This raises the question of whether EMA is truly beneficial in this context, or whether computing post hoc averages might yield more balanced and representative domain styles. Exploring this trade-off experimentally would provide valuable insight into the impact of temporal smoothing versus unbiased aggregation on the effectiveness of test-time style adaptation.

5.3 Future Work

Looking forward, several promising research directions could further improve and extend our approach.

One avenue is to incorporate advanced training-time DG techniques such as feature stylization and supervised contrastive learning. Jeon et al. (2021) introduced a framework that stylizes feature statistics to synthesize novel domain features while preserving class-critical information. They also employed a domain-aware contrastive loss to ensure domain-invariant yet class-discriminative representations. Exploring a similar feature stylization strategy or a contrastive objective in our training pipeline could enhance the model's robustness to unseen domains. Another interesting direction is to examine test-time input transformations as an alternative or complement to model adaptation. The incorporation of test-time style shifting proposed by Park et al. (2023) into our approach could reveal the relative merits of input-level against model-level adaptation. This may help build a unified framework that capitalizes on both approaches by using style shifts to quickly handle larger appearance gaps, while relying on model adaptation for remaining representation alignment. The comparison of our proposed method against these two very interesting approaches would offer a comprehensive picture of where each approach excels and how they might be combined effectively.

We can further make our TTA strategy adaptive to each target domain or even test batch. As proven by our results, not every domain shift requires the same level of adaptation. A future extension could be to develop a mode selection mechanism that uses signals like feature or prediction variance to decide when to deploy full TTA, light-weight adjustments or no adaptation at all. For example, if the models predictions in a new domain show high uncertainty or instability, the system could trigger a stronger adaptation mode, e.g. by updating more network layers or iterations. If variance is low, which would indicate well-aligned domains, the model could skip or reduce adaptation to avoid unnecessary drift. Such an adaptive TTA controller would make the model more reliable across varying degrees of domain shift.

Our results indicated that on certain datasets, a simple training-time augmentation like MixStyle (Zhou et al., 2021) can already yield robust performance, in some cases even outperforming TTA. This opens up the possibility of combining MixStyle with our TTA procedure to harness the strengths of both. Prior works have shown that integrating MixStyle into a DG pipeline alongside TTA can substantially boost accuracy. For instance, Park et al. (2023) demonstrated that their style-shifting scheme achieves its best results when combined with MixStyle augmentation during training. Similarly, Yamashita and Hotta (2024) found that their MixStyle-based contrastive approach outperformed both standard TTA and standalone DG methods on multiple benchmarks. Inspired by these findings, a hybrid method that applies MixStyle during source training and then performs the TTA might be especially beneficial for datasets like VLCS, where style diversity is limited. We suggest that

this hybrid could include the synthetic feature statistics produced by MixStyle as its own domain and thus could be used to augment during inference.

Finally, an extension to additional benchmarks for verification of our approach would be beneficial. So far, evaluations have focused on the PACS and VLCS datasets; as future directions, targeting larger and more diverse DG benchmarks such as Office-Home or DomainNet could be of interest. Moreover, it would be valuable to assess performance on data from medical imaging. Medical imaging offers a compelling testbed. Pronounced domain shifts arise naturally across hospitals, scanner vendors, protocols and patient populations, which creates style and distribution changes similar to the challenges our method seeks to address. At the same time, the stakes for generalization are high, as clinical models must perform reliably when deployed outside the training site. Because retraining with local data is often infeasible due to privacy, regulatory or resource constraints, a TTA strategy that operates without labeled target data is especially well suited. Additionally, the growing availability of multi-center medical datasets provides a rich opportunity for LODO evaluations in this domain.

In summary, the results highlight both the potential and the limitations of TTA. On datasets with clear domain separation such as PACS, TTA can provide significant benefits, particularly when configured appropriately. On datasets with less separation and higher intra-domain variability like VLCS, the MixStyle baseline remains more robust. The additional analysis of variance and disagreement underscores that stable adaptation is as important as raw accuracy gains, and future work should explore approaches that enhance both dimensions simultaneously.

6 Conclusion

This thesis investigated whether TTA through the injection of training-domain feature statistics can improve domain generalization in image classification, and if so, at which layers of a deep network such interventions are most effective. The central research questions asked how layer-wise test-time style adaptation influences robustness under domain shift, and under what conditions it provides measurable benefits. The evidence gathered across extensive experiments with PACS and VLCS shows that the answer depends on the depth of intervention and on the nature of the dataset. Shallow and mid-level feature manipulations yield tangible improvements when the primary domain shift is stylistic, as in PACS, while deeper interventions consistently harm performance. On datasets where domain variation is less stylistic and more semantic, as in VLCS, test-time style injection rarely surpasses a strong baseline.

The study as carried out by using a LODO cross-validation setup to ensure that each target domain remained fully unseen during training, which created OOD deployment. The approach relied on a ResNet-50 backbone, which was chosen for its wide adoption in DG research and its clear layer hierarchy. This facilitated a systematic investigation of shallow, mid and deep interventions. Style statistics were extracted and maintained with EMA updates during training. At test time, forward hooks replaced the statistics of unseen-domain inputs with those from source domains to produce augmented predictions.

This methodological pipeline had several strengths. It ensured reproducibility through fixed seeds, systematic hyperparameter optimization and per-domain evaluation across folds. It also allowed direct comparison between the training-time augmenation baseline MixStyle (Zhou et al., 2021) and TTA. At the same time, it presented challenges. The maintenance of per-domain required careful separation to avoid distortions and combining layer outputs through averaging proved to be detrimental. Moreover, the LODO setup inherently reduces the amount of training data available in each fold, which amplified variability in smaller domains. Despite these challenges, the design was robust enough to yield patterns in how TTA interacts with depth and dataset characteristics.

Several contributions emerge from this thesis. First, it establishes depth as a critical factor in test-time adaptation. While theory and style transfer literature suggested this, the empirical confirmation across two benchmarks provides evidence for this claim. The contrast between the strong performance of shallow modes and the collapse of deep ones formalizes what had often been assumed. Second, the approach highlights how dataset properties interact with adaptation strategies. The stylistic variation of PACS proved to be effective for shallow adaptation techniques. The limits of this method were exposed through the more content-driven shifts of VLCS, which illustrates the non-interchangeability of benchmarks due to the nature of the shift included in the data. Third, the thesis shows that per-sample variance across augmented predictions can serve as a useful uncertainty measure. For the PACS benchmark, using this metric to identify uncertain outputs enabled an effective

selective prediction strategy, which improved overall accuracy. On VLCS, its usefulness varied, but even there it provided moderate signals in some domains. This demonstrates that TTA can serve a dual role, not only as an adaptation mechanism but also as a diagnostic tool for uncertainty. Finally, the work clarifies the relationship between training-time and test-time strategies. MixStyle (Zhou et al., 2021) offers broad robustness by blending styles during training; test-time injection offers targeted alignment to specific source styles. On PACS, the two can reach similar outcomes, with the test-time approach improving over the baseline in some cases, while on VLCS, the baseline often remains superior. This suggests that training-time diversity and test-time alignment towards known domains address different aspects of robustness and that their effectiveness appears to depend on the dataset's type of shift.

Overall, the findings present a nuanced view of test-time style injection as a strategy for DG. It is neither a universal solution nor an ineffective tool. Applied at appropriate network depths and in contexts where domain shifts are primarily stylistic, it can yield gains in accuracy and provide informative uncertainty estimates. By contrast, indiscriminate use, particularly at deeper layers or on datasets characterized by semantic variation, tends to undermine performance. These results highlight that adaptation must respect both the hierarchical organization of CNNs and the nature of the used datasets. The process also emphasized the value of reflective experimentation.

Beyond numerical outcomes, the research also illustrates the value of reflective experimentation. Some of the most instructive insights arose from the unexpected findings, such as the collapse of the average mode or the limited effectiveness on VLCS, which helped to delineate the boundaries of the method. These observations prompt further consideration of how best to combine statistics across layers and how to determine when test-time adaptation is warranted.

In conclusion, this thesis contributes to a more refined understanding of test-time adaption for domain generalization. It demonstrates that style-statistics injection can be an effective tool, but only under specific conditions. By clarifying where and why it succeeds, the study adds to the broader effort of developing models that remain robust under distribution shift.

A Appendix

A.1 Algorithms

In this section, we provide pseudocode for the core methods introduced in our work. The algorithms illustrate how style statistics are extracted, managed and applied during test-time augmentation. They form the foundation of our method for domain-aware TTA to improve robustness on unseen target domains.

- Algorithm 1: Describes how domain-specific mean and variance statistics are obtained for a given domain as well as retrieved from pre-computed dictionaries. Depending on the chosen mode, statistics can be taken from one layer, a subset of layers or averaged across layers.
- Algorithm 2: Shows how intermediate feature maps are normalized and restyled during the forward pass of a network by injecting the domain-specific style statistics extracted after model training. This mechanism enables controlled feature adaptation without retraining the model.
- Algorithm 3: Explains the procedure for collecting and persisting style statistics from a trained model. It involves loading checkpoints, attaching hooks to feature extractors, running dummy forwards and saving extracted statistics for later use.
- Algorithm 4: Summarizes the initialization and inference process of our TTA classifier. It prepares the model, loads source-domain style statistics and during prediction applies restyling hooks for different target domains to enable domain-aware feature adaptation at test-time.

Algorithm 1: Get Style Statistics

Function `GET_STYLE_STATS(`*domain_idx, optional layer_idx*`):`

- **if** *layer_idx is given* **then**
  - **return** $(\mu\_dict[layer\_idx][domain\_idx], \sigma\_dict[layer\_idx][domain\_idx])$ reshaped to $[1, C, 1, 1]$;
- **switch** *mode* **do**
  - **case** *single* **do**
    - $\ell \leftarrow target\_layer$;
    - **return** $\mu\_dict[\ell][domain\_idx], \sigma\_dict[\ell][domain\_idx] \rightarrow [1, C, 1, 1]$;
  - **case** *selective* **do**
    - $L \leftarrow target\_layer\ list$;
    - $\mu \leftarrow mean(\{\mu\_dict[\ell][domain\_idx] \mid \ell \in L\})$;
    - $\sigma \leftarrow mean(\{\sigma\_dict[\ell][domain\_idx] \mid \ell \in L\})$;
    - **return** $\mu, \sigma \rightarrow [1, C, 1, 1]$;
  - **case** *average* **do**
    - $\mu_s, \sigma_s \leftarrow [\,]$;
    - **for** ℓ *in sorted(all layers)* **do**
      - $\mu_s.append(interpolate\_to\_size(\mu\_dict[\ell][domain\_idx], 256))$;
      - $\sigma_s.append(interpolate\_to\_size(\sigma\_dict[\ell][domain\_idx], 256))$;
    - $\mu \leftarrow mean(stack(\mu_s))$;
    - $\sigma \leftarrow mean(stack(\sigma_s))$;
    - **return** $\mu, \sigma \rightarrow [1, 256, 1, 1]$;

Algorithm 2: Domain-Aware Hook Forward Pass

Function `DOMAIN_AWARE_HOOK_FORWARD(`*output, mode, layer_idx, domain_idx*`):`

```
if mode = "direct" then
    (μ, σ) ← style_stats.get_style_stats(domain_idx, layer_idx)
else if mode = "file" then
    (μ, σ) ← read_from_loaded_stats(layer_idx, modus);
    if modus = "average" then
        adjust (μ, σ) by repeating to match channel count
reshape (μ, σ) → [1, C, 1, 1];
feat_μ ← mean(output over spatial dims);
feat_σ ← std(output over spatial dims);
normalized ← (output − feat_μ)/(feat_σ + ε);
transformed ← normalized ∗ σ + μ;
return transformed
```

Algorithm 3: Extract Style Statistics from Saved Model

Function `EXTRACT_FROM_SAVED_MODEL(`*model_path, domain_name, ModelClass, model_args*`):`

```
model ← load ModelClass with checkpoint(model_path);
model.enable_style_stats(True);
model.eval();
if domain_indices not provided then
    domain_indices ← all domains;
foreach extractor ∈ extractors do
    foreach domain_idx ∈ domain_indices do
        attach_hooks(model, extractor, domain_idx);
        model(dummy_input, domain_idx);
        remove_hooks();
        transfer_stats(model.style_stats, extractor, domain_idx);
save_all_extractors(domain_name);
```

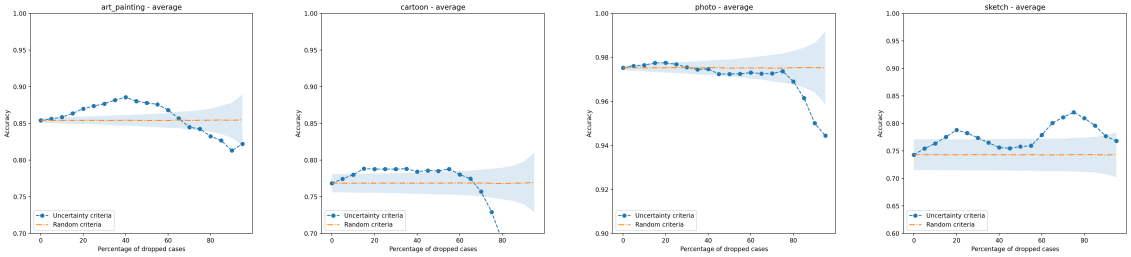
Algorithm 4: TTAClassifier: Init and Predict

```
Class TTAClassifier(model, stats_root, test_domain, mode, device,
 domain_names, seed):
    move model to device
    freeze model parameters
    set model to eval mode
    target_layers ← parse(mode)
        single_k → [k]
        selective_i_j → [i, j]
        average → [0, 1, 2, 3]
    style_stats ← empty dict
    foreach src_domain ∈ domain_names except test_domain do
        path ← stats_root / seed/ style_stats / test_domain / mode / pth file
        if file exists(path) then
            style_stats[src_domain] ← load(path)
    Method PREDICT(test_loader):
        available_targets ← domain_names \ {test_domain}
        foreach (images, labels, _) ∈ test_loader do
            baseline_logits ← model(images)
            foreach target_domain ∈ available_targets do
                handles ← empty list
                foreach layer_idx ∈ target_layers do
                    (μ, σ) ← style_stats[target_domain][layer_idx]
                    hook ← DomainAwareHook( style_stats = (μ, σ), layer_idx =
                     layer_idx, domain_idx = target_domain )
                    handle ←
                     register_forward_hook(model.layer(layer_idx).last_block,
                     hook)
                    append handle to handles
                logits_target ← model(images)
                foreach handle ∈ handles do
                    handle.remove()
```

A.2 Figures

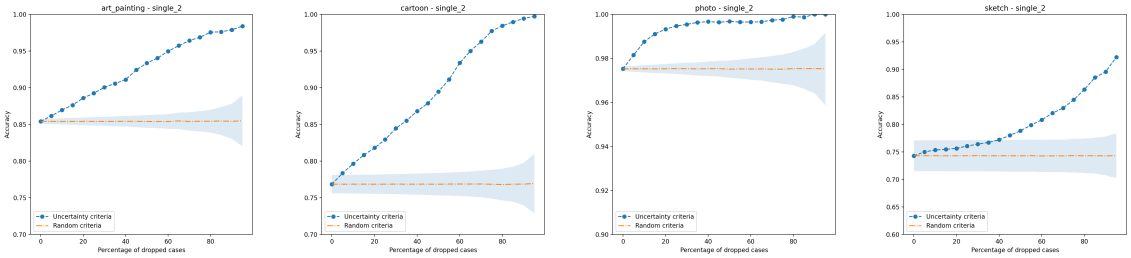
The following figures present uncertainty−accuracy drop curves for the remaining style statistic modes across different domains on the PACS and VLCS benchmarks. Each curve illustrates how predictive uncertainty can be leveraged as a rejection criterion to improve reliability. Each figure corresponds to one mode of style statistic usage, with separate subplots for the different target domains. Together, these plots provide a qualitative assessment of the effectiveness of uncertainty scores under different strategies.

A.2.1 PACS



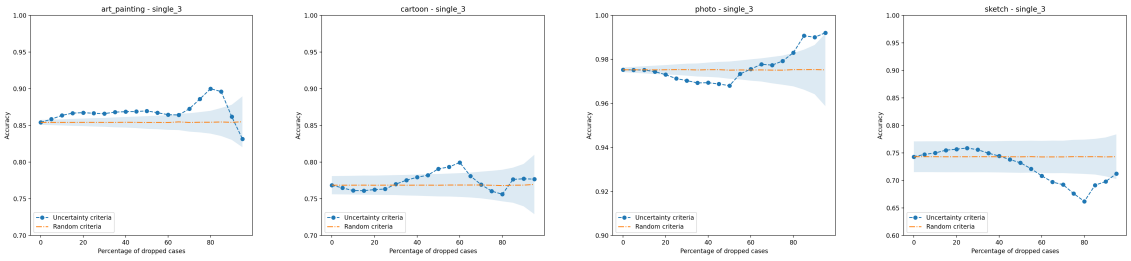
(a) *Art Painting* (b) *Cartoon* (c) *Photo* (d) *Sketch*

Figure 14: Mode *average*



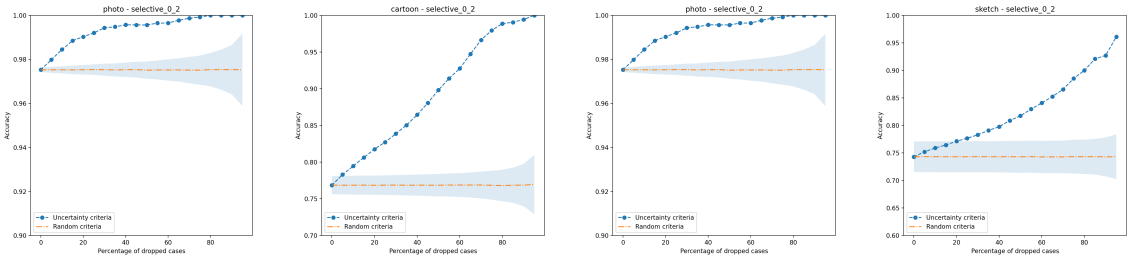
(a) *Art Painting* (b) *Cartoon* (c) *Photo* (d) *Sketch*

Figure 15: Mode *single_2*



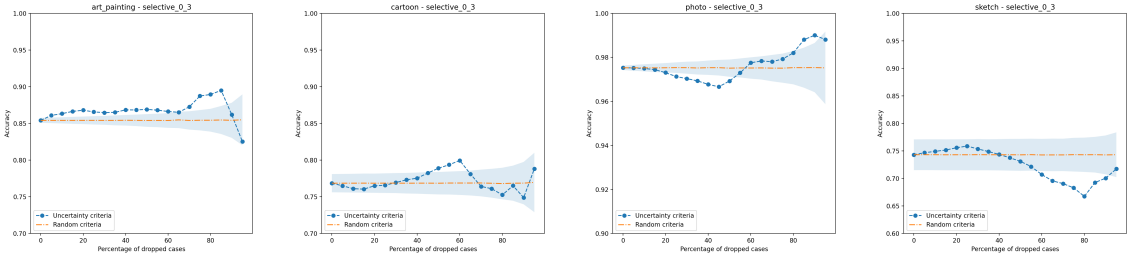
(a) *Art Painting* (b) *Cartoon* (c) *Photo* (d) *Sketch*

Figure 16: Mode *single_3*



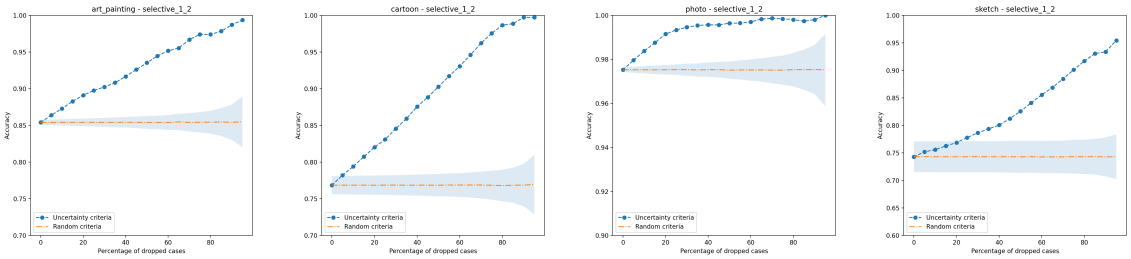
(a) *Art Painting* (b) *Cartoon* (c) *Photo* (d)

Figure 17: Mode *selective_0_2*



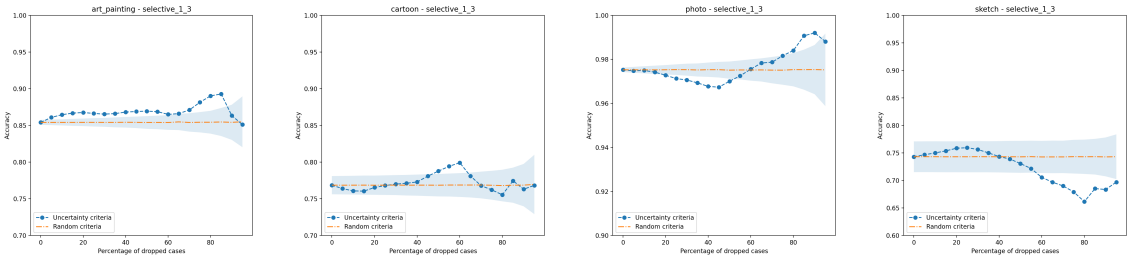
(a) (b) *Cartoon* (c) *Photo* (d)

Figure 18: Mode *selective_0_3*



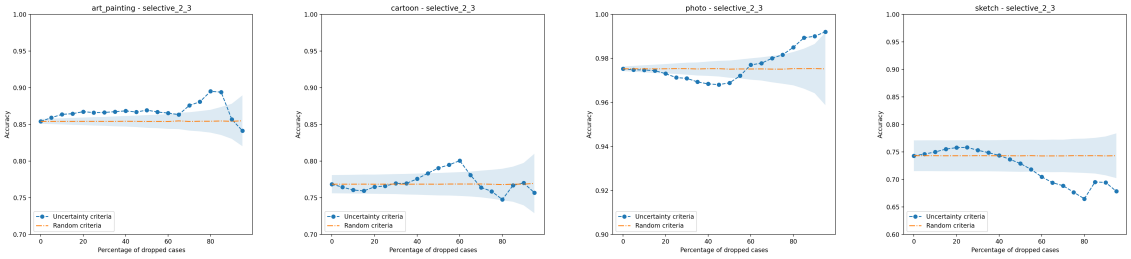
(a) *Art Painting* (b) *Cartoon* (c) *Photo* (d) *Sketch*

Figure 19: Mode *selective_1_2*



(a) *Art Painting* (b) *Cartoon* (c) *Photo* (d) *Sketch*

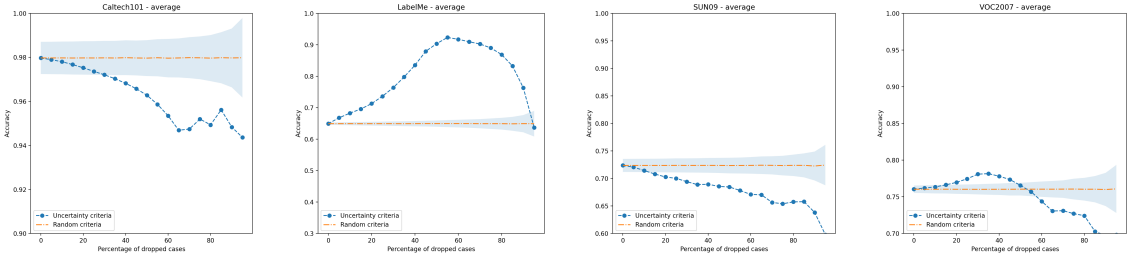
Figure 20: Mode *selective_1_3*



(a) *Art Painting* (b) *Cartoon* (c) *Photo* (d) *Sketch*

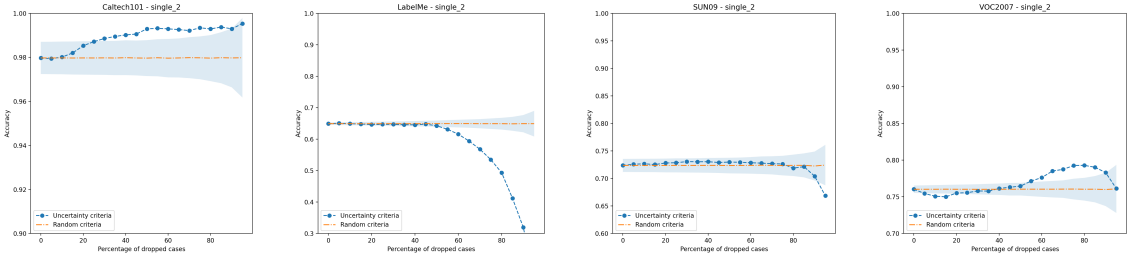
Figure 21: Mode *selective_2_3*

A.2.2 VLCS



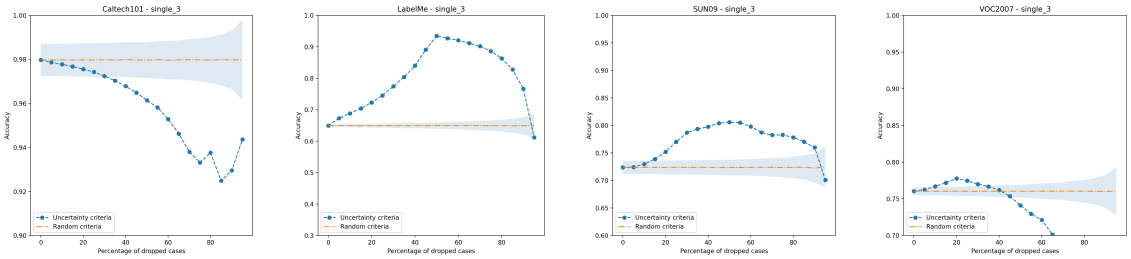
(a) *Caltech101* (b) *LabelMe* (c) *SUN09* (d) *VOC2007*

Figure 22: Mode *average*



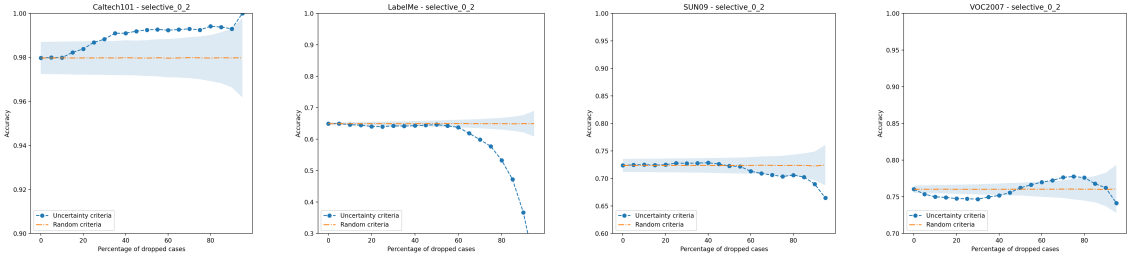
(a) *Caltech101* (b) *LabelMe* (c) *SUN09* (d) *VOC2007*

Figure 23: Mode *single_2*



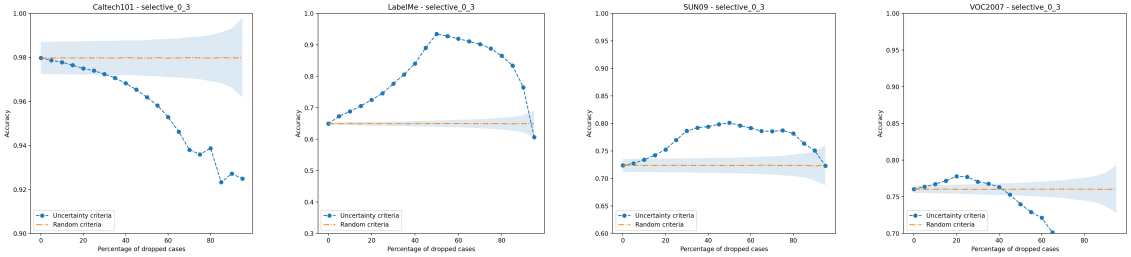
(a) *Caltech101* (b) *LabelMe* (c) *SUN09* (d) *VOC2007*

Figure 24: Mode *single_3*



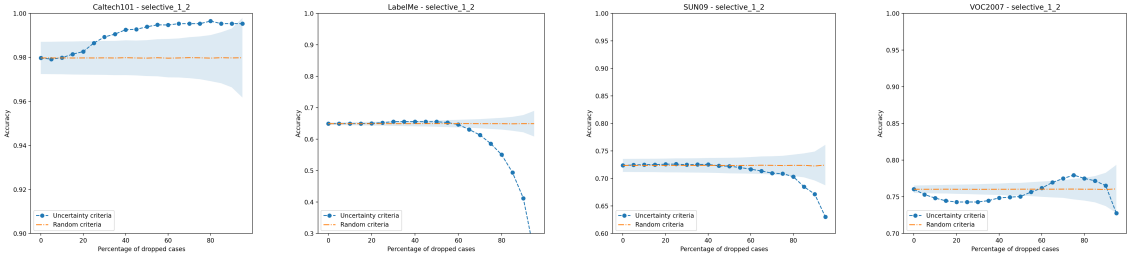
(a) *Caltech101* (b) *LabelMe* (c) *SUN09* (d) *VOC2007*

Figure 25: Mode *selective_0_2*



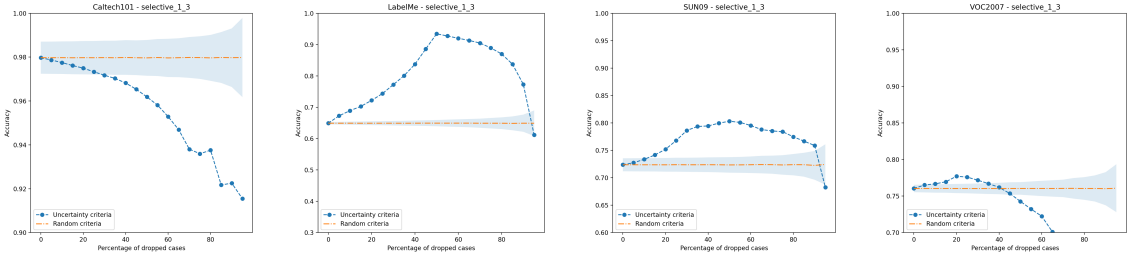
(a) *Caltech101* (b) *LabelMe* (c) *SUN09* (d) *VOC2007*

Figure 26: Mode *selective_0_3*



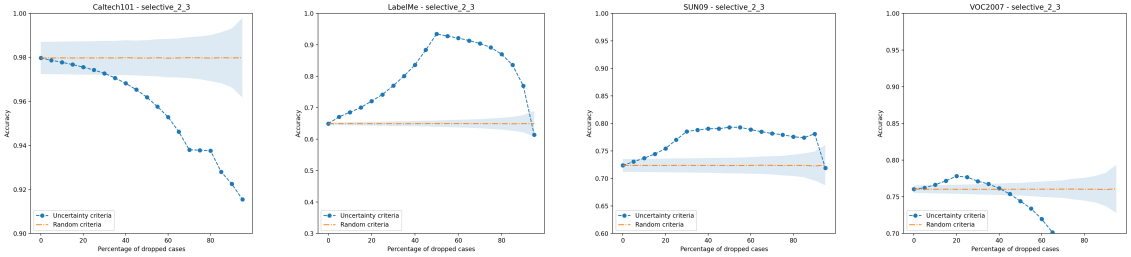
(a) *Caltech101* (b) *LabelMe* (c) *SUN09* (d) *VOC2007*

Figure 27: Mode *selective_1_2*



(a) *Caltech101* (b) *LabelMe* (c) *SUN09* (d) *VOC2007*

Figure 28: Mode *selective_1_3*



(a) *Caltech101* (b) *LabelMe* (c) *SUN09* (d) *VOC2007*

Figure 29: Mode *selective_2_3*

A.3 Tables

A.3.1 Hyperparameter Configurations

These tables list the hyperparameter configurations used during model training on the PACS and VLCS benchmarks. Reported are the optimizer settings, learning rate schedules, regularization parameters and other key values that governed training. These configurations provide transparency and reproducibility for the reported experimental results.

Table 9: Hyperparameter configuration used for model training for the PACS dataset.

Hyperparameter	**Value**
Learning Rate	0.0039615723
Optimizer	SGD
Scheduler	ReduceLROnPlateau
Weight Decay	0.0006719769
Batch Size	8
Momentum	0.2070932
Nesterov	False
ReduceLROnPlateau – Factor	0.3742400
ReduceLROnPlateau – Patience	4
Dropout	0.1548816

Table 10: Hyperparameter configuration used for model training for the VLCS dataset.

Hyperparameter	**Value**
Learning Rate	2.23×10^{-5}
Optimizer	Adam
Scheduler	ReduceLROnPlateau
Weight Decay	0.0001243755
Batch Size	64
β_1	0.9212345
β_2	0.9409845
Epsilon	5.3319572×10^{-7}
ReduceLROnPlateau – Factor	0.1976971
ReduceLROnPlateau – Patience	4
Dropout	0.1348674

A.3.2 Accuracy Scores

These tables report the mean prediction accuracies obtained during TTA when augmented under different style statistic modes. The results are presented separately for PACS and VLCS, with each table corresponding to one held-out test domain. Accuracies are averaged across seeds, with the respective MixStyle (Zhou et al., 2021) baseline included for comparison. Presented values offer insight into how domain-aware augmentation strategies affect performance across datasets and target domains.

PACS

Table 11: Mean Accuracy for Test Domain *Art Painting*, augmented with Feature Statistics towards the Trainings Domains, across all modes and seeds

Mode	**Mean Acc per Mode for Target Domain in %**		
	Cartoon	Photo	Sketch
single_0	86.88 ± 0.39	86.62 ± 1.04	87.0 ± 1.3
single_1	$\mathbf{88.14 \pm 0.51}$	$\mathbf{86.87 \pm 0.91}$	$\mathbf{87.4 \pm 1.74}$
single_2	84.2 ± 1.17	82.44 ± 2.81	82.54 ± 3.19
single_3	11.34 ± 2.65	17.88 ± 6.99	11.29 ± 2.0
selective_0_1	87.5 ± 0.68	86.72 ± 1.37	86.61 ± 2.03
selective_0_2	85.2 ± 1.2	82.54 ± 3.27	82.6 ± 2.96
selective_0_3	11.34 ± 2.65	17.88 ± 6.99	11.29 ± 2.0
selective_1_2	84.78 ± 0.96	81.53 ± 3.52	82.02 ± 2.19
selective_1_3	11.34 ± 2.65	17.88 ± 6.99	11.29 ± 2.0
selective_2_3	11.34 ± 2.65	17.88 ± 6.99	11.29 ± 2.0
average	16.49 ± 3.5	16.49 ± 3.5	16.49 ± 3.5
Baseline for Test Domain	85.75 ± 1.77		

Table 12: Mean Accuracy for Test Domain *Cartoon*, augmented with Feature Statistics towards the Trainings Domains, across all modes and seeds

Mode	**Mean accuracy across seeds for Mode in %**		
	Art Painting	Photo	Sketch
single_0	74.35 ± 1.62	76.14 ± 1.65	78.88 ± 0.92
single_1	78.13 ± 0.9	78.95 ± 1.32	$\mathbf{81.64 \pm 1.45}$
single_2	78.12 ± 1.33	80.13 ± 0.62	79.04 ± 1.28
single_3	18.02 ± 1.28	14.46 ± 2.56	15.37 ± 1.06
selective_0_1	77.1 ± 0.3	77.94 ± 1.77	81.09 ± 1.2
selective_0_2	$\mathbf{78.57 \pm 0.84}$	$\mathbf{80.63 \pm 1.08}$	79.02 ± 1.34
selective_0_3	18.02 ± 1.28	14.46 ± 2.56	15.37 ± 1.06
selective_1_2	78.56 ± 1.06	80.28 ± 1.71	79.51 ± 1.37
selective_1_3	18.02 ± 1.28	14.46 ± 2.56	15.37 ± 1.06
selective_2_3	18.02 ± 1.28	14.46 ± 2.56	15.37 ± 1.06
average	16.6 ± 0.0	16.6 ± 0.0	16.6 ± 0.0
Baseline for Test Domain	77.7 ± 2.53		

Table 13: Mean Accuracy for Test Domain *Photo*, augmented with Feature Statistics towards the Trainings Domains, across all modes and seeds

Mode	**Mean accuracy across seeds for Mode in %**		
	Art Painting	Cartoon	Sketch
single_0	$\mathbf{97.55 \pm 0.39}$	$\mathbf{97.19 \pm 0.33}$	$\mathbf{96.61 \pm 0.07}$
single_1	97.25 ± 0.39	96.79 ± 0.58	96.03 ± 0.31
single_2	96.01 ± 1.37	95.75 ± 0.48	96.03 ± 0.31
single_3	15.97 ± 8.57	18.19 ± 7.08	10.98 ± 0.14
selective_0_1	97.07 ± 0.53	96.75 ± 0.19	96.27 ± 0.12
selective_0_2	95.87 ± 1.18	95.29 ± 0.1	95.27 ± 0.42
selective_0_3	15.97 ± 8.57	18.19 ± 7.08	10.98 ± 0.14
selective_1_2	94.97 ± 2.13	95.01 ± 0.75	94.95 ± 0.51
selective_1_3	15.97 ± 8.57	18.19 ± 7.08	10.98 ± 0.14
selective_2_3	15.97 ± 8.57	18.19 ± 7.08	10.98 ± 0.14
average	11.44 ± 0.61	11.18 ± 0.24	11.18 ± 0.24
Baseline for Test Domain	96.99 ± 0.69		

Table 14: Mean Accuracy for Test Domain *Sketch*, augmented with Feature Statistics towards the Trainings Domains, across all modes and seeds

Mode	**Mean accuracy across seeds for Mode in %**		
	Art Painting	Cartoon	Photo
single_0	71.73 ± 7.06	72.98 ± 6.1	73.5 ± 5.22
single_1	$\mathbf{78.19 \pm 3.21}$	$\mathbf{79.18 \pm 2.25}$	$\mathbf{79.02 \pm 3.15}$
single_2	73.61 ± 4.55	69.69 ± 4.77	71.16 ± 6.02
single_3	14.3 ± 8.86	14.46 ± 9.0	7.19 ± 7.24
selective_0_1	76.81 ± 4.32	77.32 ± 2.68	77.35 ± 5.16
selective_0_2	74.3 ± 2.22	70.97 ± 4.19	73.26 ± 3.49
selective_0_3	14.3 ± 8.86	14.46 ± 9.0	7.19 ± 7.24
selective_1_2	75.57 ± 4.11	71.15 ± 5.51	72.84 ± 6.33
selective_1_3	14.3 ± 8.86	14.46 ± 9.0	7.19 ± 7.24
selective_2_3	14.3 ± 8.86	14.46 ± 9.0	7.19 ± 7.24
average	9.26 ± 9.0	9.26 ± 9.0	9.26 ± 9.0
Baseline for Test Domain	73.84 ± 4.12		

VLCS

Table 15: Mean Accuracy for Test Domain *Caltech101*, augmented with Feature Statistics towards the Trainings Domains, across all modes and seeds

Mode	**Mean accuracy across seeds for Mode in %**		
	LabelMe	SUN09	VOC2007
single_0	*97.74 ± 0.56*	*97.5 ± 0.57*	*97.55 ± 0.43*
single_1	96.28 ± 1.38	96.56 ± 1.34	96.68 ± 1.05
single_2	87.58 ± 4.56	90.08 ± 4.02	92.77 ± 3.14
single_3	43.88 ± 30.48	61.48 ± 0.0	61.48 ± 0.0
selective_0_1	95.71 ± 1.61	95.64 ± 1.74	96.0 ± 1.25
selective_0_2	85.82 ± 4.81	88.5 ± 4.6	90.32 ± 3.36
selective_0_3	43.88 ± 30.48	61.48 ± 0.0	61.48 ± 0.0
selective_1_2	83.49 ± 5.82	85.65 ± 5.09	89.54 ± 3.15
selective_1_3	43.88 ± 30.48	61.48 ± 0.0	61.48 ± 0.0
selective_2_3	43.88 ± 30.48	61.48 ± 0.0	61.48 ± 0.0
average	61.48 ± 0.0	61.48 ± 0.0	61.48 ± 0.0
Baseline for Test Domain	$\mathbf{97.97 \pm 0.73}$		

Table 16: Mean Accuracy for Test Domain *LabelMe*, augmented with Feature Statistics towards the Trainings Domains, across all modes and seeds

Mode	**Mean accuracy across seeds for Mode in %**		
	Caltech101	SUN09	VOC2007
single_0	62.76 ± 0.25	*62.87 ± 0.25*	64.2 ± 0.12
single_1	*63.93 ± 0.47*	62.72 ± 0.25	*64.72 ± 0.38*
single_2	63.75 ± 1.66	61.01 ± 2.41	63.44 ± 1.93
single_3	46.57 ± 0.0	46.57 ± 0.0	46.57 ± 0.0
selective_0_1	63.24 ± 0.08	62.07 ± 0.14	64.37 ± 0.55
selective_0_2	62.15 ± 1.67	59.01 ± 2.72	62.47 ± 2.02
selective_0_3	46.57 ± 0.0	46.57 ± 0.0	46.57 ± 0.0
selective_1_2	62.77 ± 1.63	59.06 ± 2.81	63.19 ± 2.31
selective_1_3	46.57 ± 0.0	46.57 ± 0.0	46.57 ± 0.0
selective_2_3	46.57 ± 0.0	46.57 ± 0.0	46.57 ± 0.0
average	46.57 ± 0.0	46.57 ± 0.0	46.57 ± 0.0
Baseline for Test Domain	**64.91 ± 0.36**		

Table 17: Mean Accuracy for Test Domain *SUN09*, augmented with Feature Statistics towards the Trainings Domains, across all modes and seeds

Mode	**Mean accuracy across seeds for Mode in %**		
	Caltech101	LabelMe	VOC2007
single_0	69.79 ± 1.38	70.35 ± 1.53	*70.7 ± 1.36*
single_1	*69.84 ± 0.53*	*70.48 ± 1.2*	70.07 ± 1.14
single_2	59.51 ± 4.89	57.98 ± 0.13	66.12 ± 2.81
single_3	38.51 ± 0.0	28.4 ± 0.0	38.51 ± 0.0
selective_0_1	69.22 ± 0.41	69.69 ± 0.93	69.88 ± 1.13
selective_0_2	55.86 ± 4.88	57.5 ± 0.34	63.47 ± 2.66
selective_0_3	38.51 ± 0.0	28.4 ± 0.0	38.51 ± 0.0
selective_1_2	54.33 ± 4.63	56.93 ± 0.58	62.75 ± 3.23
selective_1_3	38.51 ± 0.0	28.4 ± 0.0	38.51 ± 0.0
selective_2_3	38.51 ± 0.0	28.4 ± 0.0	38.51 ± 0.0
average	38.51 ± 0.0	38.51 ± 0.0	38.51 ± 0.0
Baseline for Test Domain	**72.37 ± 1.18**		

Table 18: Mean Accuracy for Test Domain *VOC2007*, augmented with Feature Statistics towards the Trainings Domains, across all modes and seeds

Mode	**Mean accuracy across seeds for Mode in %**		
	Caltech101	LabelMe	SUN09
single_0	77.3 ± 1.06	76.9 ± 0.56	$\mathbf{76.87 \pm 0.71}$
single_1	$\mathbf{77.7 \pm 0.85}$	$\mathbf{77.03 \pm 0.86}$	76.5 ± 0.88
single_2	72.92 ± 3.52	66.35 ± 6.39	70.9 ± 1.69
single_3	44.4 ± 0.0	28.6 ± 13.68	44.4 ± 0.0
selective_0_1	76.95 ± 0.82	76.69 ± 0.79	75.71 ± 0.93
selective_0_2	70.35 ± 3.69	63.84 ± 5.88	69.1 ± 0.97
selective_0_3	44.4 ± 0.0	28.6 ± 13.68	44.4 ± 0.0
selective_1_2	69.32 ± 3.86	62.86 ± 5.72	67.87 ± 0.96
selective_1_3	44.4 ± 0.0	28.6 ± 13.68	44.4 ± 0.0
selective_2_3	44.4 ± 0.0	28.6 ± 13.68	44.4 ± 0.0
average	44.4 ± 0.0	44.4 ± 0.0	44.4 ± 0.0
Baseline for Test Domain	76.02 ± 0.49		

A.3.3 Linear Mixed-Effects Model

These tables provide additional results of the linear mixed-effect model analyses conducted for both PACS and VLCS. Reported are estimated fixed effects for different TTA modes and their comparison agains the MixStyle (Zhou et al., 2021) baseline as well as per-domain improvements relative to the baseline. Coefficients are shown with corresponding confidence intervals, while domain × mode breakdowns highlight how gains or drops vary across target domains. These results provide an additional perspective on the consistency and reliability of the observed accuracy differences under different strategies.

PACS

Table 19: Fixed Effects for PACS (TTA-only)

	Coef.	CI Low	CI High
Intercept [average]	0.134	0.027	0.241
selective_0_1	0.715	0.678	0.752
selective_0_2	0.694	0.657	0.731
selective_0_3	0.007	-0.030	0.044
selective_1_2	0.692	0.655	0.729
selective_1_3	0.007	-0.030	0.044
selective_2_3	0.007	-0.030	0.044
single_0	0.699	0.662	0.736
single_1	0.723	0.686	0.759
single_2	0.690	0.653	0.727
single_3	0.007	-0.030	0.044
seed Var	16.000		

single_k = C(mode)[T.single_k] selective_i_j = C(mode)[T.selective_i_j]

Table 20: Fixed Effects for PACS (MixStyle vs. best TTA modes)

	Coef.	CI Low	CI High
Intercept [base]	0.835	0.786	0.883
single_0	-0.002	-0.016	0.012
single_1	0.022	0.008	0.036
sel_0_1	0.014	0.000	0.028
seed Var	23.823	1.581	46.065

single_k = C(mode, Treatment(reference='base'))[T.single_k]
sel_i_j = C(mode, Treatment(reference='base'))[T.selective_i_j]

Table 21: Domains × Modes: Improvement over base for PACS (paired per seed)

Domain	Mode	Mean Δ	SD	SE	CI Low	CI High
Art Painting	sel_0_1	0.0157	0.0194	0.0112	-0.0325	0.0638
	sel_0_2	-0.0193	0.0230	0.0133	-0.0764	0.0377
	sel_1_2	-0.0260	0.0208	0.0120	-0.0777	0.0257
	single_0	0.0145	0.0153	0.0088	-0.0235	0.0526
	single_1	0.0209	0.0158	0.0091	-0.0184	0.0601
	single_2	-0.0232	0.0204	0.0118	-0.0738	0.0274
Cartoon	sel_0_1	0.0101	0.0182	0.0105	-0.0351	0.0552
	sel_0_2	0.0170	0.0162	0.0094	-0.0232	0.0573
	sel_1_2	0.0174	0.0218	0.0126	-0.0367	0.0716
	single_0	-0.0125	0.0091	0.0053	-0.0350	0.0101
	single_1	0.0187	0.0191	0.0110	-0.0287	0.0661
	single_2	0.0139	0.0123	0.0071	-0.0166	0.0444
Photo	sel_0_1	-0.0030	0.0040	0.0023	-0.0130	0.0071
	sel_0_2	-0.0152	0.0026	0.0015	-0.0215	-0.0088
	sel_1_2	-0.0202	0.0064	0.0037	-0.0362	-0.0042
	single_0	0.0012	0.0042	0.0024	-0.0092	0.0116
	single_1	-0.0013	0.0049	0.0029	-0.0136	0.0110
	single_2	-0.0106	0.0048	0.0028	-0.0226	0.0013
Sketch	sel_0_1	0.0331	0.0159	0.0092	-0.0064	0.0726
	sel_0_2	-0.0100	0.0036	0.0021	-0.0191	-0.0010
	sel_1_2	-0.0066	0.0303	0.0175	-0.0820	0.0688
	single_0	-0.0111	0.0575	0.0332	-0.1539	0.1317
	single_1	0.0495	0.0027	0.0016	0.0428	0.0562
	single_2	-0.0236	0.0241	0.0139	-0.0834	0.0362

Mean Δ = mean Improvement over base.
single_k = single_k sel_i_j = selective_i_j
All values are based on paired seeds ($n = 3$ per condition).

VLCS

Table 22: Fixed Effects for VLCS (TTA-only)

	Coef.	CI Low	CI High
Intercept [average]	0.477	0.417	0.538
sel_0_1	0.284	0.245	0.322
sel_0_2	0.213	0.176	0.250
sel_0_3	-0.036	-0.074	0.001
sel_1_2	0.204	0.167	0.241
sel_1_3	-0.036	-0.074	0.001
sel_2_3	-0.036	-0.074	0.001
single_0	0.293	0.256	0.330
single_1	0.291	0.254	0.329
single_2	0.233	0.196	0.270
single_3	-0.036	-0.074	0.001
seed Var	4.249	0.603	7.895

single_k = C(mode)[T.single_k] sel_i_j = C(mode)[T.selective_i_j]

Table 23: Fixed Effects for VLCS (MixStyle vs. best TTA modes)

	Coef.	CI Low	CI High
Intercept [base]	0.777	0.706	0.847
single_0	-0.006	-0.012	0.000
single_1	-0.008	-0.013	0.002
sel_0_1	-0.014	-0.02	0.008
seed Var	314.071451	23.7665	604.3764

single_k = C(mode, Treatment(reference='base'))[T.single_k]
sel_i_j = C(mode, Treatment(reference='base'))[T.selective_i_j]

Table 24: Domain × Mode: Improvement over base for VLCS (paired per seed)

Domain	Mode	Mean Δ	SD	SE	CI Low	CI High
Caltech101	sel_0_1	-0.0193	0.0069	0.0040	-0.0363	-0.0023
	sel_0_2	-0.0950	0.0335	0.0194	-0.1783	-0.0118
	sel_1_2	-0.1149	0.0377	0.0218	-0.2085	-0.0212
	single_0	-0.0012	0.0038	0.0022	-0.0107	0.0084
	single_1	-0.0121	0.0039	0.0022	-0.0217	-0.0025
	single_2	-0.0757	0.0297	0.0172	-0.1496	-0.0018
LabelMe	sel_0_1	-0.0165	0.0027	0.0015	-0.0230	-0.0099
	sel_0_2	-0.0366	0.0180	0.0104	-0.0814	0.0082
	sel_1_2	-0.0320	0.0177	0.0102	-0.0758	0.0119
	single_0	-0.0159	0.0042	0.0024	-0.0264	-0.0054
	single_1	-0.0108	0.0015	0.0008	-0.0145	-0.0072
	single_2	-0.0214	0.0166	0.0096	-0.0627	0.0200
SUN09	sel_0_1	-0.0252	0.0044	0.0026	-0.0362	-0.0142
	sel_0_2	-0.1317	0.0305	0.0176	-0.2075	-0.0559
	sel_1_2	-0.1411	0.0317	0.0183	-0.2199	-0.0624
	single_0	-0.0184	0.0071	0.0041	-0.0359	-0.0008
	single_1	-0.0198	0.0038	0.0022	-0.0293	-0.0104
	single_2	-0.1091	0.0323	0.0186	-0.1893	-0.0290
VOC2007	sel_0_1	0.0050	0.0071	0.0041	-0.0126	0.0226
	sel_0_2	-0.0818	0.0131	0.0076	-0.1144	-0.0493
	sel_1_2	-0.0926	0.0130	0.0075	-0.1249	-0.0604
	single_0	0.0108	0.0064	0.0037	-0.0051	0.0267
	single_1	0.0113	0.0072	0.0042	-0.0067	0.0293
	single_2	-0.0589	0.0178	0.0103	-0.1031	-0.0147

Mean Δ = mean Improvement over base.
single_k = single_k sel_i_j = selective_i_j
All values are based on paired seeds ($n = 3$ per condition).

A.4 Use of Generative AI

Generative AI, such as ChatGPT, has been used by the author to paraphrase sections of text, create certain citation references, and adjust spelling and punctuation. It was also used to generate sections of code for the described software project. All generated text and code has been carefully proofread and adjusted by the author.

Bibliography

Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems, 2015. URL `https://www.tensorflow.org/`. Software available from tensorflow.org.

Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. Optuna: A Next-generation Hyperparameter Optimization Framework. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 2623–2631, 2019.

Simone Angarano, Mauro Martini, Francesco Salvetti, Vittorio Mazzia, and Marcello Chiaberge. Back-to-Bones: Rediscovering the role of backbones in domain generalization. *Pattern Recognition*, 156:110762, 2024. doi: 10.1016/j.patcog.2024.110762.

Martin Arjovsky, Léon Bottou, Ishaan Gulrajani, and David Lopez-Paz. Invariant Risk Minimization. *arXiv preprint*, 2020. URL `https://arxiv.org/abs/1907.02893`.

Gilles Blanchard, Gyemin Lee, and Clayton Scott. Generalizing from several related classification tasks to a new unlabeled sample. In *Advances in Neural Information Processing Systems*, volume 24, 2011.

Zhaowei Cai, Avinash Ravichandran, Subhransu Maji, Charless Fowlkes, Zhuowen Tu, and Stefano Soatto. Exponential Moving Average Normalization for Self-Supervised and Semi-Supervised Learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 194–203, 2021.

Woong-Gi Chang, Tackgeun You, Seonguk Seo, Suha Kwak, and Bohyung Han. Domain-Specific Batch Normalization for Unsupervised Domain Adaptation. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7346–7354, 2019. doi: 10.1109/CVPR.2019.00753.

Liang Chen, Yong Zhang, Yibing Song, Ying Shan, and Lingqiao Liu. Improved Test-Time Adaptation for Domain Generalization. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 24172–24182, 2023.

Wonjeong Choi, Jungwuk Park, Dong-Jun Han, Younghyun Park, and Jaekyun Moon. Consistency-Guided Temperature Scaling Using Style and Content Information for Out-of-Domain Calibration. *arXiv preprint*, 2024. URL `http://arxiv.org/abs/2402.15019`.

Pedro Conde, Tiago Barros, Rui L. Lopes, Cristiano Premebida, and Urbano J. Nunes. Approaching Test Time Augmentation in the Context of Uncertainty Calibration for Deep Neural Networks. *arXiv preprint*, 2024. URL `https://arxiv.org/abs/2304.05104`.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009.

Chen Fang, Ye Xu, and Daniel N. Rockmore. 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*, pages 1657–1664, 2013.

Tiago M. Fragoso, Wesley Bertoli, and Francisco Louzada. Bayesian Model Averaging: A Systematic Review and Conceptual Classification. *International Statistical Review*, 86(1):1–28, 2018. doi: 10.1111/insr.12243.

Yarin Gal and Zoubin Ghahramani. Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning. In *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pages 1050–1059, 2016. URL `https://proceedings.mlr.press/v48/gal16.html`.

Leon Gatys, Alexander S Ecker, and Matthias Bethge. Texture Synthesis Using Convolutional Neural Networks. In *Advances in Neural Information Processing Systems*, volume 28, 2015.

Robert Geirhos, Patricia Rubisch, Claudio Michaelis, Matthias Bethge, Felix A. Wichmann, and Wieland Brendel. ImageNet-trained CNNs are biased towards texture; increasing shape bias improves accuracy and robustness. In *International Conference on Learning Representations*, 2018.

Ishaan Gulrajani and David Lopez-Paz. In Search of Lost Domain Generalization. *arXiv preprint*, 2020. URL `https://arxiv.org/abs/2007.01434`.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016. doi: 10.1109/CVPR.2016.90.

Dan Hendrycks, Norman Mu, Ekin D. Cubuk, Barret Zoph, Justin Gilmer, and Balaji Lakshminarayanan. AugMix: A Simple Data Processing Method to Improve Robustness and Uncertainty. *arXiv preprint*, 2020. URL `https://arxiv.org/abs/1912.02781`.

Xun Huang and Serge Belongie. Arbitrary Style Transfer in Real-Time with Adaptive Instance Normalization. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 1501–1510, 2017.

Yunzhi Huang, Luyi Han, and Haoran Dou. Generative feature style augmentation for domain generalization in medical image segmentation. *Pattern Recognition*, 162:111416, 2025. doi: 10.1016/j.patcog.2025.111416.

Sergey Ioffe. Batch Renormalization: Towards Reducing Minibatch Dependence in Batch-Normalized Models. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

Sergey Ioffe and Christian Szegedy. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. In *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 448–456, 2015. URL `https://proceedings.mlr.press/v37/ioffe15.html`.

Md Tauhidul Islam, Zixia Zhou, Hongyi Ren, Masoud Badiei Khuzani, Daniel Kapp, James Zou, Lu Tian, Joseph C. Liao, and Lei Xing. Revealing hidden patterns in deep neural network feature space continuum via manifold learning. *Nature Communications*, 14(1):8506, 2023. doi: 10.1038/s41467-023-43958-w.

Seogkyu Jeon, Kibeom Hong, Pilhyeon Lee, Jewook Lee, and Hyeran Byun. Feature stylization and domain-aware contrastive learning for domain generalization. *Proceedings of the 29th ACM International Conference on Multimedia*, pages 22–31, 2021. doi: 10.1145/3474085.3475271.

Jiming Jiang. *Linear and Generalized Linear Mixed Models and Their Applications*. Springer New York, New York, NY, 2007.

Vaibhav Khamankar, Sutanu Bera, Saumik Bhattacharya, Debashis Sen, and Prabir Kumar Biswas. Histopathological Image Analysis with Style-Augmented Feature Domain Mixing for Improved Generalization. *arXiv preprint*, 2023. URL `https://arxiv.org/abs/2310.20638`.

Masanari Kimura. Understanding Test-Time Augmentation. In *International Conference on Neural Information Processing*, pages 558–569, 2021.

Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy M. Hospedales. Deeper, Broader and Artier Domain Generalization. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 5542–5550, 2017. doi: 10.1109/ICCV.2017.591.

Pan Li, Da Li, Wei Li, Shaogang Gong, Yanwei Fu, and Timothy M. Hospedales. A Simple Feature Augmentation for Domain Generalization. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 8866–8875, 2021. doi: 10.1109/ICCV48922.2021.00876.

Junjie Mai, Chongzhi Gao, and Jun Bao. Domain Generalization Through Data Augmentation: A Survey of Methods, Applications, and Challenges. *Mathematics*, 13(5), 2025. doi: 10.3390/math13050824.

Krikamol Muandet, David Balduzzi, and Bernhard Schölkopf. Domain Generalization via Invariant Feature Representation. In *International Conference on Machine Learning*, pages 10–18, 2013. URL `https://proceedings.mlr.press/v28/muandet13.html`.

Arun Narayanan, Ananya Misra, Khe Chai Sim, Golan Pundak, Anshuman Tripathi, Mohamed Elfeky, Parisa Haghani, Trevor Strohman, and Michiel Bacchiani. Toward Domain-Invariant Speech Recognition via Large Scale Training. In *2018 IEEE Spoken Language Technology Workshop (SLT)*, pages 441–447, 2018. doi: 10.1109/SLT.2018.8639610.

Cheng Ouyang, Chen Chen, Surui Li, Zeju Li, Chen Qin, Wenjia Bai, and Daniel Rueckert. Causality-Inspired Single-Source Domain Generalization for Medical Image Segmentation. *IEEE Transactions on Medical Imaging*, 42(4):1095–1106, 2022. doi: 10.1109/TMI.2022.3224067.

Jungwuk Park, Dong-Jun Han, Soyeong Kim, and Jaekyun Moon. Test-Time Style Shifting: Handling Arbitrary Styles in Domain Generalization. In *Proceedings of the 40th International Conference on Machine Learning*, pages 27114–27131, 2023. URL `https://proceedings.mlr.press/v202/park23d.html`.

Xingchao Peng, Qinxun Bai, Xide Xia, Zijun Huang, Kate Saenko, and Bo Wang. Moment Matching for Multi-Source Domain Adaptation. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 1406–1415, 2019.

Manuel Schwonberg, Fadoua El Bouazati, Nico M. Schmidt, and Hanno Gottschalk. Augmentation-based Domain Generalization for Semantic Segmentation. In *2023 IEEE Intelligent Vehicles Symposium (IV)*, pages 1–8, 2023. doi: 10.1109/IV55152.2023.10186752.

Divya Shanmugam, Davis W. Blalock, Guha Balakrishnan, and John Guttag. When and Why Test-Time Augmentation Works. *arXiv preprint*, 2020. doi: 10.48550/arXiv.2011.11156.

Divya Shanmugam, Davis W. Blalock, Guha Balakrishnan, and J. Guttag. Better Aggregation in Test-Time Augmentation. *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 1214–1223, 2021. doi: 10.1109/ICCV48922.2021.00125.

Zeinab Sherkatghanad, Moloud Abdar, Mohammadreza Bakhtyari, and V. Makarenkov. BayTTA: Uncertainty-aware medical image classification with optimized test-time augmentation using Bayesian model averaging. *Knowledge-Based Systems*, 327:114–123, 2024. doi: 10.48550/arXiv.2406.17640.

Pratik Sheth, Raha Moraffah, K. Selçuk Candan, Adrienne Raglin, and Huan Liu. Domain Generalization–A Causal Perspective. *arXiv preprint*, 2022. URL `https://arxiv.org/abs/2209.15177`.

Antonio Torralba and Alexei A. Efros. Unbiased look at dataset bias. In *CVPR 2011*, pages 1521–1528, 2011. doi: 10.1109/CVPR.2011.5995347.

Laurens van der Maaten and Geoffrey Hinton. Visualizing Data using t-SNE. *Journal of Machine Learning Research*, 9:2579–2605, 2008. URL `https://www.jmlr.org/papers/volume9/vandermaaten08a/vandermaaten08a.pdf`.

Guotai Wang, Wenqi Li, Michael Aertsen, Jan Deprest, Sébastien Ourselin, and Tom Vercauteren. Aleatoric uncertainty estimation with test-time augmentation for medical image segmentation with convolutional neural networks. *Neurocomputing*, 338:34–45, 2019. doi: 10.1016/j.neucom.2019.01.103.

Yue Wang, Lei Qi, Yinghuan Shi, and Yang Gao. Feature-Based Style Randomization for Domain Generalization. *IEEE Transactions on Circuits and Systems for Video Technology*, 32(8):5495–5509, 2022. doi: 10.1109/TCSVT.2022.3152615.

Yunqi Wang, Furui Liu, Zhitang Chen, Yik-Chung Wu, Jianye Hao, Guangyong Chen, and Pheng Ann Heng. Contrastive-ACE: Domain Generalization Through Alignment of Causal Mechanisms. *IEEE Transactions on Image Processing*, 32: 235–250, 2023. doi: 10.1109/TIP.2022.3227457.

Yeming Wen, Ghassen Jerfel, Rafael Muller, Michael W. Dusenberry, Jasper Snoek, Balaji Lakshminarayanan, and Dustin Tran. Combining Ensembles and Data Augmentation can Harm your Calibration. *arXiv preprint*, 2021. URL `https://arxiv.org/abs/2010.09875`.

Qinwei Xu, Ruipeng Zhang, Ya Zhang, Yanfeng Wang, and Qi Tian. A Fourier-Based Framework for Domain Generalization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 14383–14392, 2021. doi: 10.1109/CVPR46437.2021.01415.

Kota Yamashita and Kazuhiro Hotta. MixStyle-Based Contrastive Test-Time Adaptation: Pathway to Domain Generalization. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 1029–1037, 2024. doi: 10.1109/CVPRW63382.2024.00109.

Rikiya Yamashita, Jin Long, Snikitha Banda, Jeanne Shen, and Daniel L. Rubin. Learning Domain-Agnostic Visual Representation for Computational Pathology Using Medically-Irrelevant Style Transfer Augmentation. *IEEE Transactions on Medical Imaging*, 40(12):3945–3954, 2021. doi: 10.1109/TMI.2021.3101985.

Jee Sook Yoon, Kwanseok Oh, Yooseung Shin, Maciej A. Mazurowski, and Heung-Il Suk. Domain Generalization for Medical Image Analysis: A Review. *Proceedings of the IEEE*, 112(10):1583–1609, 2024. doi: 10.1109/JPROC.2024.3507831.

Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? In *Advances in Neural Information Processing Systems*, volume 27, 2014.

Zhaoxia Yu, Michele Guindani, Steven F. Grieco, Lujia Chen, Todd C. Holmes, and Xiangmin Xu. Beyond t-test and ANOVA: applications of mixed-effects models for more rigorous statistical analysis in neuroscience research. *Neuron*, 110(1): 21–35, 2022. doi: 10.1016/j.neuron.2021.10.030.

Matthew D. Zeiler and Rob Fergus. Visualizing and Understanding Convolutional Networks. In *Computer Vision – ECCV 2014*, volume 8689, pages 818–833, 2014. doi: 10.1007/978-3-319-10590-1_53.

Hongyi Zhang, Moustapha Cisse, Yann N. Dauphin, and David Lopez-Paz. mixup: Beyond Empirical Risk Minimization. *arXiv preprint*, 2018. URL `https://arxiv.org/abs/1710.09412`.

Ling Zhang, Xiaosong Wang, Dong Yang, Thomas Sanford, Stephanie Harmon, Baris Turkbey, Ziyue Xu, Bradford J. Wood, Holger Roth, Andriy Myronenko, and Daguang Xu. Generalizing Deep Learning for Medical Image Segmentation to Unseen Domains via Deep Stacked Transformation. *IEEE Transactions on Medical Imaging*, 39(7):2531–2540, 2020. doi: 10.1109/TMI.2020.2973595.

Kaiyang Zhou, Yongxin Yang, Yu Qiao, and Tao Xiang. Domain Generalization with MixStyle. *arXiv preprint*, 2021. URL `https://arxiv.org/abs/2104.02008`.

Kaiyang Zhou, Ziwei Liu, Yu Qiao, Tao Xiang, and Chen Change Loy. Domain Generalization: A Survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(4):4396–4415, 2022a. doi: 10.1109/TPAMI.2022.3195549.

Ziqi Zhou, Lei Qi, Xin Yang, Dong Ni, and Yinghuan Shi. Generalizable Cross-Modality Medical Image Segmentation via Style Augmentation and Dual Normalization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 20856–20865, 2022b.

Declaration of Authorship

Ich erkläre hiermit gemäß §9 Abs. 12 APO, dass ich die vorstehende Abschlussarbeit selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe. Des Weiteren erkläre ich, dass die digitale Fassung der gedruckten Ausfertigung der Abschlussarbeit ausnahmslos in Inhalt und Wortlaut entspricht und zur Kenntnis genommen wurde, dass diese digitale Fassung einer durch Software unterstützten, anonymisierten Prüfung auf Plagiate unterzogen werden kann.

Place, Date

Signature