Transferability of COCO-Pretrained Object Detectors to Artistic VOC-Style Domains: A Case Study on Watercolor2k

Jeon Seungwoo

Abstract

Object detectors trained on large-scale datasets such as MS COCO often serve as universal feature extractors in transfer learning pipelines. However, their generalization ability to stylistically distinct, low-resource domains remains underexplored. This project investigates the transferability of a COCO-pretrained Faster R-CNN model to the Watercolor2k dataset—a small PASCAL VOC-style artistic object detection benchmark. We evaluate performance under various adaptation settings, including zero-shot transfer, head-only fine-tuning, partial backbone fine-tuning, and BatchNorm-only tuning. Our findings highlight that selective tuning of normalization layers alone can outperform full fine-tuning in both sample efficiency and accuracy, offering a promising direction for domain-adaptive detection in low-data regimes.

1. Introduction

Object detection has witnessed rapid progress due to deep learning models trained on large-scale datasets like COCO. However, deploying these detectors in visually distinct target domains—such as artistic or non-photorealistic datasets—poses a significant challenge. Watercolor2k is one such dataset, comprising images with watercolor-style distortions, making direct transfer from COCO difficult.

In this study, we explore the following research questions:

- Can COCO-pretrained detectors generalize in a zero-shot setting to stylistically shifted domains?
- What is the minimal level of fine-tuning required to achieve acceptable performance?
- How does the choice of trainable layers affect domain adaptation performance?
- Is it possible to achieve competitive results by fine-tuning only Batch Normalization layers?

We conduct a comprehensive ablation study using Faster R-CNN with ResNet-50 FPN backbone on Watercolor2k, comparing zero-shot inference, partial fine-tuning, and scratch training.

2. Background

2.1 Transfer Learning in Object Detection

Transfer learning in object detection aims to adapt a source-domain-trained detector to a target domain with limited annotations. Common practices include full fine-tuning or partial adaptation of specific layers.

2.2 Faster R-CNN

Faster R-CNN is a two-stage detector that comprises a region proposal network (RPN) and a region-based classification and regression head. The backbone network (ResNet-50) extracts multi-scale features which are shared across the two stages.

2.3 Watercolor2k and VOC-Style Datasets

Watercolor2k is a small-scale benchmark comprising 2,000 images derived from VOC categories (bicycle, bird, car, cat, dog, person), each stylized in an artistic domain. Despite sharing object semantics with VOC and COCO, the domain shift in appearance introduces significant generalization challenges.

3. Experiment

This section presents the empirical results of evaluating various training strategies on the Watercolor2k dataset using Faster R-CNN with a ResNet-50 backbone. We aim to understand how different transfer learning settings affect performance in small dataset, domain-shifted scenarios.

3.1. Experimental Setup

We conduct a series of experiments on the Watercolor2k dataset to evaluate how different fine-tuning strategies affect object detection performance when using a Faster R-CNN architecture pretrained on COCO. Our experiments are structured under three main settings: **Zero-shot**, **Fine-tuning**, and **Scratch** training. Additionally, we perform ablation studies to analyze the impact of freezing different subsets of the network during fine-tuning.

Dataset

We use the Watercolor2k dataset, which contains 2,000 images across 6 object classes: bicycle, bird, car, cat, dog, and person. The dataset provides annotations in COCO-style JSON format, making it compatible with standard PyTorch object detection frameworks.

Model

We adopt Faster R-CNN with a ResNet-50 FPN backbone, as implemented in the torchvision library. Pretrained weights on the COCO dataset are used in most configurations except for the scratch baseline.

Optimization and Training Setting

All models are trained using the Adam optimizer with a fixed learning rate of 0.0001 and a weight decay of 1e-4 to prevent overfitting. The training is conducted for up to 50 epochs with a batch size of 16. A

StepLR learning rate scheduler is used to decay the learning rate by a factor of 0.1 every 7 epochs. Input images are resized to 512 × 512 pixels, and data augmentation is applied during training, including random horizontal flipping with a probability of 0.5. The dataset consists of six object classes (excluding background), and both training and evaluation are performed on a single NVIDIA GeForce RTX 3090.

Fine-tuning Modes

We evaluate different fine-tuning modes under the "finetune" setting with varying backbone freezing strategies:

- **Head only**: Only the classification and regression heads are updated.
- **Layer4 + Head**: Only the final ResNet block (layer4) and head are trainable.
- **BN only**: Only BatchNorm layers are trainable (mimicking adaptive normalization).
- **Full fine-tuning**: All parameters are trainable.

These settings are controlled using the freeze_mode field in the config, and results are logged separately per configuration.

Evaluation Metric

We report Mean Average Precision (mAP) at three IoU thresholds:

- mAP@0.50
- mAP@0.75
- mAP@0.90

as computed by torchmetrics.detection.MeanAveragePrecision, which aligns with the COCO evaluation protocol.

3.2 Main Comparison: Transfer vs. Scratch

To evaluate the effectiveness of transfer learning for artistic domain adaptation, we compare several training strategies on the Watercolor2k dataset using a Faster R-CNN detector. The baselines include:

- Zero-Shot Inference (Pretrained only): A COCO-pretrained model is directly evaluated on the Watercolor2k dataset without any fine-tuning.
- Fine-Tuning: The same COCO-pretrained model is fine-tuned on the target dataset. We experiment with several degrees of backbone freezing in Section 3.3.
- Training from Scratch: The model is trained from random initialization (no pretraining) on the Watercolor2k dataset.

Setting	Pretrained	Trainable Parameters	mAP@0.5	mAP@0.75	mAP@0.9
Zero-shot	O	None	0.0021	0.00007	0.000003
Fine-tune Head only	O	Head	0.3698	0.1611	0.0084
Fine-tune Layer4 + Head	O	Layer4 + Head	0.3784	0.1668	0.0097

Fine-tune BatchNorm only	O	BatchNorm layers	0.3790	0.1672	0.0126
Full Fine-tuning	O	All	0.3468	0.1557	0.0095
Scratch (Random Init)	X	All	0.0700	0.0169	0.00004

Table.1. Comparison of Fine-Tuning Strategies on Watercolor2k

The zero-shot baseline, where the pretrained model is used without any adaptation, performs poorly across all IoU thresholds. The scratch setting, which trains all parameters from random initialization, also struggles due to the limited size and diversity of the Watercolor2k dataset.

In contrast, all transfer learning configurations significantly outperform these baselines. Among them, fine-tuning only the BatchNorm layers achieves the best performance across all mAP thresholds, suggesting that statistical alignment via BN adaptation is highly effective for domain adaptation.

3.2 Ablation Study: Effect of Fine-Tuning Scope

To understand which model components are most critical for domain adaptation, we conduct an ablation study across various fine-tuning scopes:

The performance gap between head-only and Layer4+head (0.3698 → 0.3784 at mAP@0.5) suggests that shallow feature refinement provides marginal gains. However, BatchNorm-only tuning not only outperforms all other methods, it also demonstrates the power of statistical adaptation without full parameter updates.

Interestingly, full fine-tuning underperforms despite higher training flexibility, which may be due to overfitting or deviation from well-initialized COCO features. These results suggest that selective fine-tuning is not only more efficient but also more effective in low data regimes.

3.3 Convergence and Stability

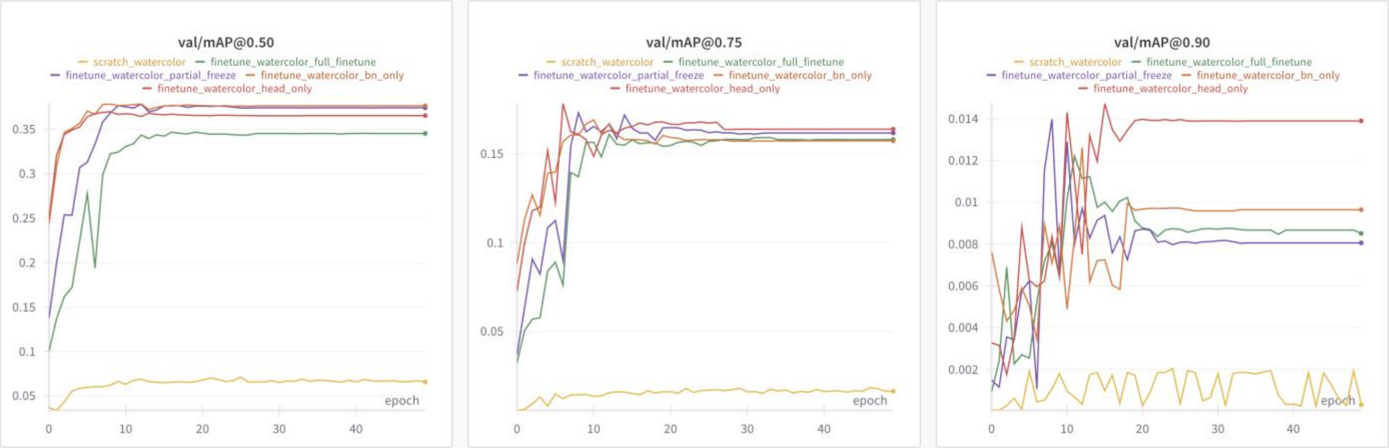


Fig.1. Convergence of validation mAP at different IoU thresholds during training

As illustrated in Figure 1, convergence behaviors vary significantly across training strategies:

- BatchNorm-only tuning achieves the fastest and smoothest convergence across all mAP thresholds.
- Head-only and Layer4+Head settings also show rapid stabilization within ~10 epochs.
- Full Fine-tuning converges more slowly and shows instability, possibly due to the higher parameter space and overfitting risk.
- Training from scratch exhibits the slowest and most unstable convergence, with minimal final performance.

These findings reinforce that lightweight fine-tuning approaches not only generalize better but also converge faster, making them especially suited for small target domains like Watercolor2k.

3.4. Visualization

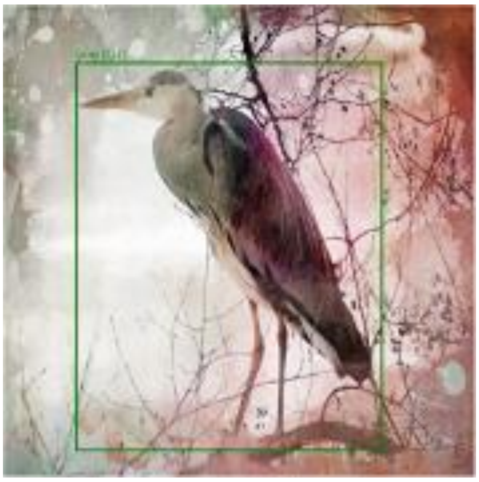
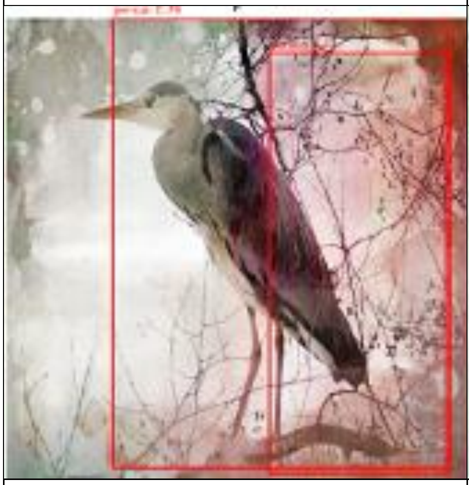
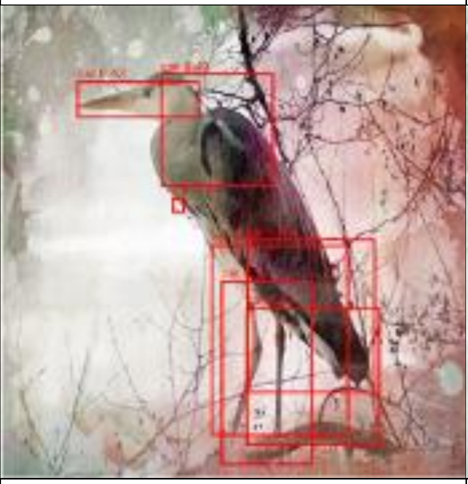
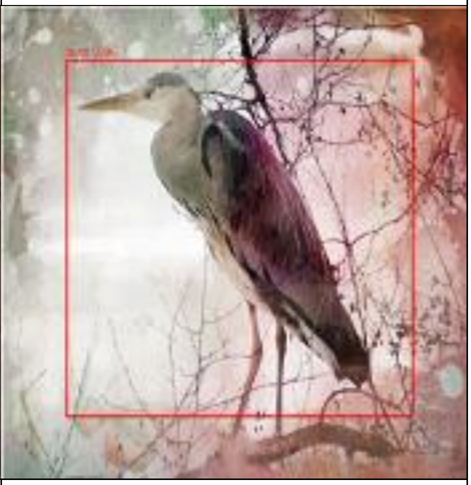


Fig.2. Ground Truth

Scratch	Zero-shot	Head only
		
Layer4 + Head	Full Finetune	BachNorm-Only

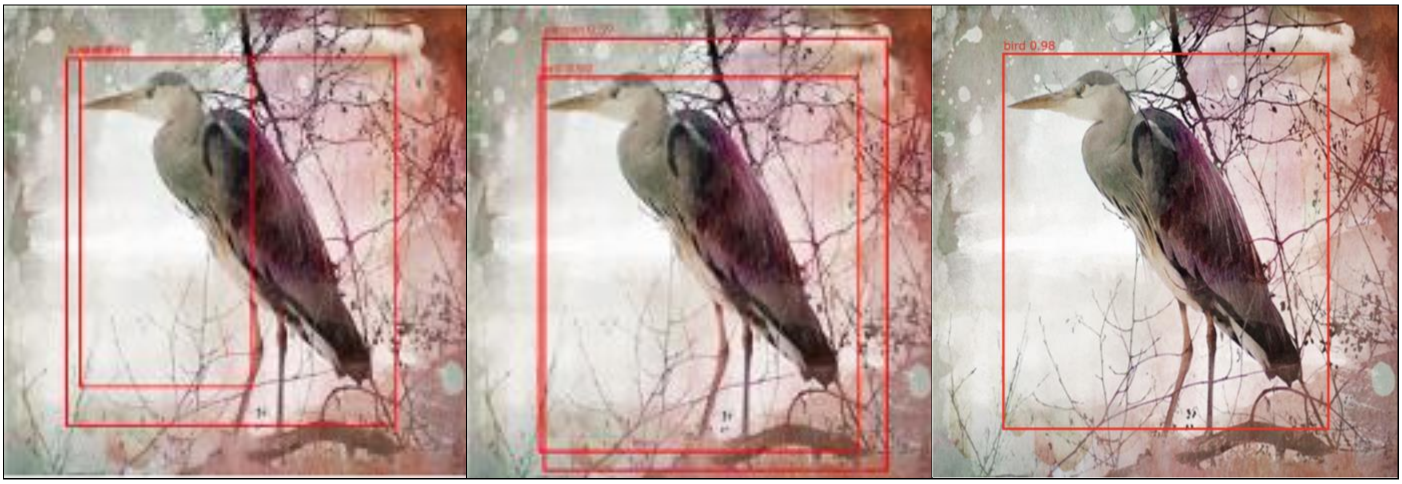


Fig.3. Predicted Detection

6. Conclusion

In this project, we explored various transfer learning strategies for adapting a COCO-pretrained Faster R-CNN model to the Watercolor2k dataset, an artistic domain with limited annotations and a significant domain shift from natural images. Through a comprehensive set of experiments and ablation studies, we draw the following key conclusions:

Effectiveness of Transfer Learning:

Even without any fine-tuning, a COCO-pretrained model showed limited ability to generalize in a zero-shot setting. However, all fine-tuning variants substantially outperformed both the zero-shot and scratch-trained baselines, reaffirming the utility of pretrained knowledge in low-resource domains.

Minimal Adaptation is Sufficient:

Surprisingly, fine-tuning only a small subset of the model—such as the detection head or BatchNorm layers—was enough to yield strong performance. Specifically, BatchNorm-only fine-tuning achieved the best results across all mAP thresholds, highlighting the importance of feature distribution alignment over complete weight updates.

Selective Fine-Tuning Outperforms Full Adaptation:

Contrary to common intuition, full fine-tuning underperformed compared to partial strategies. This suggests that updating all weights may lead to overfitting or degradation of useful pretrained representations, especially in small datasets like Watercolor2k.

Faster Convergence with Less Training:

The BatchNorm-only and head-only strategies not only reached higher final accuracy but also converged more quickly and stably. In contrast, full fine-tuning showed slower convergence and greater instability, making it less desirable in resource-constrained settings.

Future Work

This project focused on fine-tuning strategies, but future work can explore the effect of data augmentation on domain adaptation. Given the limited size of the Watercolor2k dataset, applying and comparing augmentations such as color jitter, random erasing, and CutMix could further improve generalization. A systematic ablation on augmentation combinations may reveal more effective training pipelines for stylized domains.

Reference

1. Yosinski, J., Clune, J., Bengio, Y., & Lipson, H. (2014). *How transferable are features in deep neural networks?* In *Advances in Neural Information Processing Systems (NeurIPS)*. https://arxiv.org/abs/1411.1792
2. Ren, S., He, K., Girshick, R., & Sun, J. (2015). *Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks.* In *Advances in Neural Information Processing Systems (NeurIPS)*. https://arxiv.org/abs/1506.01497
3. Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., ... & Zitnick, C. L. (2014). *Microsoft COCO: Common Objects in Context.* In *European Conference on Computer Vision (ECCV)*. https://arxiv.org/abs/1405.0312
4. Everingham, M., Van Gool, L., Williams, C. K. I., Winn, J., & Zisserman, A. (2010). *The Pascal Visual Object Classes (VOC) Challenge.* In *International Journal of Computer Vision (IJCV)*. https://link.springer.com/article/10.1007/s11263-009-0275-4
5. Inoue, N., Furuta, R., Yamasaki, T., & Aizawa, K. (2018). *Cross-Domain Weakly-Supervised Object Detection through Progressive Domain Adaptation.* In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. https://arxiv.org/abs/1803.11365
6. Li, Y., Wang, N., Shi, J., Liu, J., & Hou, X. (2016). *Revisiting Batch Normalization for Practical Domain Adaptation.* In *arXiv preprint arXiv:1603.04779*. https://arxiv.org/abs/1603.04779
7. Padilla, R., Passos, W. L., Dias, T. L., Netto, S. L., & da Silva, E. A. B. (2021). *A Comparative Analysis of Object Detection Metrics with a Companion Open-Source Toolkit.* In *Electronics*, 10(3), 279. https://doi.org/10.3390/electronics10030279