Understanding Domain Adaptation Using CORAL in Computer Vision

ABSTRACT

This paper investigates whether Domain Adaptation techniques can significantly improve the performance of Convolutional Neural Networks (CNNs) in image classification across varying domains and distributions. Our work uses Deep CORAL with EfficientNetV2 for domain adaptation using the Office-31 dataset, building off of B. Sun's, J. Feng's, and K. Saenko's paper on 'Correlation Alignment for Unsupervised Domain Adaptation' [13]. We compare its performance to a regular EfficientNetV2 model that doesn't use domain adaptation, measuring improvements with metrics as follows: accuracy, precision, recall, and F1 score. Due to Domain Adaptation allowing CNNs to recognize domain-invariant features, we hypothesize that integrating Domain Adaptation, specifically the CORAL Loss technique, will. The CORAL-implemented model demonstrated a 4.15% average boost in average precision, recall, F1, and accuracy across all 3 trials. GitHub can be accessed here.

Introduction

The significance of Domain Adaptation remains of theoretical interest. It has real-world implications that are based on varying conditions and varying images based on light, sound, saturation, etc. Domain Adaptation is beneficial to systems that involve medical imaging, autonomous work, and remote sensing, where collecting new data for every possible domain is either expensive or impractical or both. However, designing effective domain adaptation algorithms remains a challenge. Problems such as negative transfer, distribution mismatch, and a lack of labeled target data require consideration before deployment and use.

Domain Adaptation has been held as a promising technique to fix the domain shift problem by realigning the input data to the model's training data such that it will be seen as the same distribution as the training data while still preserving enough dimensions and key insights such that there will be no significant data loss in the input data.

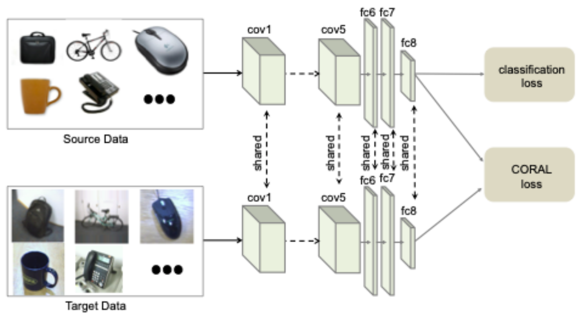

Figure 1. A diagram of a CORAL-implemented Convolutional Neural Network architecture [13].

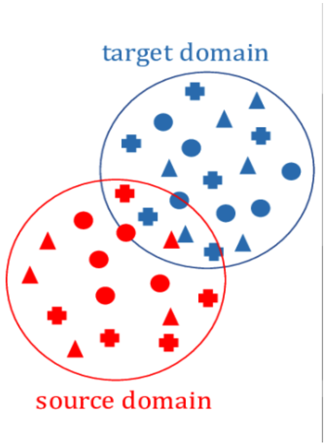

Figure 2. A diagram of domain shift [12].

The different circles represent the different domains with the shapes representing features. Domain-invariant features are represented through the shapes that exist in both domains (circles). Domains can vary from type of camera, image saturation, blur, time of day, etc used, in this case, to capture images. For example, in an object recognition task, domain-invariant features might focus on the object's shape and texture rather than lighting conditions, which may vary between source and target domains.

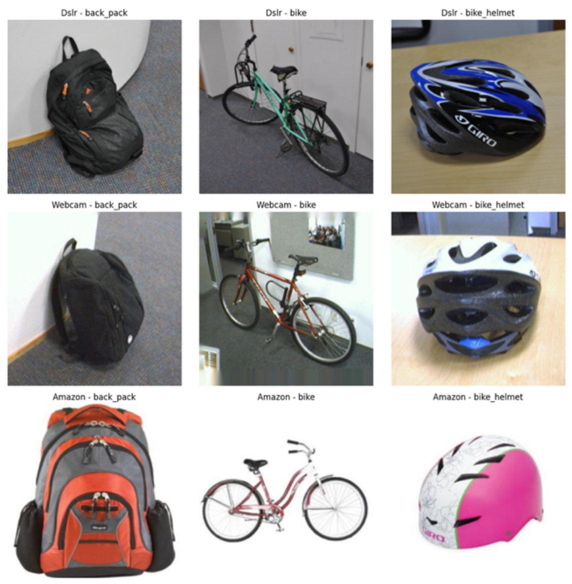

Figure 3. A sample of images from each type of domain: Amazon, Webcam, DSLR

This is a group of images extracted from the Office-31 dataset. As seen, these object images are captured from different lighting, backgrounds, and resolutions, which represent varying domains of feature representations.

Furthermore, Domain-invariant features are features that remain consistent and meaningful across different domains, allowing machine learning models to perform well even when the data distribution changes between training (source domain) and testing (target domain). Models trained on domain-invariant features can generalize better to different environments; therefore, these models can perform better on separate data distributions as long as the features are still detectable in the datasets. In general, domain-invariant features are the key insights that are extracted out of a dataset, which allows machine learning models to generalize and perform better over different domains.

Domain shift (or distributional shift) is a major problem that may negatively affect the performance of our machine learning models when we put them in production [5]. Domain shift happens when our training, validation, and test data are drawn from a probability distribution that is different from the distribution of the data on which we will use our predictive models [5]. One of the main consequences of the domain shift is that our estimates of the expected loss on the test set may be biased. While domain shift is hard to overcome, we can gauge its adverse effects on out-of-sample predictions by taking special precautions when we form our test samples. These are predictions by the model on input data that falls outside of the domain of the sample trained by the model.

Domain Adaptation offers a path forward in computer vision by enabling models to have high performance across varying domains and distributions without extensive training. By focusing on reducing domain shift, we can reduce the arising bias that comes with introducing models in real-world settings and ensure that models remain reliable and trustworthy.

Related work

As per Farahani et al, Domain Adaptation is a field of transfer learning that aims to improve the performance of a target model over insufficient or a lack of annotated data. Transfer learning refers to a class of machine learning problems where either the tasks and/or domains may change between source and target, while in domain adaptations, only domains differ, and tasks remain unchanged [2].

Consider a scenario where a model is being developed to classify road signs. The labeled training data (source domain) consists of high-resolution images of European road signs captured in clear weather using DSLR cameras. The model performs well on the source domain, however, the goal is to also make the model work effectively on images captured from dashcams in the United States under different conditions, such as poor resolution and varying weather conditions. This data, one in the United States, has no labels available. In this context, Domain Adaptation becomes crucial. Although tasks remain the same, the feature distributions between the source and target domains differ significantly due to external conditions, such as camera type, quality of resolution, or regional differences. Domain Adaptation helps mitigate falling in this gap by aligning feature representations between source and target domains regardless of labels in the source data. A method like CORAL can align the statistical properties (covariances) to generalize better to the unlabeled target domain.

In contrast, generic transfer learning would involve using a pretrained model on a broad dataset, such as ImageNet, and then fine-tune it on your labeled source domain (European road signs). While this approach helps the model benefit from extracting visual features, it doesn't address the underlying domain shift between the source and target domains. Transfer learning also typically assumes some labeled data in the target domain for fine-tuning.

The key distinction is that Domain Adaptation is designed to handle domain shift between datasets sharing the same task, whereas transfer learning focuses on reusing knowledge from a related task or dataset, often requiring some level of supervision not necessarily required in Domain Adaptation.

In classification tasks, the objective is to learn a function that maps input data to labels. For instance, in image classification, a classifier assigns each image to a specific category, such as a dog or a cat. To achieve the best predictive performance, a model is typically trained on the source dataset by minimizing the expected error on the labeled source data. This is done by learning the model that minimizes the loss between the predicted and true labels in the source domain as per Farahani et al [2].

CORAL Architecture for Unsupervised Domain Adaptation

In domain adaptation, domains can be considered as an object consisting of three main parts: input or feature space represented as X, output or label space Y, and, which is joined with the probability distribution of $p(x, y)$, creating a domain $D = \{X, Y, p(x, y)\}$. Y refers to either the binary or multi-class spaces of {-1,1} or {1,…K} where K is the number of classes), [2].

Unsupervised Domain Adaptation (UDA) focuses on scenarios where labeled data is available only in the source domain, while the target domain lacks labels completely. The goal is to train a model using the labeled source data that can generalize well to the target domain. This is challenging due to the distribution shift between the source and target domains. Domain adaptation aims to build a classifier that can handle the shift in data distribution between the source and target domains [2].

To address this challenge, Domain adaptation (DA) techniques aim to bridge the gap between source and target domains. This is achieved through aligning feature distributions among different data domains to create effective, adaptive models. This technique is particularly useful when there are differences in data characteristics or when dealing with constraints on labels in data. DA techniques, such as CORAL, help bridge domain shifts by minimizing domain discrepancies by aligning source and target features [2].

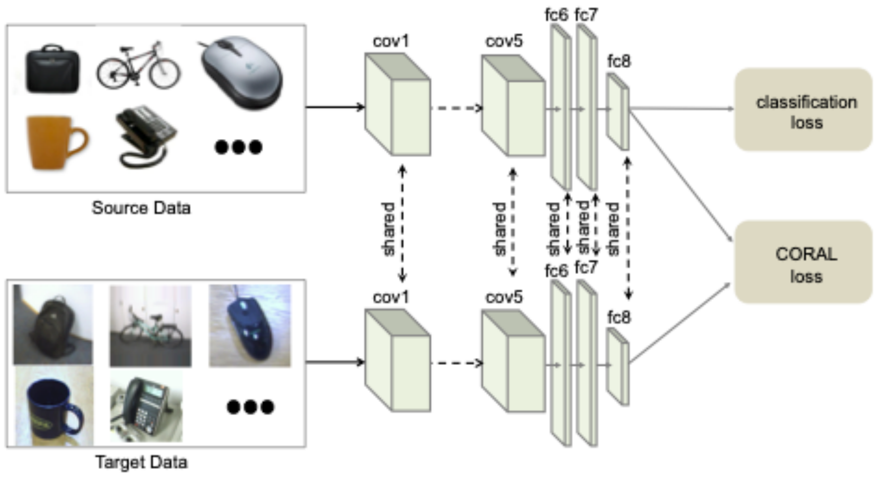

Figure 4. CORAL model architecture [13]

For an Unsupervised Domain Adaptation model, where the target data is unlabeled, the CORAL Loss model is designed to address the challenge of Unsupervised Domain Adaptation by aligning the covariance matrices of the source and target features [4]. In the architecture seen in Figure 4, source and target data are passed as inputs through the same convolutional neural network. Although the goal is to reduce, in this case, the Cross-Entropy Loss and increase accuracy, implementing CORAL Loss adds the second goal of training the feature extractors (Convolutional,Max Pooling, etc.) to align the same object, in different domains, to be recognized as similar feature maps alongside classification loss.

The classification loss, as well as CORAL Loss, is extracted from the model and backpropagated as such. The input to the classification loss remains the same. However, CORAL Loss is extracted from the outputs passed after the final feature extraction layer, when both the source and target data are passed into the model, with the covariance matrices being calculated before being passed into the CORAL Loss function. The backpropagation algorithm with respect to CORAL Loss, determining the covariance matrix, and calculating the CORAL Loss are described below.

CORAL Loss

Suppose we are given source training batches of $D_S = \{x_i\}$ such that $x \in R^d$ with labels $L_S = \{y_i\}$, where $i \in 1, \dots, L$ and unlabeled target data $D_T = \{u_i\}$ $u \in R^d$. Assume the number of source and target data is n_s and n_t, respectively. In this case, both **x** and **u** are the specific d-dimensional deep layer activation function $\sigma(I)$ inputs labeled I that we are trying to tune. C_S and C_T are the respective second-order (covariance) matrices for the source and target data for the features, as per Sun and Saenko [4]. Covariance matrices represent how each feature varies with other features in a dataset, hence their correlation.

The covariance matrices are given by:

$$C_S = \frac{1}{n_S - 1}(D_S^T D - \frac{1}{n_s}(\mathbf{1}^T D_S)^T (1^T D_S))$$
$$C_T = \frac{1}{n_T - 1}(D_T^T D - \frac{1}{n_s}(\mathbf{1}^T D_T)^T (1^T D_T))$$

Simply, CORAL Loss is the distance between these two matrices as given below:

$$l_{CORAL} = \frac{1}{4d^2}\left|\left|C_S - C_T\right|\right|_F^2$$

Where $||\cdot||_F^2$ represents the Frobenius Norm for square matrices. The loss is then managed like other native evaluations such as MSE, CE; the loss is calculated per batch and averaged over training steps.

The Frobenius norm quantifies the distance between the source and target covariance matrices in high-dimensional space. This distance represents the domain discrepancy, and the neural network is trained to minimize this discrepancy, thereby reducing domain shift and improving generalization across different domains [4]. The model is trained to recognize the domain-invariant features by aligning the covariance matrices to the same domain.

The gradient, with respect to the input features, can overall be back propagated by such:

$$\frac{\partial l_{CORAL}}{\partial D_S^{ij}} = \frac{1}{d^2(n_s - 1)}\left(\left(D_S^T - \frac{1}{n_s}(\mathbf{1}^T D_S)^T \mathbf{1}^T\right)^T (C_S - C_T)\right)^{ij}$$

Note that minimizing this loss can lead to overfitting to the source domain, resulting in reduced performance on the target domain. Just reducing CORAL Loss alone may degenerate some features. The network may project the Sources and targets to a single point, and the loss approaches 0 [4]. Secondly, CORAL Loss and classification loss are designed to be utilized simultaneously to address both the loss' limitations. Classification loss does not account for the domain discrepancy that CORAL Loss measures. This is where weighted CORAL Losses arise.

$$l_{TOTAL} = l_{CLASS} + \sum_{i=1}^{t} \lambda_i \, l_{CORAL}$$

Where t is the number of CORAL Loss layers and λ is a weight (discount) factor that trades off adaptation and accuracy to converge to the lowest loss [13].

When integrating different Domain Adaptation techniques in models, it is essential to recognize the various types of domain shifts in order to understand what techniques are required to mitigate the effects of domain discrepancy. Below are common distribution shifts that describe various domain discrepancy issues in real-world data.

Conditional Distribution Shift

This occurs when the marginal distribution of the input features changes between the source and target domains, but the conditional distribution of the labels given the inputs remains the same. In other words, while the underlying relationship between the features and labels remains stable, the input features themselves have shifted. For instance, in an object recognition task, if a model is trained on images of objects taken under one set of lighting conditions and then applied to images taken under different lighting, a covariate shift occurs. The model may struggle because the features it learned during training are no longer sufficient to generalize to the new conditions.

Let $p(y_t, x_t)$ be a joint probability distribution such that is the output $y_t$it be output and x_t is the input which will be extracted from our model. Domain shift happens when the training, validation, and/or testing isn't drawn from the joint probability distribution but from a conditional probability: $p(z_t \in U)$

Where z_t is a random latent variable; note that z_t it is dependent on y_t, x_t and U is a proper subset.

Covariate Shift

Covariate shift refers to a situation in machine learning where the distribution of the input data (features) changes between the training and testing phases, while the relationship between the input and output (the conditional distribution of the output given the input) remains the same. This shift can lead to degraded performance because the model was trained on data that doesn't fully represent the distribution it encounters during testing or real-world deployment [4].

Let $p(X_{train})$ represent the data distribution of the input data applied during the training phase. Let $p(X_{test})$ represent the data distribution during the testing phase:

In the current problem, we assume that $p(X_{train}) \neq p(X_{test})$, however $p(Y) = p(X)$ where X is all input data and Y represents output labels [4]. Hence, $p(Y|X)$, represents the relationship between the input features and output labels remain constant across the source and target domains.

If the model is trained on a specific data distribution, there may be bias to a certain distribution and/or certain patterns that are not the domain-invariant features. To represent this distribution shift, metrics such as the Kolmogorov-Smirnov Test and the Jensen-Shannon Divergence are used [1].

Concept Shift

Concept shift refers to the case where the dependence of the label upon the features differs between the target and the source domains, often depending on time, in which case it is termed a concept drift. The distribution of the features is nevertheless assumed to be the same in both domains as per Lemberger and Panico [3].

Subspace Mapping

Subspace mapping describes a situation where observations are distributed alike as physical objects in the source and target domains, but where the features used to describe them in one or the other are different and related by an unknown change of coordinates. For example, an object seen from different angles, as per Lemberger and Panico [3].

Experimental Setup

One of the most widely used datasets for studying domain adaptation is the **Office-31 dataset** [11]. This dataset includes 4,110 images across 31 object categories captured in three distinct domains: **DSLR**, **Webcam**, and **Amazon**. These domains represent different imaging conditions: DSLR images are of high quality and resolution, whereas Webcam images are grainy and noisy. The Amazon domain consists of product images downloaded from the e-commerce platform, often featuring different backgrounds and lighting conditions. As mentioned earlier, the domain shift represented in this dataset is covariate shift as the source input distribution does not equal the target distribution, however, the relationship between the input and labels stay constant, represented as $p(Y|X)$The overall distribution of this dataset makes Office-31 a powerful dataset to test model robustness.

Domain Shift

In a Domain Adaptation, it's crucial to measure how different the source and target data distributions are, since the primary challenge lies in training a model on one domain and ensuring it performs well on another with a potentially different distribution.

This is where Kolmogorov-Smirnov (KS), and Jensen-Shannon (JS) Divergence become valuable tools [1]. The KS test is useful for statistically testing whether the source and target distributions (e.g., pixel intensities or individual feature values) come from the same population, offering a formal hypothesis test with p-values [1].

JS Divergence provides a smooth, symmetric measure to quantify the distance between two probability distributions, making it particularly effective for comparing SoftMax outputs or high-dimensional learned feature embeddings.

KS Test answers the question of whether two dataset distributions (in our case, the webcam and DSLR) are drawn from the same data distribution by measuring the maximum vertical distance. If the distance is small, the distributions are similar; if it is large, the distributions are different. The KS statistic measures the maximum distance between two cumulative distribution functions (CDFs). KS plots the CDF of each dataset and finds the biggest gap between them.

JS Divergence, however, focuses on probability distributions, specifically the average of the two distributions, and measures how each differs. The JSD measures the similarity between two probability distributions (P) and (Q)

We determine the results of JSD and KS

Distribution Shift from DSLR to Webcam:

- Jensen-Shannon Divergence: 0.1669
- KS Test
    1. Statistic = 0.6670
    2. P-value = 0.0000

The results indicate a clear distribution shift between the DSLR and Webcam datasets. A Jensen-Shannon Divergence of 0.1669 suggests a moderate difference in the average pixel distributions, reflecting changes in lighting, contrast, or color profiles. The Kolmogorov-Smirnov test further supports this, with a high statistic of 0.6670 and a p-value of 0.0000, confirming a statistically significant difference in image-level intensity distributions. Together, these metrics highlight the need for domain adaptation techniques to address the shift when training models across these datasets.

Objectives of the Experiment

This experiment investigates the value of Unsupervised Domain Adaptation to improve the generalization of image recognition models to capture domain-invariant features and to recognize and classify images accurately in different environments outside of the training dataset. We will apply the supervised Deep CORAL domain adaptation technique using the CORAL Loss function. Note that, for efficiency, we will be using the first 5 classes instead of 31. Unlike B. Sun's, J. Feng's, and K. Saenko's CORAL implementation with AlexNet [13], this experiment will use EfficientNet version 2 (V2). EfficientNetV2, a deep convolutional neural network, is widely known for its robust SOTA performance, known for its balance between efficiency and accuracy, resulting in models that train faster and are significantly smaller than other models, hence, it was picked to reduce computational workload and time required between experiments. This experiment is designed to test two phases:

Control Experiment: EfficientNetV2 Without Domain Adaptation

Experimental Setup

- Batch Size: 64
- Epochs: 20
- Optimizer: Adam
- Loss Function: Cross-entropy loss for classification
- Validation Split: 20% - The split is a randomized sample of 20% of the training data used to mitigate the effects of bias
- Metrics: Accuracy, Precision, Recall, F1 Score

The first phase of the experiment involves training a regular EfficientNetV2 model without domain adaptation. In this experiment, the EfficientNetV2 model is trained on the **DSLR domain**, which consists of high-quality images. During each trial of training EfficientNet, the model is evaluated on the target domain, Webcam. The Webcam images are lower in quality and noisier compared to DSLR images, representing a distribution shift.

Domain Adaptation Experiment: EfficientNetV2 with Deep CORAL

Experimental Setup

- Batch Size: 64
- Epochs: 20
- Optimizer: Adam
- Loss Function: Cross-entropy loss for classification, CORAL loss for domain feature alignment
- Validation Split: 20%
- Metrics: Accuracy, Precision, Recall, F1 Score

We use the following formula for calculating total loss of the model:

$$l_{TOTAL} = l_{CLASS} + \sum_{i=1}^{t} \lambda_i \, l_{CORAL}$$

Where, in this experiment, the λ_i is set to 1 to provide a fair balance between Cross-Entropy Loss and Coral Loss.

Results

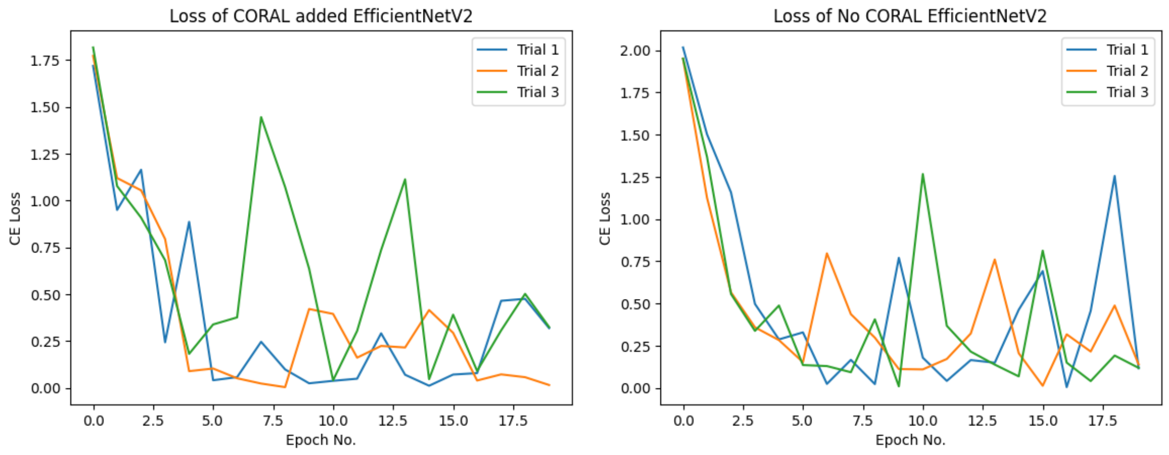

Figure 5. The cross-entropy loss of CORAL-added and No CORAL EfficientNetV2 over epochs across 3 trials

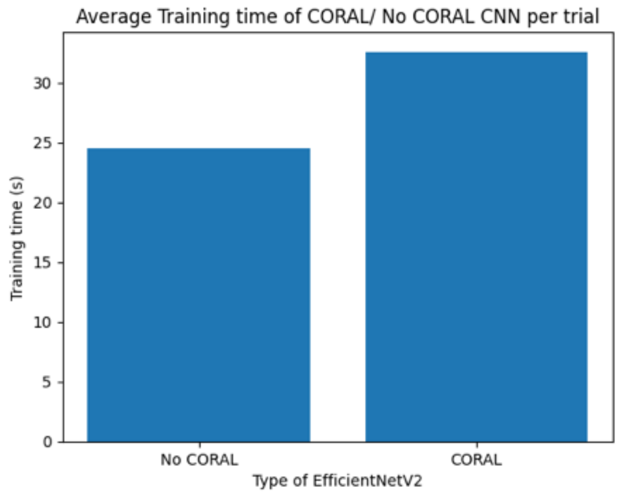

Figure 6. The average training time per trial of CORAL and No CORAL EfficientNetV2

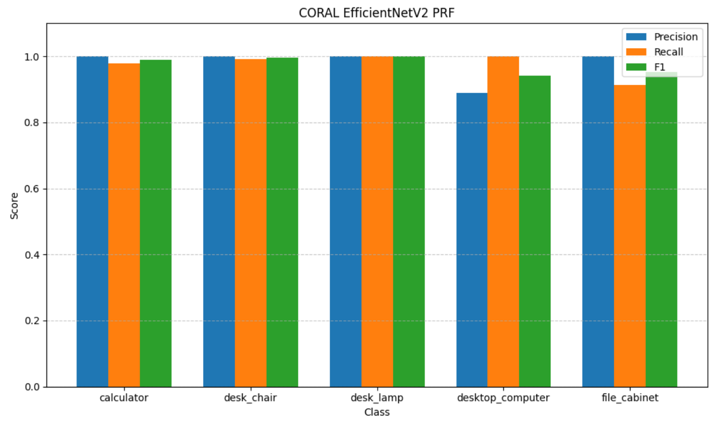

Figure 7: Bar Graph of Precision, Recall, and F1 per class for CORAL EfficientNetV2

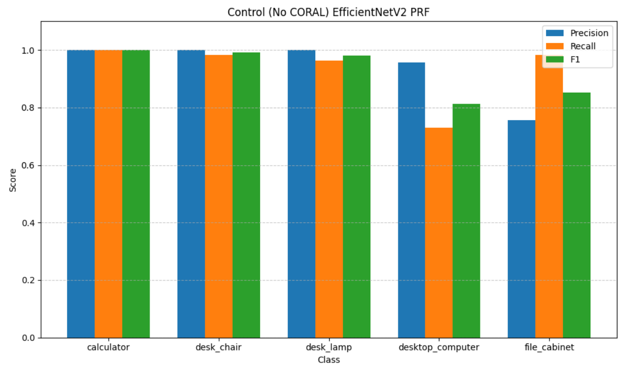

Figure 8: Bar Graph of Precision, Recall, and F1 per class for Control EfficientNetV2

CORAL	calculator	desk_chair	desk_lamp	desktop_computer	file_cabinet
Precision	1.0	1.0	1.0	0.8898	1.0
Recall	0.9785	0.9917	1.0	1.0	0.9123
F1	0.9889	0.9958	1.0	0.9410	0.9518
Control					
Precision	1.0	1.0	1.0	0.9565	0.7555
Recall	1.0	0.9833	0.9630	0.7302	0.9825
F1	1.0	0.9915	0.9804	0.8121	0.8510

Figure 9. Comparison of Baseline and CORAL added EfficientNetV2 average Accuracy, and average Precision, Recall, and F1 per class over trials

Model	Accuracy	Precision	Recall	F1
Baseline (No Coral)	0.9432	0.9424	0.9318	0.9270
CORAL	0.9793	0.9780	0.9765	0.9755

Figure 10. Comparison of Baseline and CORAL added EfficientNetV2 Accuracy, and average Precision, Recall, F1

Discussion

The use of CORAL in EfficientNetV2 has resulted in significant performance improvements in terms of a variety of classification metrics when using Domain Adaptation. As seen from Figures 2,3, and 4, the CORAL-implemented EfficientNetV2 generally performed better than the baseline model (non-CORAL) on the target sample. This verifies the hypothesis that aligning the source and target features using CORAL makes the model more generalizable in domain discrepancy scenarios.

As seen in Fig. 5, , the CORAL-implemented model demonstrated a 4.15% average boost in precision, recall, F1, and accuracy against the metrics. While this margin of improvement in absolute terms appears small, relative to domain adaptation work, marginal values are likely to approach better real-world standards in deployment. Additionally, this gain held across most categories and improved the F1 values of the desktop_computer and file_cabinet classes, suggesting broad advantages.

One of the most vivid results is obtained by evaluating the model on poorly performed classes, which, in this case, are desktop_computer and file_cabinet. These classes had previously given abnormally low precision, recall, and F1 scores, especially compared to the rest of the metrics across the classes. With CORAL implementation in the training procedure, both these classes saw a boost in their F1 scores, which insightfully shows that CORAL enhances the model's capability for learning generalized features that are consistent across multiple domains, even for classes where capturing these features towards a low loss might be difficult, and may have been overcomplicated by CORAL-implementation. This acts to further strengthen CORAL as an effective tool in the case where specific classes are adversely affected by domain shift.

Training dynamics further suggests the effectiveness of CORAL. Figure 1 shows that the categorical cross-entropy loss after training was 14% smaller in the case of CORAL implementation in comparison to the baseline. This suggests better-calibrated and confident predictions, which is a quality required in models that are being deployed and exposed to real-world contexts. There are also less significant jumps in loss for the CORAL-added model. Moreover, a lower loss indicates improvements in convergence, reinforcing the fact that CORAL provides a smooth optimization path by reducing the distance/discrepancy between source and target features.

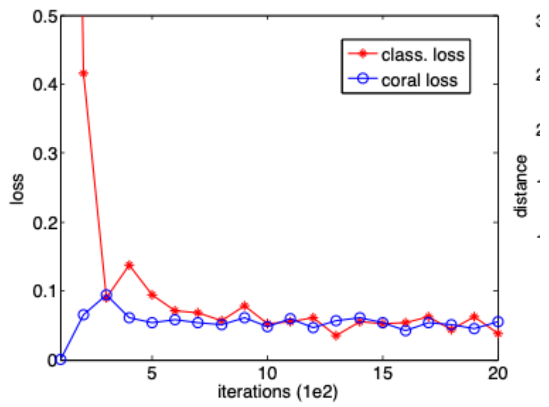

Figure 11. Classification loss vs CORAL Loss in minimizing domain discrepancy

Training dynamics were similar to those of B. Sun and K. Saenko [13]. The CORAL may not have resulted in a lesser distance (Frobenius Norm) between the source and target samples, the CORAL Loss suffered les abrupt fluctuations than the classification loss.

This improved result trades off with computing time. Incorporating CORAL resulted in additional computational cost and increased training time by approximately 29.55% across all 3 trials. This is because of additional matrix operations to calculate second-order statistics alignment across domains. While this increases the model's complexity, the performance boost may be worthwhile in situations where accuracy and dependability in the target domain are critical.

Limitations

Integration of CORAL into EfficientNetV2 has resulted in notable improvements in domain adaptation performance, however, several limitations exist that challenge consideration.

The incorporation of CORAL introduces an additional computational workload. The alignment of covariance, second-order statistics between the source and target samples requires extra matrix operations, especially on high-dimensional features, leading to a 29.55% increase in training time. This computational demand may uncover challenges for deployment while still factoring in resource usage and management, and resource-constrained areas or applications that require real-time processing.

If the CORAL loss is emphasized during training, there is a potential risk of the network learning degenerate features. For example, minimizing CORAL loss alone might lead the model to project both source and target to a point in a multidimensional space, resulting in alignment that yields a lack of discriminative power [4].

The effectiveness of CORAL is dependent upon the quality of the extracted features by the CNN. Utilizing pre-trained architectures that may not capture domain-invariant features, such as AlexNet, can limit the alignment potency [6]. Furthermore, joint and concurrent training of CORAL and a classifier may not efficiently align features if the features themselves are not satisfactory.

CORAL operates under the assumption that the solution to the domain shift is by aligning only second-order statistics between domains. This may not hold in scenarios where greater-order statistical differences exist that limit CORAL's ability to adapt to complex, real-world scenarios [6].

In summary, while CORAL enhances domain adaptation capabilities, its limitations, ranging from computational demands to statistical misrepresentation and non-feasibility, should be carefully considered when deploying in real-world settings.

Future Work

Building off of CORAL, several open challenges and promising paths remain. Future research needs to be directed at constructing unified frameworks that can address multi-source and multi-target paradigms. Currently, the majority of methods are only for single-source and single-target cases, whereas real-world tasks often involve multiple sets of uniformly distributed domains. Designing algorithms that can effectively leverage multiple sources of information and adapt to a variety of target conditions is nontrivial but a significant step towards real-world deployment.

One direction is to explore computationally light variations or hybrid methods that maintain CORAL's alignment capability without significantly increasing training time. Because CORAL is seeing a 29.55% increase in training time due to second-order matrix computation, optimizing CORAL's implementation or integrating light-weight domain alignment techniques might make it more applicable in real-time or resource-constrained scenarios.

Further, domain robustness and fairness are key areas of future research for model robustness. Domain shifts are likely to exaggerate fluctuations in performance, especially when models operate in safety-critical and socially sensitive domains. Future work needs to focus on the development of adaptation techniques that are not only accurate but also invariant to adversarial forces and are robust against dataset bias, especially. Recent studies have explored adversarial approaches to mitigate such biases and enhance model resilience [7], [8]. Moreover, transparency and fairness in adapted models are crucial, particularly in applications ranging from healthcare to surveillance and autonomous systems (self-driving cars).

Lastly, understanding the decision-making process and the gears that turn behind domain adaptation models is crucial for accountability. Explainable AI (XAI) models, such as LIME and SHAP, have been utilized to uncover model predictions. Integrating XAI into domain adaptation systems can aid in recognizing model behavior and ensuring utmost transparency [9], [10].

Addressing these challenges: enhancing robustness and fairness, improving sim-to-real transfer and hosting interpretability, will become critical for the advancement and deployment of domain adaptation in computer vision.

Conclusion

This study highlights the importance of Domain Adaptation in mitigating the effects of domain shift in computer vision. By utilizing CORAL with EfficientNetV2, we were able to align the feature distributions of the source (DSLR) and. Target (Webcam) domains, resulting in significantly improved performance on target domain classification tasks. Compared to the baseline model of EfficientNetV2, the CORAL-enhanced model consistently improved the performance over varying domains across all evaluated classes.

The Jensen-Shannon Divergence (JSD) and the Kolmogorov-Smirnov Test (KS Test) confirmed a detectable distributional shift between the DSLR and Webcam domains, restating the need for an adaptation technique that centralizes all training data to the model. Deep CORAL effectively minimized the discrepancy by aligning the covariance matrices, allowing the model to learn domain-invariant features quicker than a non-CORAL-enhanced model.

Overall, our results effectively demonstrate that domain adaptation techniques such as the investigated CORAL are not only theoretically significant but also useful in practical terms. In real-world scenarios, domain adaptation enables models to generalize better across varying conditions, recognizing domain-invariant features regardless of external factors and stimuli.

Acknowledgements

I would like to thank my mentor, Dr. Nirmala Ramakrishnan, for her valuable guidance and feedback on my drafts and roadmap to writing this paper. Dr. Ramakrishnan's support and mentorship were instrumental in helping me complete this work.

References

[1] "Beware the Kolmogorov-Smirnov test! – Astrostatistics and Astroinformatics Portal," *Psu.edu*, 2019. https://asaip.psu.edu/articles/beware-the-kolmogorov-smirnov-test/ (accessed May 08, 2025).

[2] A. Farahani, S. Voghoei, K. Rasheed, and H. R. Arabnia, "A Brief Review of Domain Adaptation," *arXiv preprint arXiv:2010.03978*, Oct. 2020, doi: 10.48550/arXiv.2010.03978.

[3] P. Lemberger and I. Panico, "A Primer on Domain Adaptation," *arXiv preprint arXiv:2001.09994*, Jan. 2020, doi: 10.48550/arXiv.2001.09994.

[4] B. Sun and K. Saenko, "Deep CORAL: Correlation Alignment for Deep Domain Adaptation," in *European Conference on Computer Vision (ECCV)*, 2016, pp. 443–450, doi: 10.1007/978-3-319-49409-8_35.

[5] Taboga, Marco (2021). "Domain shift", Lectures on machine learning. https://www.statlect.com/machine-learning/domain-shift.

[6] Z.-Y. Wang and D.-K. Kang, "P-Norm Attention Deep CORAL: Extending Correlation Alignment Using Attention and the P-Norm Loss Function," *Applied Sciences*, vol. 11, no. 11, p. 5267, Jun. 2021, doi: 10.3390/app11115267.

[7] E. Tzeng, J. Hoffman, K. Saenko, and T. Darrell, "Adversarial Discriminative Domain Adaptation," *arXiv preprint arXiv:1702.05464*, Feb. 2017, doi: 10.48550/arXiv.1702.05464.

[8] J. Huang, D. Guan, A. Xiao, and S. Lu, "RDA: Robust Domain Adaptation via Fourier Adversarial Attacking," *arXiv preprint arXiv:2106.02874*, Jun. 2021, doi: 10.48550/arXiv.2106.02874.

[9] M. T. Ribeiro, S. Singh, and C. Guestrin, "'Why Should I Trust You?': Explaining the Predictions of Any Classifier," in *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Demonstrations*, San Diego, CA, USA, Jun. 2016, pp. 97–101, doi: 10.18653/v1/N16-3020.

[10] "IIPSeries - Conferences & Edited Books," *Iipseries.org*, 2024. https://www.iipseries.org/searching.php?type=title (accessed May 18, 2025).

[11] X. Hu, "Office31," Kaggle, 2022. [Online].Available: **https://www.kaggle.com/datasets/xixuhu/office31**

[12] S. Zhao and H. Lang, "Improving Deep Subdomain Adaptation by Dual-Branch Network Embedding Attention Module for SAR Ship Classification," in IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing, vol. 15, pp. 8038-8048, 2022, doi: 10.1109/JSTARS.2022.3206753.
keywords: {Feature extraction;Marine vehicles;Synthetic aperture radar;Transfer learning;Measurement;Data mining;Semantics;Deep subdomain adaptation network (DSAN);domain adaptation (DA);dual-branch network (DBN);ship classification;synthetic aperture radar (SAR);transfer learning (TL)}

[13] B. Sun, J. Feng, K. Saenko, "Deep CORAL: Correlation Alignment for Deep Domain Adaptation," *arXiv preprint arXiv:1612.01939*, Dec. 2016. [Online]. Available: https://arxiv.org/abs/1612.01939