Domain Adaptation Using CORAL in Computer Vision

Dr. Nirmala Ramakrishnan

Aditya Chakraborty

**Abstract**

Computer Vision has been an active area of research for a long time. Over the past decade, Deep Learning has really pushed the field forward, with Convolutional Neural Networks (CNNs) excelling in tasks like image segmentation, object detection, and classification. But one big challenge remains—getting these models to work just as well across different environments. This problem, known as domain shift or domain discrepancy, occurs when the data the model was trained on (the source domain) is different from the data it's tested on (the target domain). These differences in data can seriously damage the model’s performance, especially in real-world scenarios where data often varies across domains.

To tackle this problem, Domain Adaptation techniques are designed to bridge the gap between different distributions by helping models identify domain-invariant features. These features exist across all data distributions of the object the model has to learn to detect. In our work, we evaluate Deep CORAL with ResNet18 for domain adaptation using the Office-31 dataset. We compare its performance to a regular ResNet18 model that doesn’t use domain adaptation, measuring improvements with metrics such as accuracy, precision, recall, and F1 score. Our results will determine if Domain Adaptation is necessary to improve model performance or if only model transfer learning is required.

**Introduction**

This paper will determine if Domain Adaptation is necessary to improve model performance or if only model transfer learning is required.

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. This process is achieved by learning **domain-invariant features**. Domain-invariant features are the 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). 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. 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 of a dataset, which allows machine learning models to learn better over different domains.

**Problem**

*Taboga,*

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.

Domain shift happens when our [training, validation, and test](https://www.statlect.com/machine-learning/training-validation-and-test-samples) data are drawn from a [probability distribution](https://www.statlect.com/probability-distributions/) that is different from the distribution of the data on which we will use our [predictive models](https://www.statlect.com/machine-learning/predictive-model).

Several problems well-known in machine learning and statistics (e.g., dependence structures, randomization bias, structural breaks, non-representative samples) can be seen as special cases of domain shift.

One of the main consequences of domain shift is that our estimates of the expected loss on the test set may be [biased](https://www.statlect.com/glossary/unbiased-estimator).

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.

We will explain and describe four types of domain shifts: conditional distribution shift and covariate shift, concept shift and subspace mapping. We will tackle covariate shifts and use the Deep CORAL Loss algorithm for feature alignment, which will be covered.

**Related work**

*Saenko et al.,*

Suppose we are given source training examples of such that with labels where and unlabeled target data , . Assume the number of source and target data are and respectively. In this case, both **x** and **u** are the specific d-dimensional deep layer activation function of of input labeled I that we are trying to tune. and are the respective second-order (covariance) matrices for the source and target data for the features.

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

Where represents the Frobenius Norm for squared matrices.

The gradient, with respect to the input features, can overall be backpropagated by the loss:

Note that minimizing this loss can lead to overfitting to the source domain result 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 becomes zero.

Where t is the number of CORAL Loss layers and is a weight (discount) that trades off adaptation and accuracy to converge to an equilibrium.

*Farahani et al.,*

Domain Adaptation is a field of transfer learning that aims to improve the performance of a target model over insufficient or 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.’

In domain adaptation, domains can be considered as three main parts: input or feature space X, output or label space Y, which is joined with the probability distribution of , therefore . Y refers to either the binary or multi-class spaces of {-1,1} or {1,…K} (K is the number of classes), respectively.

In unsupervised domain adaptation (UDA), the goal is to train a model that performs effectively on a target domain where labeled data is not available. The source domain provides labeled data in the form of feature-label pairs. Domain adaptation addresses this issue by reducing the differences between domains and training a model that can generalize well to the target data. In essence, domain adaptation aims to build a classifier that can handle the shift in data distribution between the source and target domains.

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 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.

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 of labels in data. It is also very vital in consistent domain shifts over time, such as changes in a stock market such as Bitcoin or any other financial market. DA techniques, such as domain-invariant feature learning, help bridge domain shifts by minimizing domain discrepancies.

## **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 be a joint probability distribution such that output is output and is 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:

Where is a random latent variable, note that is dependent on and 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.

Let represent the data distribution of the input data applied during the training phase. Let represent the data distribution during the testing phase:

In the current problem, we assume that , however where X is all input data and Y is output labels.

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 Kolmogorov-Smirnov Test and Jensen-Shannon Divergence are used.

*Lemberger et al.*

## **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.

## **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

**Methodology**

One of the most widely used datasets for studying domain adaptation is the **Office-31 dataset**. 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. The overall distribution of this dataset makes Office-31 a powerful dataset to test model robustness.

**Objectives of the Experiment**

This experiment investigates the value of 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 model. 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 ten classes instead of 31.

The experiment is designed to test two phases

## **Control Experiment: ResNet18 Without Domain Adaptation**

**Experimental Setup**

* **Batch Size**: 64
* **Epochs**: 10
* **Optimizer**: Adam
* **Loss Function**: Cross-entropy loss for classification, CORAL loss for domain feature alignment
* **Validation Split**: 20%
* **Metrics**: Accuracy, Precision, Recall, F1 Score

The first phase of the experiment involves training a regular ResNet18 model without domain adaptation. ResNet18, a deep convolutional neural network, is widely known for its robust SOTA performance with 50 layers and it utilizes residual connection to combat the vanishing gradient problem.

In this experiment, the ResNet18 model is trained on the **DSLR domain**, which consists of high-quality images. After training, the model's performance is evaluated on the **Webcam domain**, which serves as the target domain. The Webcam images are lower in quality and more noisy compared to DSLR images, representing a distribution shift.

## **Domain Adaptation Experiment: ResNet18 with Deep CORAL**

This experiment will answer the following objective and will be set up such.

Objective:

* Is Domain Adaptation required to create accurate models?

**Experimental Setup**

* **Batch Size**: 32
* **Epochs**: 10
* **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 also apply two tests to assess the **distribution shift** between the source and target domains:

1. **Kolmogorov-Smirnov (KS) Test**
2. **Jensen-Shannon Divergence**

**Experiments Results**

# Control experiment

A graph of different colored bars

Description automatically generated

*Figure 1*

What was astounding was that the ResNet18 model, without the CORAL Loss function, was able to achieve perfect accuracy from being trained on the DSLR domain to be evaluated over the webcam domain. This states that the model was able to capture and recognize the **domain-invariant features** that exist over all the domain distributions.

As per Figure 1, the model has poor precision, recall, and F1 score; however, it achieved a perfect evaluation over the webcam domain. This states an uneven quantity of data per class. This, however, contradicts because the dataset has been processed to have an even quantity of images per label per class. This may be because of the ImageDataGenerator class by Tensorflow; however, this should be taken lightly.

The high accuracy of the control experiment can be anticipated from the residual, skip-connections between neurons. When adapting from one domain (e.g., DSLR) to another (e.g., Webcam), it’s essential to learn features that are not biased toward domain-specific characteristics. Residual connections allow the network to retain useful features learned in earlier layers, helping to capture domain-invariant features.

To evaluate the similarity between the **webcam** and **DSLR** domains in the **Office-31** dataset, we applied two statistical tests: the **Kolmogorov-Smirnov (KS) Test** and **Jensen-Shannon (JS) Divergence** on features extracted using **ResNet18**.

* **KS Test p-value: 0.275** suggests no significant difference between the feature distributions of the two domains, indicating that they are relatively similar in terms of statistical properties.
* **JS Divergence: 0.268** reflects the moderate divergence between the domains, meaning that while they are somewhat aligned, there is still some noticeable difference.

# **Main experiment**

The main metrics used for this research: Precision, Recall, F1, was similar to the control experiment. The accuracy of the model with CORAL loss function was 99.29%. Both models show high validation accuracy, indicating that they are effective in correctly classifying instances. However, the 99.29% validation accuracy with CORAL is slightly less than the perfect accuracy of ResNet18. This decrease in performance may be derived as a random error; there isn’t any drastic changes to the model accuracy. The fact that the metrics (precision, recall, F1) are similar between the two models suggests that the CORAL loss function does not significantly degrade the model's ability to identify true positives and true negatives compared to the control. Given that your control experiment reached perfect accuracy, it's worth investigating if that was due to specific characteristics of the dataset, overfitting, or the model architecture. If ResNet18 is consistently achieving higher accuracy, it might be the more reliable choice, unless the benefits of using the CORAL loss function outweigh the slight accuracy drop.

Both models perform well, but the ResNet18 control provides perfect accuracy. The CORAL loss function's approach can still be valuable, particularly if it enhances generalization or provides other advantages (like robustness to domain shifts). It may be worth exploring further to understand the implications of the slightly lower accuracy in different contexts.

**Conclusion**

After the experiments, it has been inferred that the ResNet18 model is already trained to detect domain-invariant features, which exist across all data distributions. Secondly, Domain Adaptation over CORAL Loss took a significant amount of time in comparison to the raw ResNet18 model. For Image Recognition, Domain Adaptation is not required, rather the best approach is to utilize transfer learning for efficient and accurate performance.

**Future Work**

**Refining Transfer Learning Techniques**

Using ResNet18 in transfer learning has been really effective for image classification by leveraging pre-trained models. But there's room to improve. Future research should focus on fine-tuning these techniques to not only boost accuracy but also reduce how much computing power is needed. For example, deciding which parts of the model to freeze and which to retrain can make a big difference in adapting to new data. We could also explore other advanced models like EfficientNet or Vision Transformers (ViTs) to uncover new strategies for making transfer learning even more effective. The key is finding a good balance between the complexity of the model and its performance, so we can get better results across a wide range of applications.

**Investigating Different Loss Functions**

ResNet18 could perform even better in domain adaptation tasks if we explore alternative loss functions beyond the usual ones. While CORAL (Correlation Alignment) is a popular choice for minimizing domain differences, trying out other approaches—like adversarial or contrastive loss functions—could improve how well the model adapts to new domains. The goal is to figure out which loss functions work best in different situations, especially when there's a big gap between the source and target data. This could lead to more accurate and reliable models, even in challenging scenarios.

**Expanding Dataset Variety**

Many studies rely on a limited number of datasets, which may not capture the full range of real-world variability. It’s important for future research to test models on a wider variety of datasets with different characteristics. For example, applying CORAL and ResNet18 to datasets like DomainNet or ImageNet-C would give a better sense of how adaptable these models are under different conditions. By testing models on more diverse datasets, we can get deeper insights into how different factors affect performance and refine domain adaptation techniques to be more widely applicable.

**Comprehensive Performance Evaluation**

While transfer learning and domain adaptation with ResNet18 look promising, it’s crucial to assess the trade-offs between model efficiency and accuracy thoroughly. In many real-world applications, we have to find a balance, especially when computing resources are limited. Future work should focus on these trade-offs by using optimization techniques like pruning or quantization, which can reduce the size of the model without hurting its performance too much. The ultimate goal is to create practical models that work well even when resources are tight, making ResNet18-based solutions more feasible for real-world use.

**Exploring Hybrid Models**

There’s a lot of potential in combining transfer learning with domain adaptation to create hybrid models that can take image recognition to the next level. These models could use ResNet18's pre-trained features along with more specialized adaptation techniques, like feature alignment or adversarial approaches. Developing hybrid models could focus on speeding up training and improving accuracy, leading to efficient solutions that stay robust even when there’s a big shift between domains. For example, blending convolutional neural networks (CNNs) with transformers could allow models to capture both detailed and broad features in images, making them faster to train and more adaptable across different datasets and domains.

# Citations

# Bibliography

Babu, G. J., & Feigelson, E. (n.d.). *Beware the Kolmogorov-Smirnov test!* Retrieved from Astrostatistics and Astroinformatics Portal: https://asaip.psu.edu/articles/beware-the-kolmogorov-smirnov-test/

Farahani, A., Voghoei, S., Rasheed, K., & Arabnia, H. R. (2020). *A BRIEF REVIEW OF DOMAIN ADAPTATION.* Athens, Georgia: ArXiv.

Lemberger, P., & Panico, I. (2020). *A Primer on Domain Adaptation.* Paris: Group One Point.

Sun, B., & Saenko, K. (n/a). *Deep CORAL: Correlation Alignment for Deep Domain Adaptation.* Lowell, Massachusetts: University of Massachusetts Lowell, Boston University.

*CHATGPT WAS USED FOR ASSISTANCE DURING TRIALS IN THIS RESEARCH*