Cross Domain Image Classification of Office Objects

Kenneth Lin, Henry Marquardt, Joseph Chan

November 2024

Abstract

This project explores domain adaptation for object classification using the CALTECH-10 dataset, which contains images of 10 office object classes from four domains. The goal was to train a robust classifier on the "Amazon" and "Caltech10" domains that generalizes well to the "DSLR" and "Webcam" domains. Various feature extraction methods were evaluated, including Local Binary Patterns, Gray Level Co-occurrence Matrix, color channel (RGB) features, ResNet-101 embeddings, and Bag of Visual Words using ORB keypoints. After extraction of features, we performed principal component analysis (PCA) for dimensionality reduction and then evaluated multiple combinations of features with Support Vector Machines and Random Forests models. The results showed that models incorporating ResNet features achieved the best performance, with up to 97.6% accuracy on in-domain test data and 95.1% on out-of-domain data. Feature selection was critical for maximizing model generalizability across domains. This project highlights the importance of robust feature extraction methods like ResNet embeddings in improving cross-domain classification accuracy, while also considering computational efficiency in feature selection.

1 Introduction

Our project seeks to explore the field of domain adaptation by classifying objects in the CALTECH-10 dataset [1, 2]. The CALTECH-10 dataset contains manually labeled images of 10 classes office objects from four different image domains. The "Amazon" and "Caltech10" domains contain images that were scraped from Amazon merchant listings and Google Images, respectively. The "DSLR" and "Webcam" domains contain images that were taken manually by researchers. Motivated by real-world scenarios in which engineers seek to build robust computer vision systems using photos from the Internet (of which, in practice, we have a near-infinite supply), we seek to train a robust classifier on the "Amazon" and "Caltech10" domains that remains performant on the dramatically different "DSLR" and "Webcam" domains.

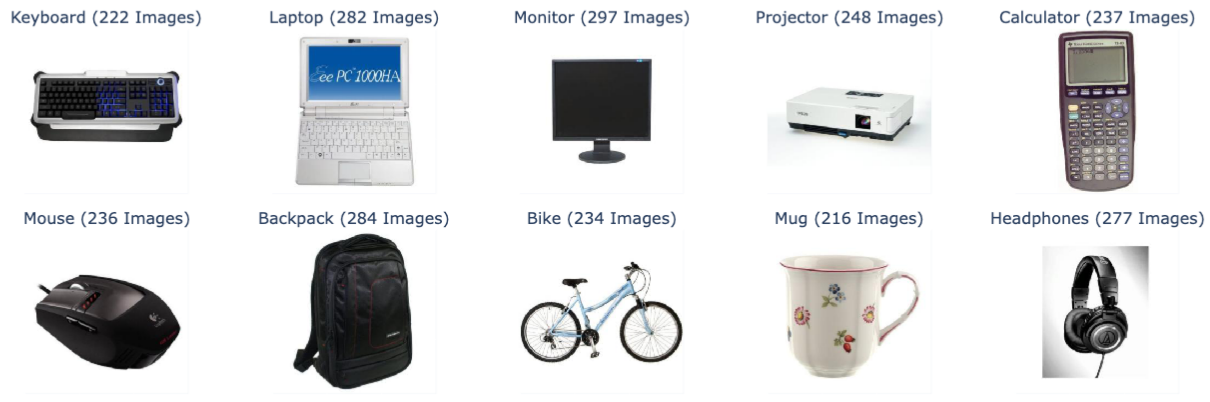

Figure 1: Sample Images from Each Label from the Amazon Domain

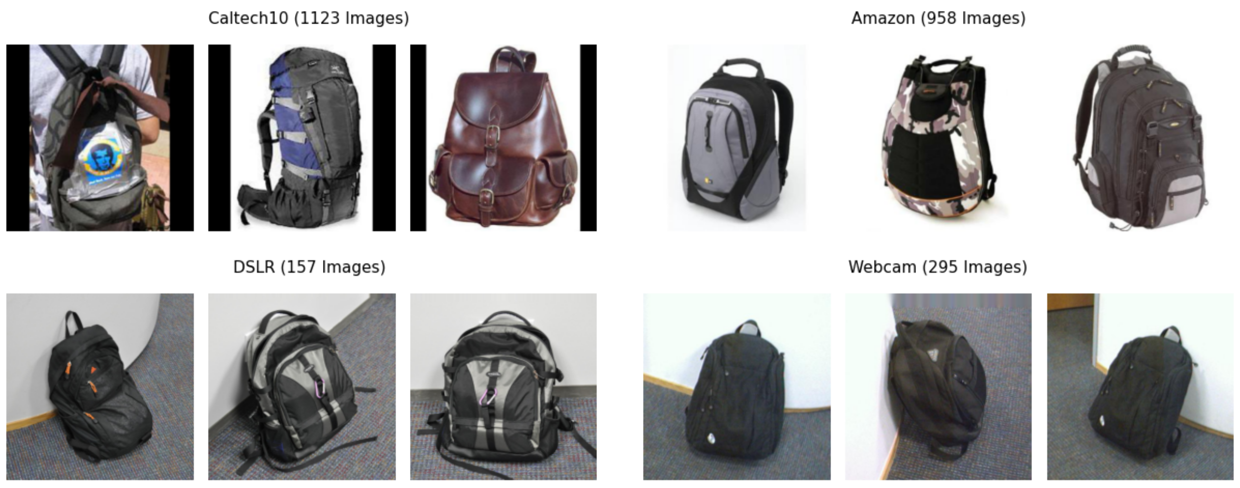

Figure 2: Sample Images from Each Domain of the Backpack Class Label

2 Feature Extraction

2.1 Image Preprocessing

Due to the methods used to collect our data, each image varies in resolution within and across domains. To resolve this issue, we padded or center-cropped all images to a standardized size of 300px $\times$ 300px. In addition, although the majority of images were represented using RGB color channels, a minority were strictly in grayscale. When computing GLCM, HoG, and BoVW features we converted all images to grayscale. By contrast, when extracting ResNet features we converted all images to RGB. Input images for the ResNet-101 model were also further resized to 224px $\times$ 224px to align with the model's training data. In this case, images were not center-cropped; instead, images were downsampled and then interpolated.

2.2 Local Binary Patterns (LBP)

Local Binary Patterns (LBP) is a simple texture descriptor that analyzes the intensity differences between a pixel and its neighboring pixels. By comparing the intensity of a central pixel with those of its neighbors within a defined radius, a binary pattern is generated. Each neighbor is assigned a binary value of 1 1 if its intensity is greater than or equal to the central pixel and 0 otherwise. These binary values are then concatenated to form a binary pattern and converted into a decimal value. The LBP values across an image are aggregated into a histogram, which serves as a feature representation of the image's texture. An example of this representation is seen in Figure 3. We hypothesized that LBP could serve as a valuable additional feature for differentiating office objects based on local textural patterns. For example, a mug typically exhibits minimal local texture variation, whereas objects like a keyboard display significant and distinct textural patterns due to the arrangement of keys.

To enhance the power of LBP, we used multi-scale LBP, which extends the analysis to neighbors at varying distances from the central pixel. Additionally, we introduced rotational invariance to ensure robustness in scenarios where objects are captured in different orientations. We did this by using the ROR method which is a rotation-invariant uniform pattern. The ROR method rotates the binary pattern circularly to find the smallest possible binary value, effectively normalizing the pattern across all orientations. This ensures that different rotations of the same texture produce the same LBP code, enhancing robustness to changes in object orientation.

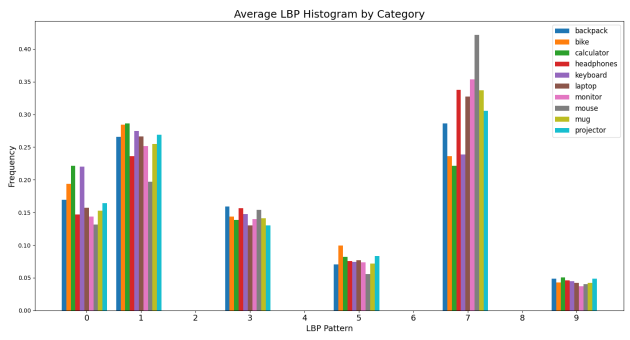

Figure 3: Average LBP histogram by category of objects

2.3 Gray Level Co-Occurrence Matrix (GLCM)

The Gray Level Co-occurrence Matrix (GLCM) is a matrix that characterizes the spatial relationship between pairs of pixels in an image. Specifically, it quantifies how often pixel pairs with specific intensity values occur in a given spatial relationship, defined by a certain distance and angular offset. An example of this matrix can be seen in Figure 4 which demonstrates a sample of original images and their corresponding matrices.

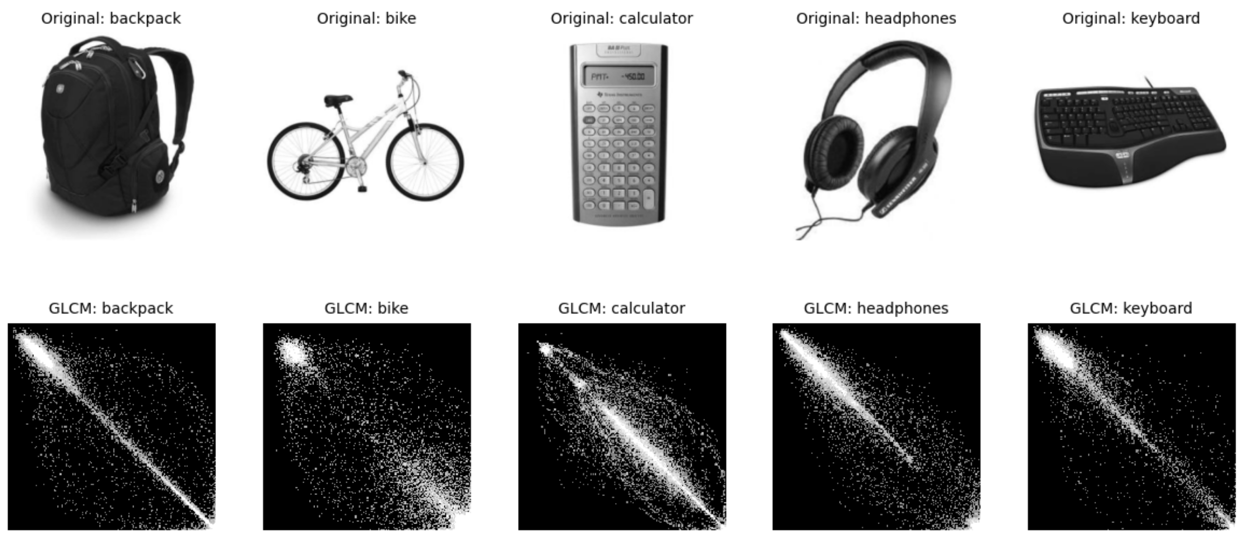

Figure 4: A sample representation of original images and associated GLCM

In our analysis, we computed multiple GLCMs for each image using various angle orientations. We ran several experiments to look for parameters with the best performance in training and validation. After experimenting we found that the best performance was achieved with angles of 0, $\pi/4$, $\pi/2$, and 3 $\pi/4$ radians and distance offsets of 1, 2, 4 and 8 pixels. We then computed the GLCM for each combination of these parameters. From these matrices, statistical measures can be applied to generate features. The most common features are contrast, dissimilarity, homogeneity and correlation. An example of these features can be seen in Figure 5 which demonstrates these four features for each of the object categories with a GLCM angle of 0 and a distance offset of 1.

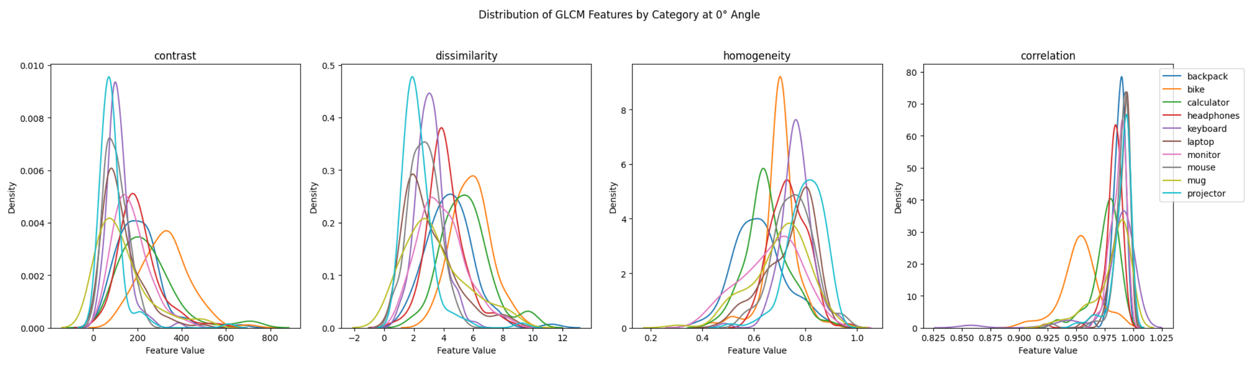

Figure 5: GLCM features with angle 0 degrees and offset of 1 pixel

One issue we considered when considering GLCMs in our feature set was whether or not it would be robust to rotation. GLCM is not rotationally invariant out of the box, so we experimented with making GLCM rotationally invariant by averaging matrices produced from multiple angles. However, in our tests we found that this degraded performance. One likely cause for this is that many real world images are in the upright position and therefore accounting for rotation could generalize poorly in a majority of real world images. Nevertheless, we did make certain choices that are more robust to rotation, including selecting angles and statistical features (such as contrast) that are less sensitive to object orientation.

2.4 Color Channel Features

The mean values of the red, green and blue (RGB) color channels were used as a feature to capture the overall color intensity of the input image. For natural scenes, the average RGB values provide an efficient summary of the image's color composition, which can help differentiate between regions with varying lighting or dominant colors. Additionally, the mean RGB value is computationally simpler and less sensitive to local variations. This makes it a complementary feature for models relying on both high-level and low-level color information.

The mean RGB value provides a holistic measure of color, making it particularly suited for applications where global color trends are significant. This is quite valuable when the training set is a good approximation of the test set, as is usually the case with machine learning tasks. However, domain adaptation requires that the training set be an imperfect approximation of the test set/real world data. This discrepancy can lead to some surprising behavior, as shown in Figure 6 which plots RGB values for backpack images from each domain. Since the Amazon photo is on a solid white background, the Color Histogram for Amazon image in Figure 6 has a disproportionately high percentage of pixels at maximum intensity (250-255). Despite these differences across domains, RGB values proved to be a valuable addition to our feature set.

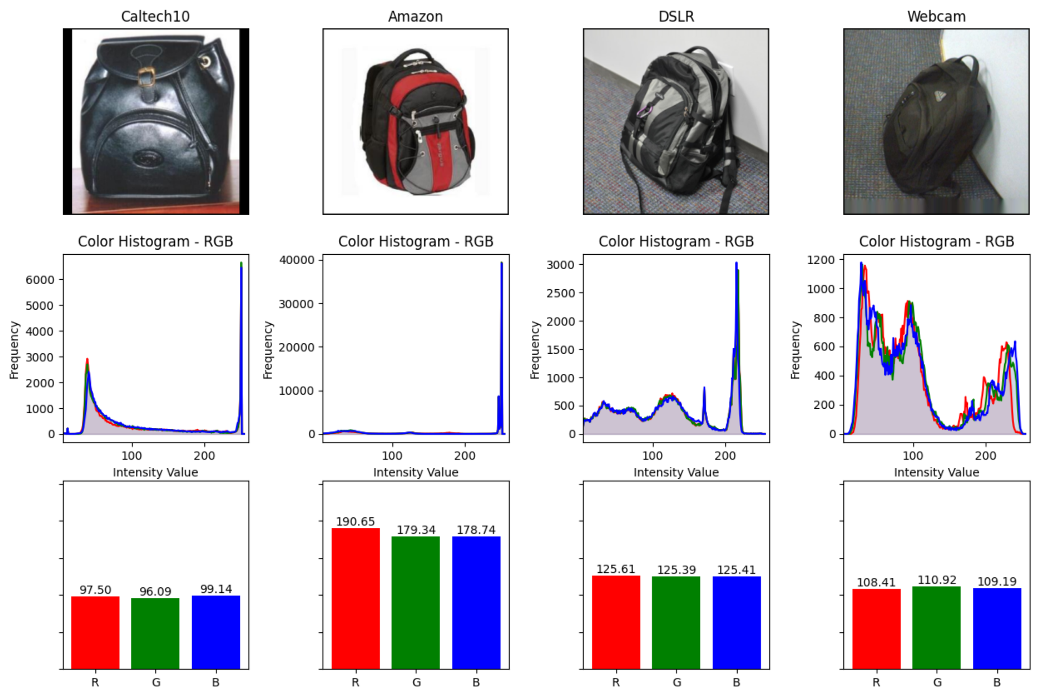

Figure 6: Original image, RGB channel intensity histogram, and mean RGB values

2.5 Gabor Filters

Gabor filters are linear filters that are orientation and frequency specific and are useful for looking at changes in texture and edges. For the office objects, the objects have differences in textures and hard edges that make Gabor a suitable feature. In addition, compared to GLCM and LBP, Gabor filters operate in the frequency domain instead of the spatial domain. This allows it to be a complementary instead of redundant feature. For these reasons we initially worked on extracting features using Gabor filters. However, during experimentation we found that extracting these features was computationally intensive with extraction of features with a single set of parameters taking 44.2 minutes. Given this inefficiency, we felt that tuning filter parameters would be unrealistic given our limited computational resources and therefore removed Gabor filter from our final pipeline.

2.6 ResNet-101 features

For our complex feature we looked at using embedding features from a neural network. After some background research we decided to use ResNet-101. ResNet-101 is a pre-trained convolutional neural network (CNN) with 101 layers, originally trained on the ImageNet-1k dataset, which contains over 1.2 million images across 1,000 object categories [3]. The ResNet family of models are highly accurate in classification tasks. We therefore hypothesized that extracting a layer from one of the final layers of a ResNet model may be useful as features in a transfer learning setting. The final layers of CNNs like ResNet contain rich high-level features that are highly generalizable for use in classifcation.

For this project, we chose to extract the 2048-dimensional feature vector from the global average pooling (GAP) layer. We felt using the GAP layer might generalize better with smaller datasets and is more robust to overfitting than the fully connected layer. We should also note that images were resized as ResNet accepts input images of 224x224 pixels with three color channels (RGB).

2.7 ORB and Bag of Visual Words

Bag of Visual Words (BoVW) is a technique that measures the number of occurrences of local image features (i.e. ”visual words”) in each image. The BoVW features for our model were computed in 3 steps:

1. Keypoints in each image were identified using ORB (Oriented Fast and Rotated BRIEF). ORB is a binary local feature detector that was selected because it is highly efficient, being significantly faster than other keypoint detector algorithms (e.g. SIFT, SURF) while offering comparable performance[4]. In addition, ORB keypoints have other desirable features such as rotational invariance, scale invariance, and robustness to changes in global illumination. Each keypoint identified by ORB has a corresponding descriptor vector, typically thought of as an embedding representation of the patch around the keypoint.

2. Descriptors were clustered into n clusters using K-Means clustering. Cluster centroids are determined on the training dataset and descriptor vectors are mapped to the cluster with the lowest Euclidean Distance. The result is a vector of length n, where the value in the i_{th} index is the number of occurrences of keypoints found in the image that are associated with the i_{th} cluster label. Each element in the vector can be thought of as a different "visual word."

3. The TF-IDF transformation (Term Frequency-Inverse Document Frequency) was applied to the computed occurrence matrix. This re-weights the BoVW features to account for the relative frequency of each word across all words identified in the training dataset. For example, the TF-IDF frequency of visual words that are present in almost every image will be penalized relative to the TF-IDF frequency of globally infrequent words.

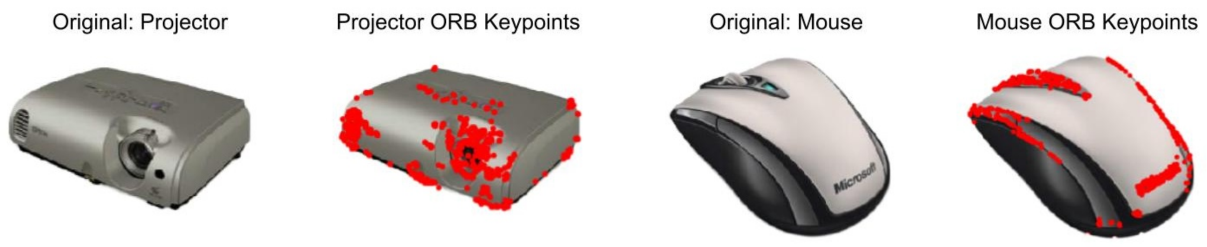

Figure 7: Examples of identified ORB Keypoints in Projector and Mouse classes

2.8 Principal Component Analysis

One of the major issues with extracting thousands of features from a dataset is the curse of dimensionality. In high dimensional spaces, the data points within that space can become sparse and this can lead to an inability to identify meaningful relationships. The problem of high dimensionality can also make it easy to overfit models and reduce generalizability. Finally, high dimensionality can increase computational resources needed and thus decrease efficiency.

To address the problem of high dimensionality, we used principal component analysis (PCA) to reduce dimensions. PCA transforms data linearly from a higher dimensional space onto a lower dimensional space. It works by identifying orthogonal axes also called principal components that capture as much of the variance in the features as possible.

The PCA of each individual feature can be seen in Figure 8. For some simple features such as LBP and GLCM very few components are required to explain the majority of the variance. For our complex feature of the ResNet-101 embedding, the number of components to explain more than 90% of the variance exceeded 200.

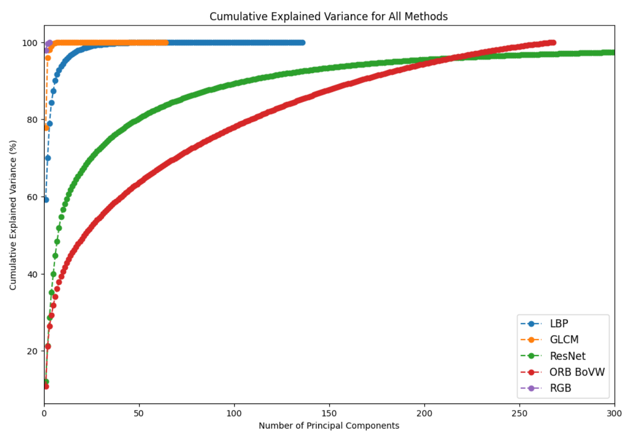

Figure 8: PCA of individual features

As part of our study and during modelling, we ran experiments to investigate the ability of different feature sets to provide added performance to our models. The first model consisted of simple features only and this was able to explain 99.9% of variance with only eight components (Figure 7). This remained the same when PCA was applied to all the simple features and ResNet-101 embedding features. Finally, we looked at PCA for ResNet and Orb features combined as this was the best performing model in testing. PCA was able to reduce the dimensions, but 50 components were required to explain 80% of the variance and 108 components were required to explain 90% of the variance.

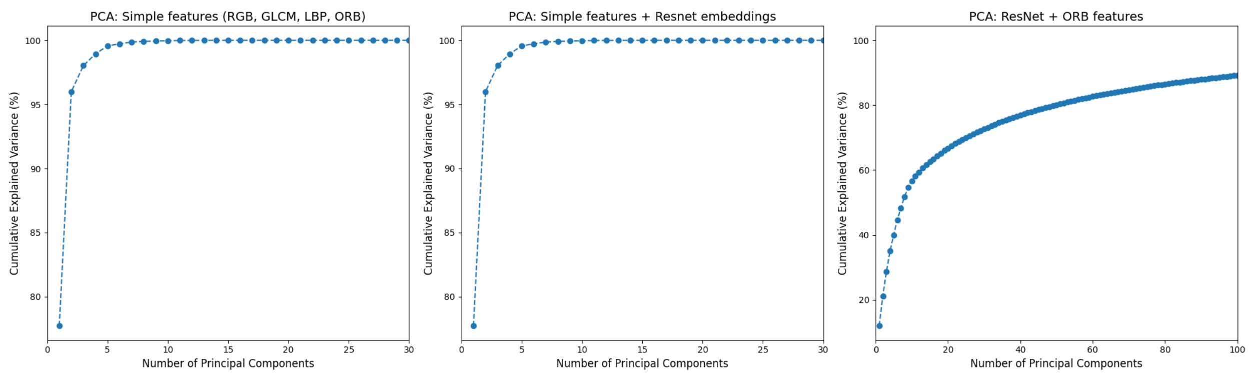

Figure 9: PCA components required to explain variance across several feature sets

2.9 T-distributed Stochastic Neighbor Embedding (t-SNE)

T-distributed Stochastic Neighbor Embedding (t-SNE) is a dimensionality reduction method primarily used for data visualization. Unlike PCA, which emphasizes preserving global variance, t-SNE focuses on maintaining the local relationships between data points. It embeds high-dimensional data into a lower-dimensional space (commonly 2D or 3D), ensuring that points that are close together in high-dimensional space remain close in the lower-dimensional representation.

Using t-SNE is particularly useful for visualizing and interpreting complex datasets. In our analysis, we used t-SNE to explore whether objects cluster based on their features. For certain features, such as RGB, clear clustering is not apparent as seen in Figure 10. However, for ResNet-derived features, distinct separation between object clusters is observed. This suggests that ResNet features are highly effective at distinguishing between different objects.

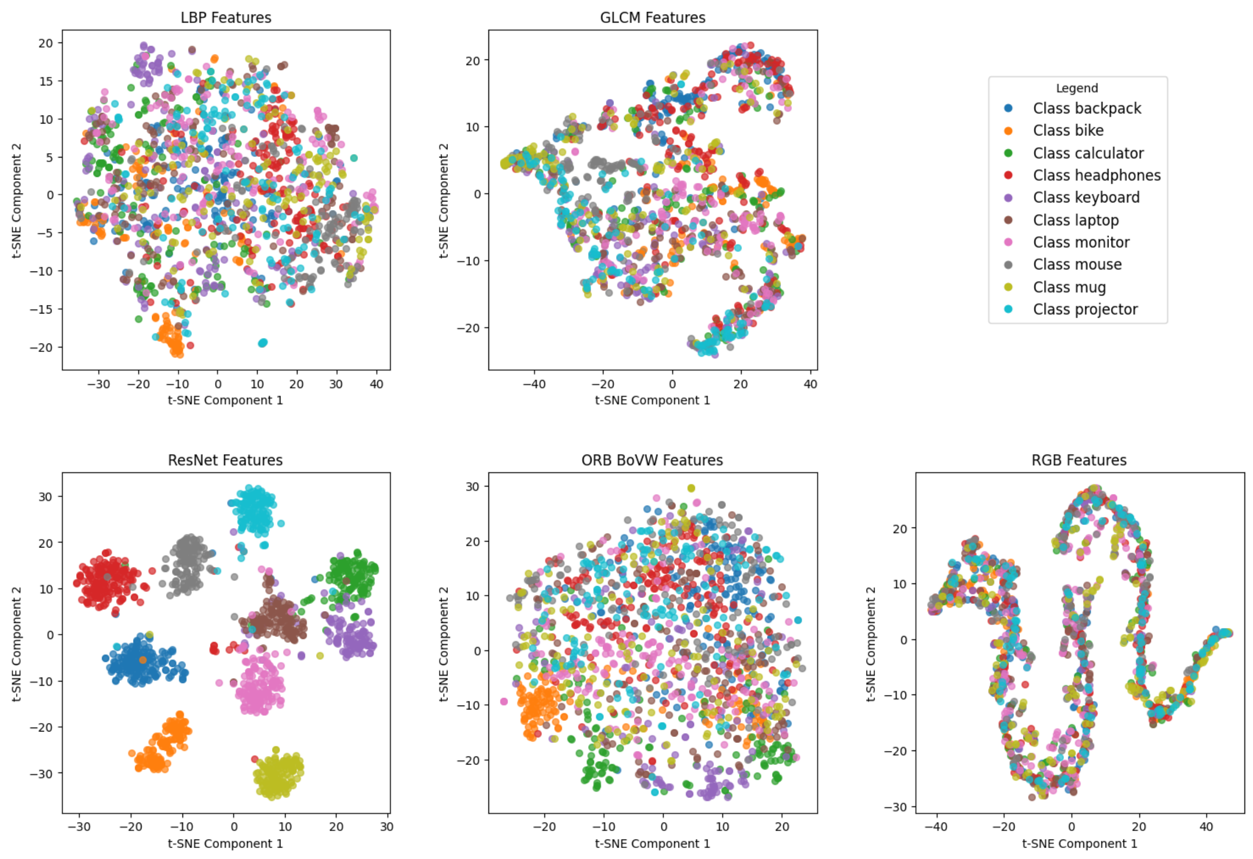

Figure 10: t-SNE for different sets of extracted features

3 Classification

The classification task in our study aimed to leverage the extracted features to build robust models capable of classifying objects in different domains. In pursuit of this goal, we trained and evaluated models with 2 common frameworks: In the context of our domain adaptation project, the performance of Support Vector Machines (SVMs) and Random Forests (RFs) is influenced by their inherent strengths and weaknesses.

SVMs should theoretically excel in the “Amazon” and “Caltech10” domains, where images are well-structured and have minimal noise. Their reliance on maximizing the margin between classes enables them to handle the clear boundaries often found in these curated datasets. When applied to the “DSLR” and “Webcam” domains, SVMs still demonstrated strong performance despite the increased variability in lighting, resolution, and perspective. This robustness can be attributed to their capacity to use kernels, such as the RBF kernel, to adapt to nonlinear separability across domains.

In contrast, Random Forests, while generally robust to noise and variability, are less effective in this context. The ensemble approach of RFs averages decision trees, which can lead to a loss of precision when subtle, high-dimensional boundaries are crucial for classification. This limitation becomes evident when distinguishing between classes in the more complex and diverse "DSLR" and "Webcam" domains, where SVMs' margin-maximization strategy provides a slight advantage.

We evaluated both of our classification models on two test sets:

- **Test Set A:** A 10% holdout of in-domain images
- **Test Set B:** The out-of-domain set of images designed to assess the models' ability to generalize across domains.

The results, summarized in Table 1, revealed the importance of feature selection and hyperparameter tuning. Support Vector Machines (SVMs) consistently outperform Random Forests (RFs) in both test sets due to their ability to model complex decision boundaries and adapt to high-dimensional data. These findings underscore the superior adaptability of SVMs for cross-domain classification tasks within this dataset.

SVM with Tuning: Models utilizing ResNet embeddings achieved the best overall performance, with up to 97.6% accuracy on Test Set A and 95.1% on Test Set B.

Random Forest with Tuning: Achieved comparable performance on in-domain data (96.2%) but slightly lower accuracy on out-of-domain data (90.3%).

Feature subsets including ResNet embeddings consistently performed better than others within each classifier type, highlighting the critical role of deep learning-based features in cross-domain tasks. It should be noted that since we did not retrain the ResNet features from scratch with randomized initial weights, images similar to the "out-of-domain" test set may have been included in the ResNet test set, potentially degrading the out-of-domain assumption of this project.

Model Type	**Features**	**In-Domain Train Accuracy**	**Test Set A**	**Test Set B**
SVM	ResNet, ORB, RGB, LBP, GLCM	0.970	0.976	0.951
SVM	ResNet, ORB	0.968	0.981	0.951
SVM	ORB	0.925	0.605	0.502
SVM	ORB, RGB	0.930	0.610	0.509
SVM	ORB, RGB, LBP, GLCM	0.990	0.767	0.365
RandomForest	ResNet, ORB, RGB, LBP, GLCM	1.000	0.962	0.903
RandomForest	ResNet, ORB	0.999	0.957	0.912
RandomForest	ORB	1.000	0.462	0.412
RandomForest	ORB, RGB	0.993	0.495	0.447
RandomForest	LBP, GLCM, ORB, RGB	1.000	0.605	0.254

Table 1: Performance Metrics of Models on Test Set A (in-domain) and Test Set B (out-of-domain)

The confusion matrices (Figure 11) illustrate the classification performance of the SVM models under both in-domain and out-of-domain scenarios, comparing results with simple features (no ResNet) to those incorporating all features, including ResNet embeddings. In the in-domain setting, the model performs quite well with only the simple features. However,incorporating ResNet features significantly improves classification accuracy, as evident from the reduced off-diagonal elements and stronger diagonal dominance in the corresponding matrix. This indicates that ResNet features enhance the model's ability to distinguish between categories with greater precision.

In the out-of-domain scenario, the benefit of ResNet features is even more pronounced. While the model using simple features struggles with significant misclassifications, the inclusion of ResNet features results in a substantial reduction of errors across almost all categories. Particularly challenging classes like "headphones" and "monitor" show marked improvement, demonstrating the effectiveness of deep feature representations. Overall, these results highlight the importance of incorporating robust feature extraction methods, like ResNet, in improving cross-domain generalization and classification accuracy.

4 Generalizability

Generalizability describes how well a trained model performs on unseen data. This is a core concern in domain adaptation, as we seek to adapt features trained on in-domain data to make predictions on out-of-domain data. To maximize generalizability, we train our models on 70% of in-domain images and perform hyperparameter tuning with an additional 20% of data. We use 5-Fold Cross Validation with Random Search over 50 iterations to identify optimal hyperparameters. Although tuning is only conducted on in-domain data to avoid leaking out-of-domain information during training, it yields parameters that are more robust to variation in in-domain training data. Finally, we report predictions on two distinct testing sets. Test Set A is the last 10% of in-domain data that was unseen in training and tuning, while Test Set B is composed of strictly out-of-domain images.

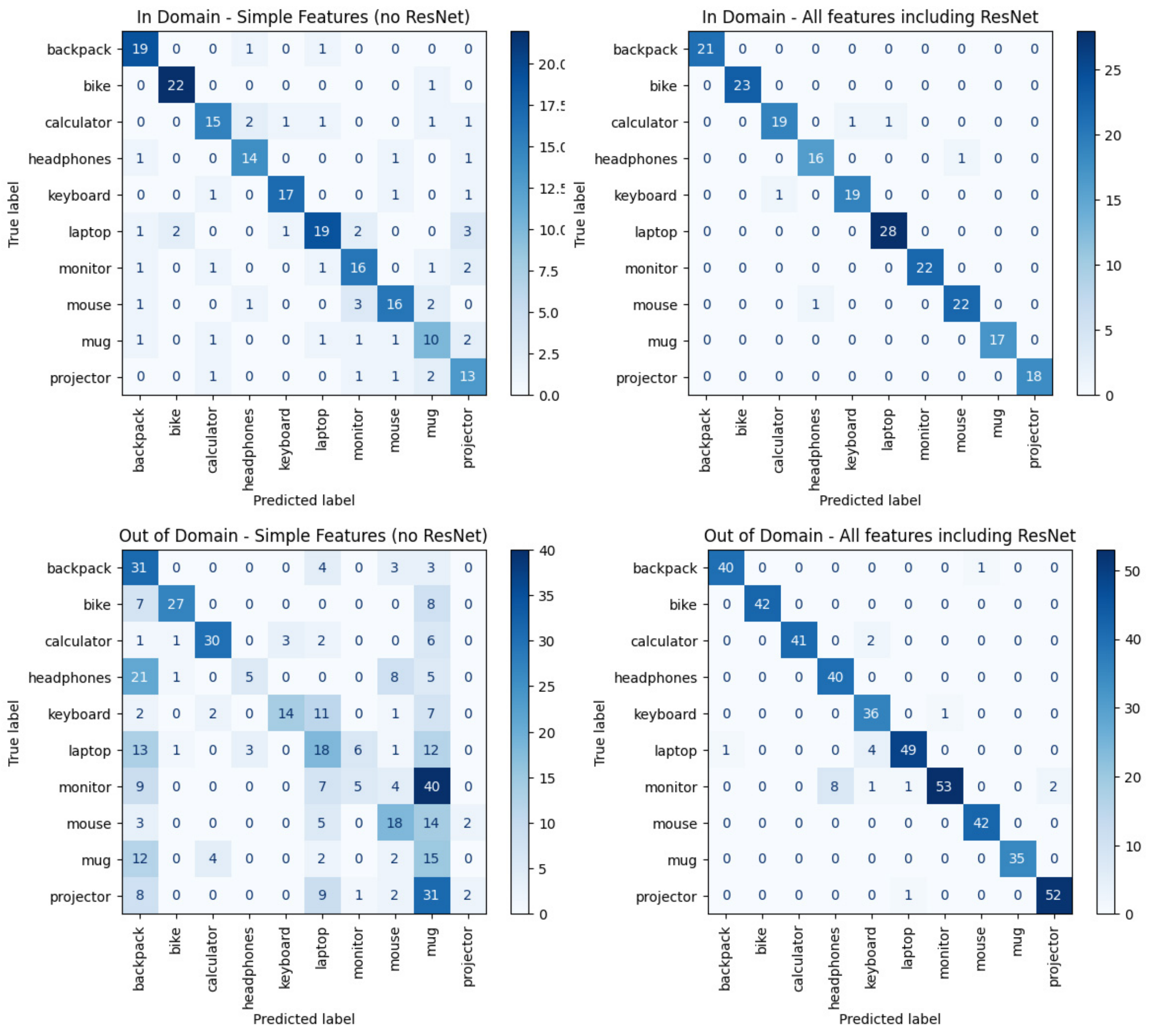

Figure 11: Confusion Matrices with SVM models - Top left: In domain test with model built on simple features, Top right: In domain test with model using all features including ResNet, Bottom left: Out of domain test with model built on simple features, Bottom right: Out of domain test with model built with all features including ResNet

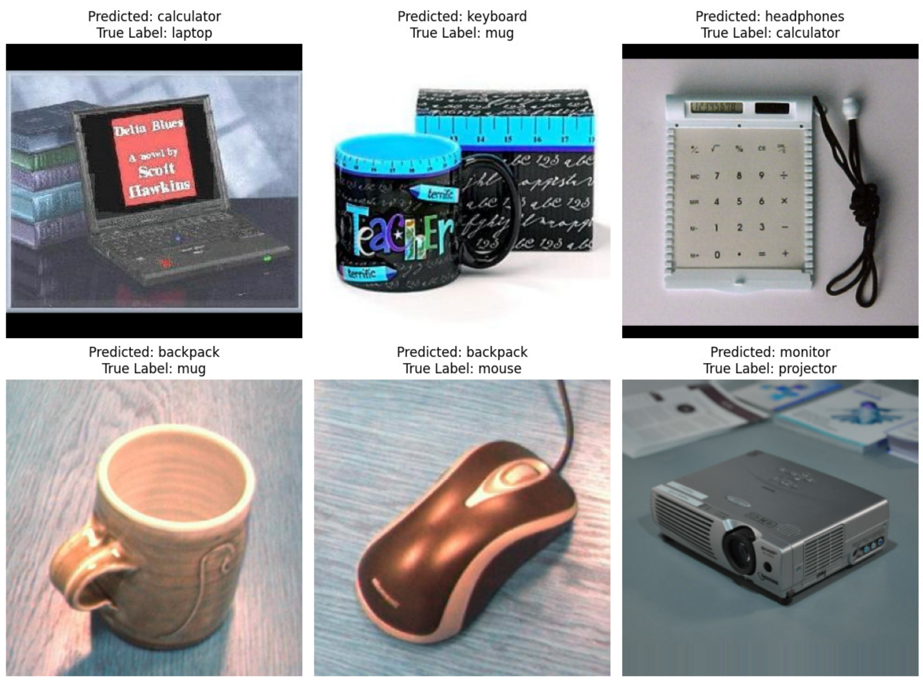

Figure 12: Examples of misclassification. Top Row: Misclassification examples from the best performing scenario (SVM model testing in domain images with all features including ResNet). Bottom Row: Misclassification examples from the worst performing scenario (random forest model built with simple features alone testing out of domain images with simple features alone)

As seen in Table 1, the choice of features was particularly important for maximizing model performance on out-of-domain data. Inclusion of Resnet-101 embeddings led to the most dramatic improvement in model performance on both test sets, likely because Resnet-101 was pretrained on a very diverse set of images. As a result, the embeddings derived from the model are extremely robust to the differences in lighting and orientation between domains.

Removal of the Resnet-101 features led to a dramatic drop in classification accuracy. Notably, the model with all features except Resnet-101 has better performance on the in-domain test set than the model with only ORB BoVW and Mean RGB features. However, its accuracy plummets on out-of-domain data. This indicates that, although the first model's features are robust to differences in Test Set A, it is dramatically overfitting when evaluated against Test Set B. By contrast, although the latter model has comparatively worse performance on in-domain data, its fewer features are robust to the differences across domains seen in Test Set B.

The difference in generalizability of models with and without ResNet-101 features is further highlighted in Figure 12. The top row shows the very few examples of misclassification in our best performing setup. Even with Resnet-101 features tested in-domain, the model can still misclassify. However, the reasons for this can generally be deduced. For example, in the top left misclassification example, the orientation of the keyboard below a screen is quite similar to what a calculator might appear at an angle. By contrast, the bottom row shows how the comparatively simpler model without ResNet-101 features generalized far more poorly; many of the misclassifications in this setup are hard to explain.

5 Efficiency

Efficiency describes the trade-off between classifier performance and computational cost required for training and inference. For example, if a novel feature A is extremely expensive to compute and provides only minor benefit to classifier performance, it is considered highly inefficient. This is a central problem in deep learning- a hypothetical classifier with millions of parameters may have high accuracy but be infeasible in practice due to high compute cost.

Since many classical CV features are expensive to compute, feature extraction had the largest impact on overall model efficiency. To standardize compute time, we timed feature extraction on the same machine which has an Intel I9-14900KF processor with 64gb of RAM. GPU was not leveraged when extracting features. We measured compute time for every feature on the entire image dataset and report the average time per observation in the table below.

Of the examined features, Gabor took the longest to compute by a significant margin. In addition, experiments with Gabor features indicated that they provided little marginal benefit to model performance. Although they did not need to be re-computed for the rest of our project, we decided to eliminate them from our model entirely to simulate real-world scenarios in which high compute cost would make it difficult to use for inference on new data. Besides Gabor, no other feature was prohibitively slow to compute. However, as demonstrated in Table 1, a model built with only Resnet-101 and ORB BoVW features maximizes efficiency without sacrificing any additional accuracy.

Feature	**Compute Time / Observation**
LBP	0.120 seconds
GLCM	0.021 seconds
Mean RGB	0.000 seconds
Gabor	2.676 seconds
Resnet-101	0.003 seconds
ORB BoVW	0.016 seconds

Table 2: Feature Extraction Time in Seconds per Observation

6 Conclusion

Overall, this project demonstrates the complexity and importance of feature selection in cross-domain image classification tasks. In particular, it highlights the importance of including highly robust, descriptive features such as BoVW and Resnet-101 embeddings to build models that generalize across a diverse set of image conditions. Without embeddings extracted from deep learning models, our models built with traditional computer vision features were prone to overfit on in-domain data despite leveraging PCA and 5-fold CV for hyperparameter tuning. Future work seeking to reduce overfitting without utilizing deep learning could explore strategies such as data augmentation that is specifically tailored to mimic the differences across domains.

References

[1] Saenko, K., Kulis, B., Fritz, M., Darrell, T. (2010). Adapting Visual Category Models to New Domains. In: Daniilidis, K., Maragos, P., Paragios, N. (eds) Computer Vision – ECCV 2010. ECCV

2010. Lecture Notes in Computer Science, vol 6314. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-15561-1

[2] Griffin, Gregory & Holub, Alex & Perona, Pietro. (2007). Caltech-256 Object Category Dataset. CalTech Report.

[3] He, K., Zhang, X., Ren, S., Sun, J. (2015). Deep Residual Learning for Image Recognition. CoRR, abs/1512.03385. Available at: http://arxiv.org/abs/1512.03385

[4] E. Rublee, V. Rabaud, K. Konolige and G. Bradski, "ORB: An efficient alternative to SIFT or SURF," 2011 International Conference on Computer Vision, Barcelona, Spain, 2011, pp. 2564-2571, doi: 10.1109/ICCV.2011.6126544.