Geographic Landmark-Based Visual Localization

Bosen Chia, Brandon Tzou, Hildah Ngondoki, Rick Pereira

W281 Computer Vision Final Project

Fall 2025

Github link: https://github.com/hngondoki/geographic_landmark_based_visual_localization

Abstract

Geographical landmarks are important in image organization and navigation systems. This project explores how landmark visual search can be improved using computer vision features and classification algorithms.We conduct experiments to identify the most effective features and present a comparative analysis of four classification methods: Logistic Regression, K Nearest Neighbour (KNN), Support Vector Machines(SVM) and a Resnet Convolution Neural Network (CNN). After experimentation we found that complex features and combinations that included them outperform the simple features selected because they capture richer and more discriminative representations of the data. Deep feature extractors such as ViT and ResNet encode high-level spatial, textural, and semantic patterns that simple handcrafted descriptors cannot. As a result, models built on these complex features provide clearer class separability, higher AUC values, and more robust generalization. A combined mix of simple and complex features yielded balanced results that had a smaller test train gap hence preferred to prevent overfitting.The SVM model achieved the strongest performance, reaching an accuracy of 98.60% and a precision of 98.61%, slightly surpassing the other models. KNN achieved nearly equivalent performance while being more time-efficient, making it the most practical choice for landmark classification.

1. Introduction

Automated landmark recognition is the process of predicting specific landmark labels directly from image properties. It is a vital technology that underpins the organization of massive photo collections, enhances image search functionality, and bolsters geospatial intelligence. Our approach focuses on developing a highly accurate landmark classification model using a chosen subset of the Google Landmark Recognition dataset (Melendez, 2023). The initial phase will concentrate on the top 8 most frequently occurring landmark categories within the full dataset to establish a classifier, choosing these specific landmarks because they are highly recognizable and share overlapping visual properties that can challenge and refine the model's discriminative capabilities.

We explore features such as Histogram of Gradients (HOG), Color Histogram (RGB) and HSV histogram, and augment it with advanced feature embedding techniques using ViT and a pretrained ResNET model. For classification, we used Logistic Regression, K Nearest Neighbour (KNN), Support Vector Machines(SVM) and a Resnet Convolution Neural

Network (CNN). Due to the large dimensionality of the features chosen, we used Principal Component Analysis (PCA) to reduce the dimensions. We then used different combinations of the features to determine the optimal combinations. We started by classifying the histogram features individually and as combinations. Followed by each of the complex feature embeddings, and finally as a combination of all features. The selected metrics are accuracy, precision, and f1-score. The precision is more important in navigation problems because it provides more relevant results. For selection of the best performing model, we shall use auc-curve because our dataset is balanced and it provides a general measure of separability between classes.

2. Exploratory Data Analysis and Data Preprocessing

We began our exploratory data analysis (EDA) by examining the top 30 most frequent categories in the dataset. During this process, we identified several categories that were not well-suited for a supervised landmark classification task. These included categories whose images did not correspond to a single, well-defined landmark (e.g., broad geographic regions or cityscapes), as well as categories containing scenes without a distinctive object or structure that could be consistently recognized.

After this review, we manually selected eight final categories for our classification experiment: *Grand_Canyon*, *Matka_Canyon*, *Eiffel_Tower*, *Edinburgh_Castle*, *Golden_Gate_Bridge*, *Niagara_Falls*, *Skopje_Fortress*, and *Nieuwe_Waterweg*. These categories were chosen because they have comparatively high sample counts, represent well-known landmarks with visually distinctive features, and collectively offer a diverse yet challenging set of classes for model learning. This selection ensures a balanced dataset with clear visual identities while still providing enough similarity across classes to test the robustness of our classification methods.

Figure 2.1. Example category that is not a landmark

Category: Corktown,_Toronto (Label: 22)

Figure 2.2. Example images of a city

Category: Golden_Gate_Bridge (Label: 19)

Figure 2.3. Example of ideal category

3. Feature Extraction

Histogram of Oriented Gradients (HOG)

The Histogram of Oriented Gradients is a feature descriptor used for object detection and recognition. HOG is ideal for landmark classifications because it works by capturing the local appearance and shape of objects within an image by characterizing the distribution of edge directions or gradients within localized portions of the image. It is well suited for representing architectural identity because it involves dividing the image into small, connected regions called cells, and for each cell, a histogram is computed that bins the gradient orientations of the pixels. These histograms are then normalized across larger overlapping blocks to account for changes in illumination and shadowing.

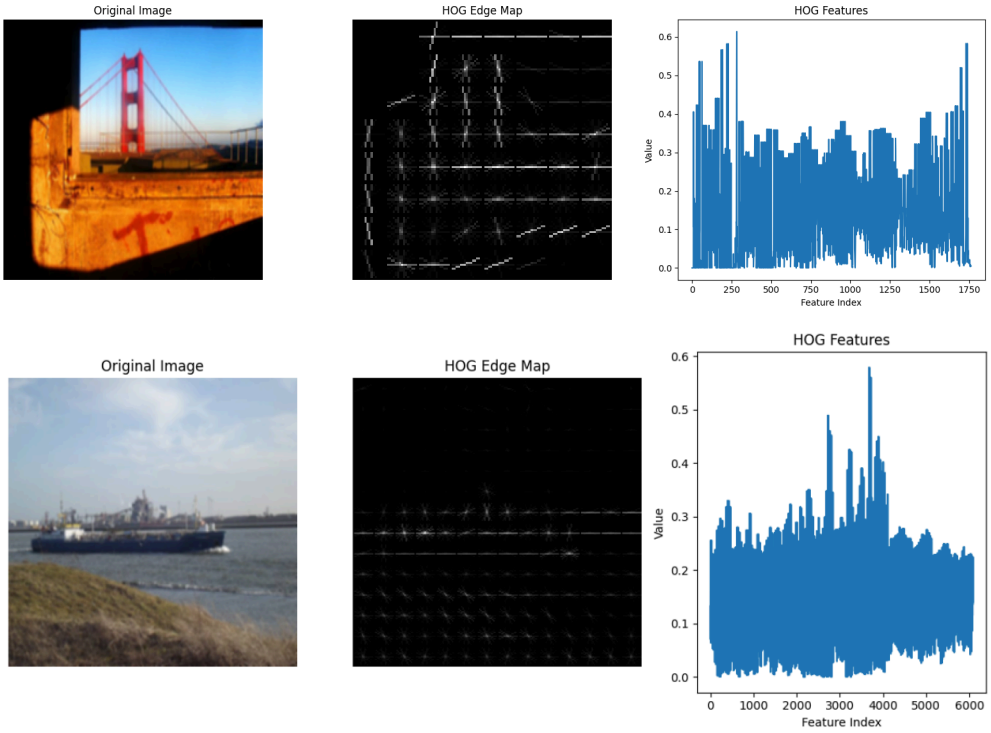

Figure 3.1. Sample images showing the original image, HOG edge detection and histogram.

HOG fits into this landmark feature extraction pipeline by providing a way to encode the architectural and geometric patterns such as lines, contours, and corners that are essential for distinguishing one landmark from another (e.g., the vertical lines of the Eiffel Tower vs. the curved arch of the Golden Gate Bridge). By focusing on gradient orientations, HOG provides a representation that is relatively invariant to local photometric changes like brightness shifts, which complements color-based features and provides the structural information useful for a classifier.

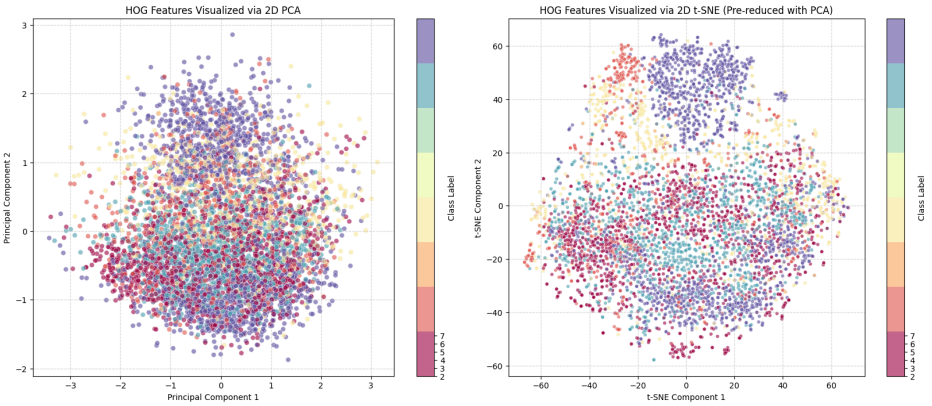

Figure 3.2. 2D PCA and t-SNE representation of HOG features illustrates limited separability between landmark classes.

RGB Color Histogram

The RGB Color Histogram provides visual and numerical representation of how colors are distributed in an image, based on the Red (R), Green (G), and Blue (B) color channels which makes it suitable for landmark classification due to the distinct color signatures of landmarks. Examination of various image types reveals that RGB channel distributions often exhibit distinct patterns that can assist in differentiating between classes. For this reason, we employ raw RGB histograms, which preserve detailed and discriminative information across channels. The resulting histograms depict independent distributions for the red, green, and blue channels, each reflecting the spread of pixel intensities within the image. Analyzing these channel-wise distributions allows us to assess overall color balance and identify cases in which one channel may dominate or be underrepresented. Although HSV histograms are frequently favored for color analysis due to their perceptual advantages, the RGB Color Histogram remains highly relevant because it corresponds directly to the way most digital image sensors acquire and encode color information.This alignment makes RGB-based analysis both practical and interpretable.

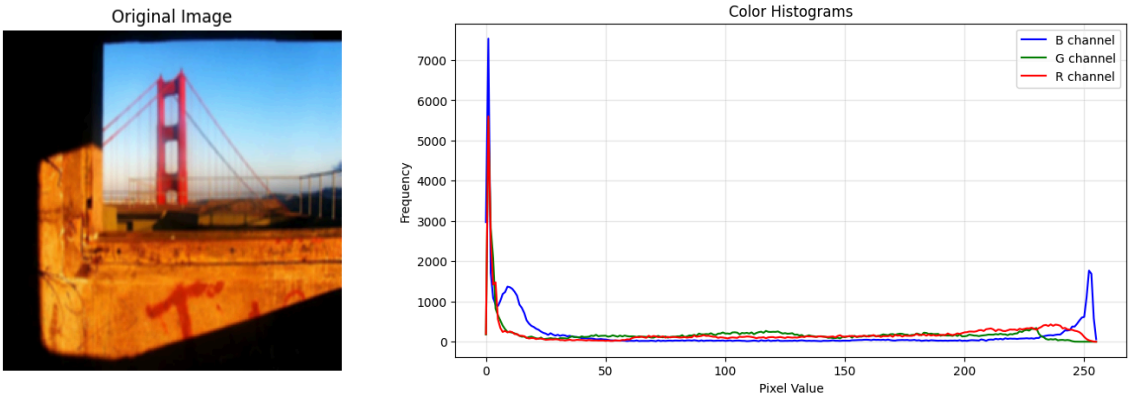

Figure 3.3. The original image(left) and color histogram(right) represent a high contrast photo with high values on both left and right sides of the histogram and fairly low middle tones.

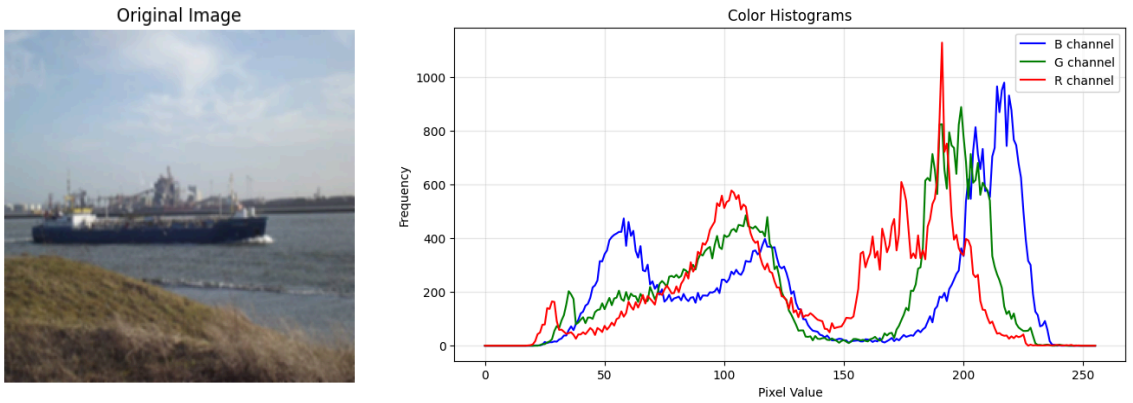

Figure 3.4. The original image(left) and color histogram(right) represent dominant color regions indicating balanced distribution of colours, with higher peaks towards the right of the histogram.

The PCA and t-SNE Visualization of Color Features show that RGB histograms alone do not provide strong discriminative power for distinguishing between landmarks. Many images share similar global color distributions, especially those containing natural scenery. While a few categories with distinctive color palettes show partial grouping, most classes overlap extensively due to the global and spatially agnostic nature of color histogram features.

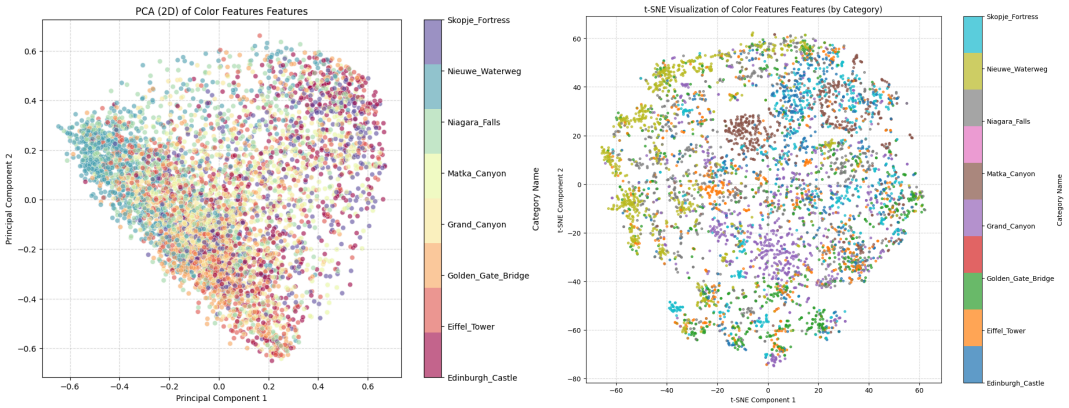

Figure 3.5: The PCA (left) and t-SNE(right) visualization demonstrates that raw RGB histograms offer limited discriminatory capability for landmark classification.

HSV Histogram

HSV separates color (Hue) from brightness (Value), making them more robust to changes in illumination and more closely aligned with human perception. This separation simplifies tasks like object recognition, where the color itself is a key feature, regardless of lighting conditions.

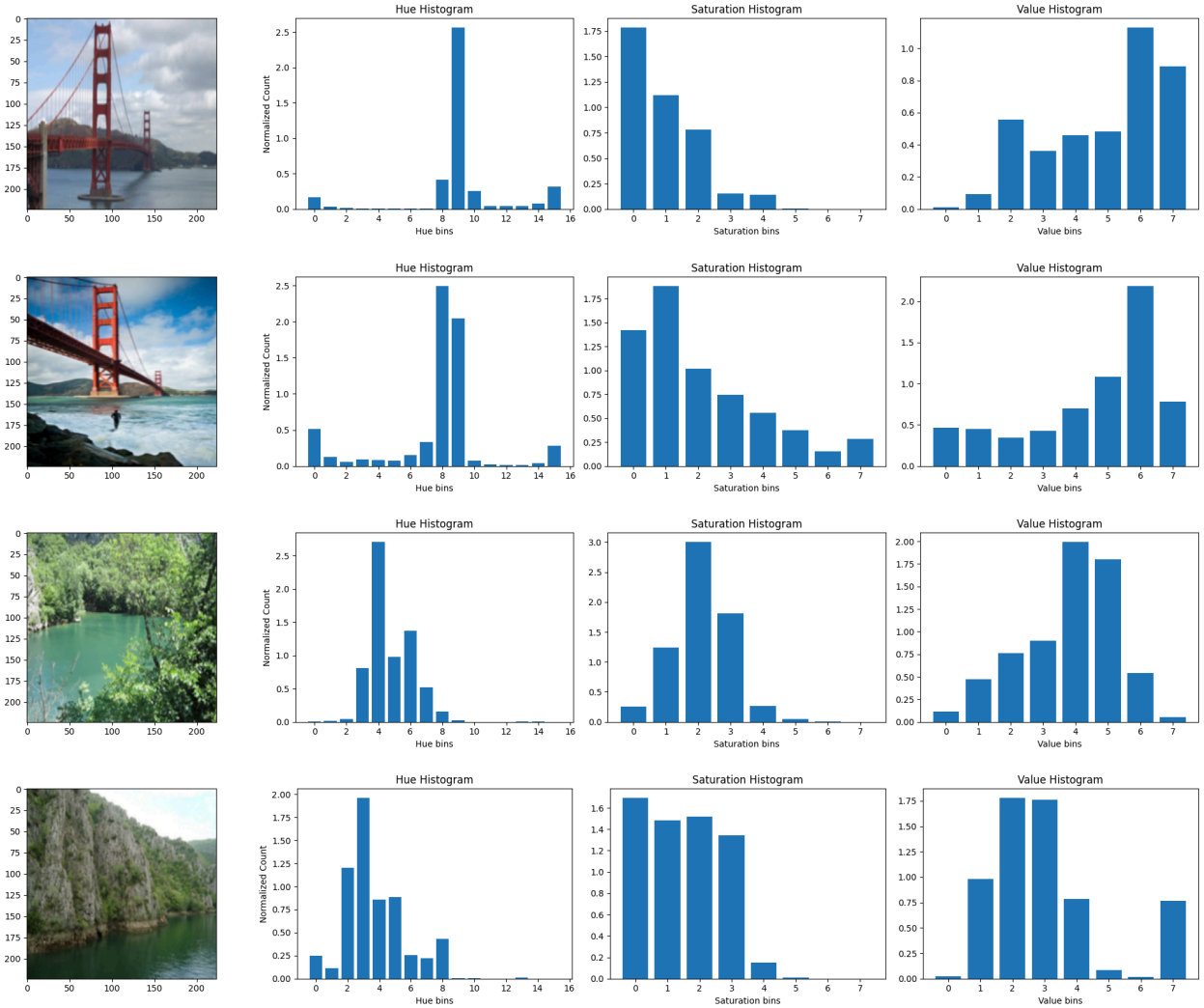

Figure 3.6. Original image (left) and HSV Similarities and differences between categories

The PCA and t-SNE representations of HSV features show poor discriminative power similar to RGB color Histograms. We observe varied or overlapping color profiles, making it difficult for histogram-based features to separate them in a low-dimensional space.Despite overall mixing, a few categories form noticeable but loose clusters, such as Nieuwe Waterweg which often contain bright water and structural elements.

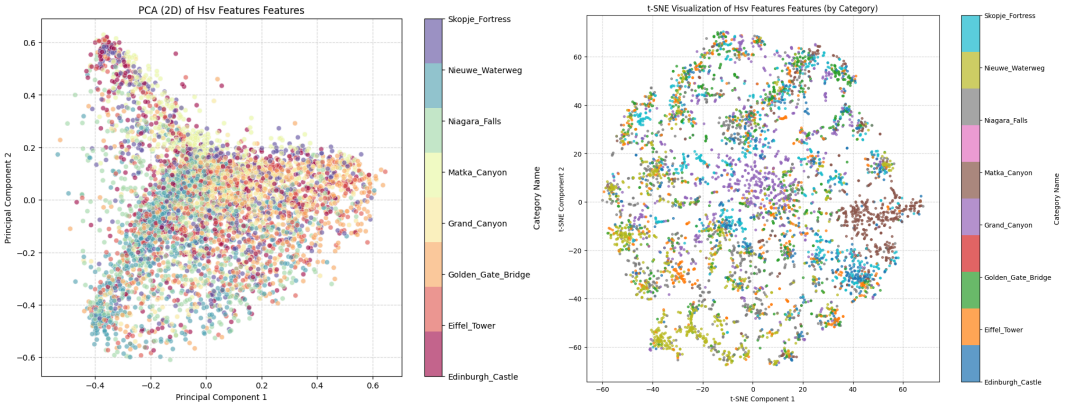

Figure 3.7. The PCA (left) and t-SNE(right) visualization demonstrates that histograms offer limited discriminatory capability for landmark classification

ViT Feature Embeddings

The Vision Transformer feature embedding is a complex, deep-learning representation that serves as a powerful alternative to traditional Convolutional Neural Networks for image recognition. It treats an image as a sequence of fixed-size, non-overlapping patches, analogous to words in a sentence. Each of these flattened image patches is mapped to a vector space through a linear projection, creating the initial patch embedding. To reintroduce the spatial information lost during the flattening process, positional encodings are added to the patch embeddings which produces highly discriminative representations suitable for landmark classifications. This complete sequence, typically prepended with a special, learnable [CLS] token that serves to aggregate global information, is then fed into a standard Transformer Encoder which multi-head self-attention mechanism to capture long-range dependencies and global relationships between all image patches simultaneously, producing a final [CLS] token vector that acts as the highly expressive, scene-centric feature embedding for the entire image.

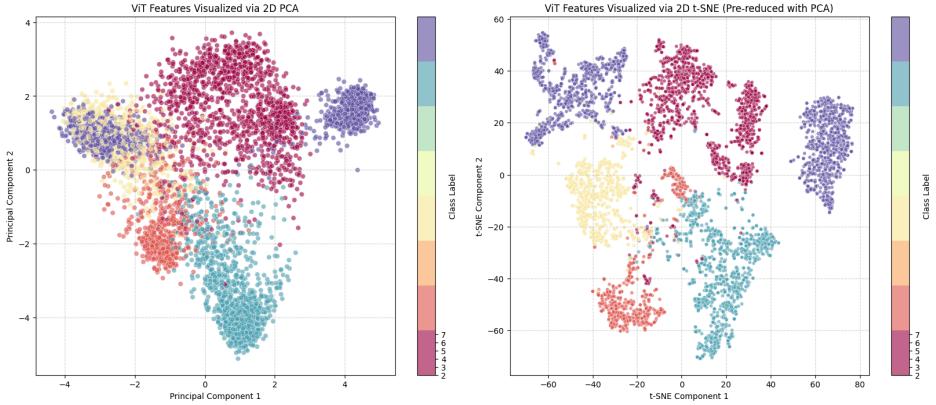

Figure 3.8. The PCA(left) and t-SNE(right) visualization demonstrates that ViT embeddings offer discriminatory capability for landmark classification

Pre-trained ResNet-50 Model Feature Embeddings

For complex feature extraction, we generate 2,048-dimensional scene-centric embeddings using a pre-trained version of the deep image encoder, ResNet-50 (Places365). This model was originally trained on a large-scale scene recognition corpus, Places365. The embeddings are pooled through Global Average Pooling, which aggregates spatial and texture activations into a single vector per image. These 2,048-dimensional embeddings capture high-level signals relevant to architectural composition, materials and surface textures, and overall spatial structure. Unlike handcrafted representations, this method avoids manually constructing edge maps or histogram bins. Instead, the network's convolutional filters in the late layers encode global shape and layout patterns that help differentiate built and natural landmarks, while early layers contribute pooled sensitivity to texture and material cues.

The first two images illustrated below produce nearly identical activation signatures across embedding dimensions, implying shared scene semantics in latent space. The third image activates different embedding indices, showing that the ResNet-50 Places365 encoder extracts globally discriminative and semantically meaningful features, supporting clear separation between similar vs distinct scenes for landmark classification.

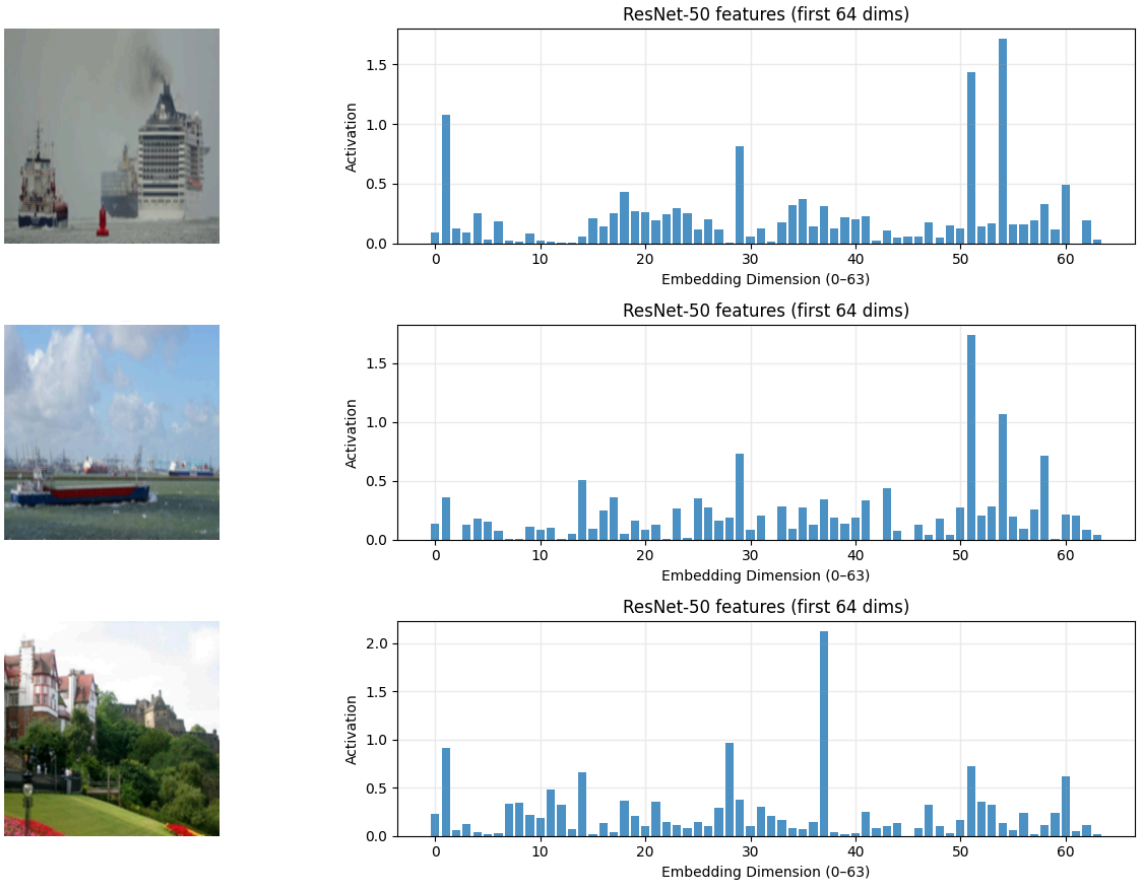

Figure 3.9.The visualization shows the activation values of the first 64 embedding dimensions extracted from ResNet-50 for three sample images.

To provide visual validation that these features are meaningful for multi-class separability, we applied 2-D PCA. The first two principal components explain 26.18% of total variance (PC1: 15.38%, PC2: 10.80%), reflecting that signal is distributed across multiple correlated embedding dimensions rather than dominated by a single trivial axis. The scatter plot reveals a partial but visible grouping structure of similar landscapes (canyons and waterfalls) versus man-made structures (towers and fortresses) in different zones of the projected space.

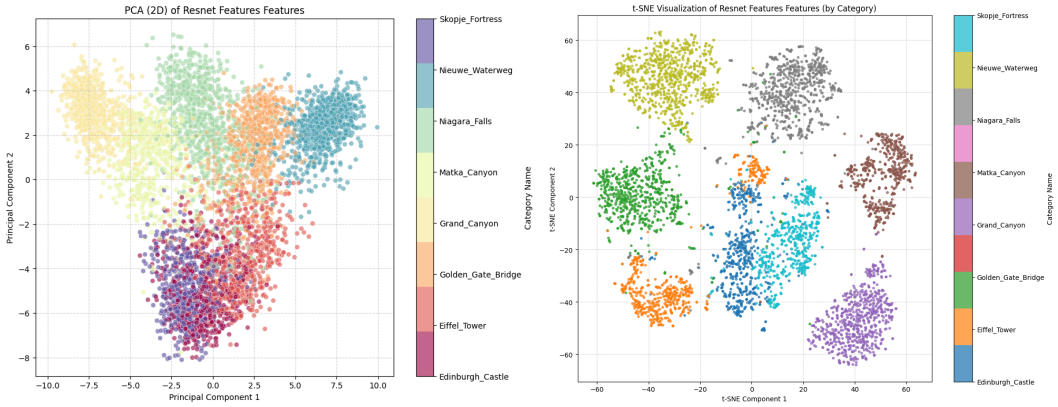

Figure 3.10. The PCA(left) and t-SNE(right) visualization demonstrates that ResNet embeddings offer discriminatory capability for landmark classification

4. Principal Component Analysis (PCA)

Our features had high-dimensional data which could cause challenges when running our classifications models including slow model training, overfitting and inefficiencies in compute and memory during training. Therefore, we used PCA to reduce dimensionality while keeping as much information (variance) as possible. We used a 90% threshold because it is a typical threshold used in machine learning research because it keeps enough information to preserve structure, removes noise and reduces dimensionality for faster models.

The figure below shows the cumulative explained variance curves for all feature types under a unified PCA procedure.

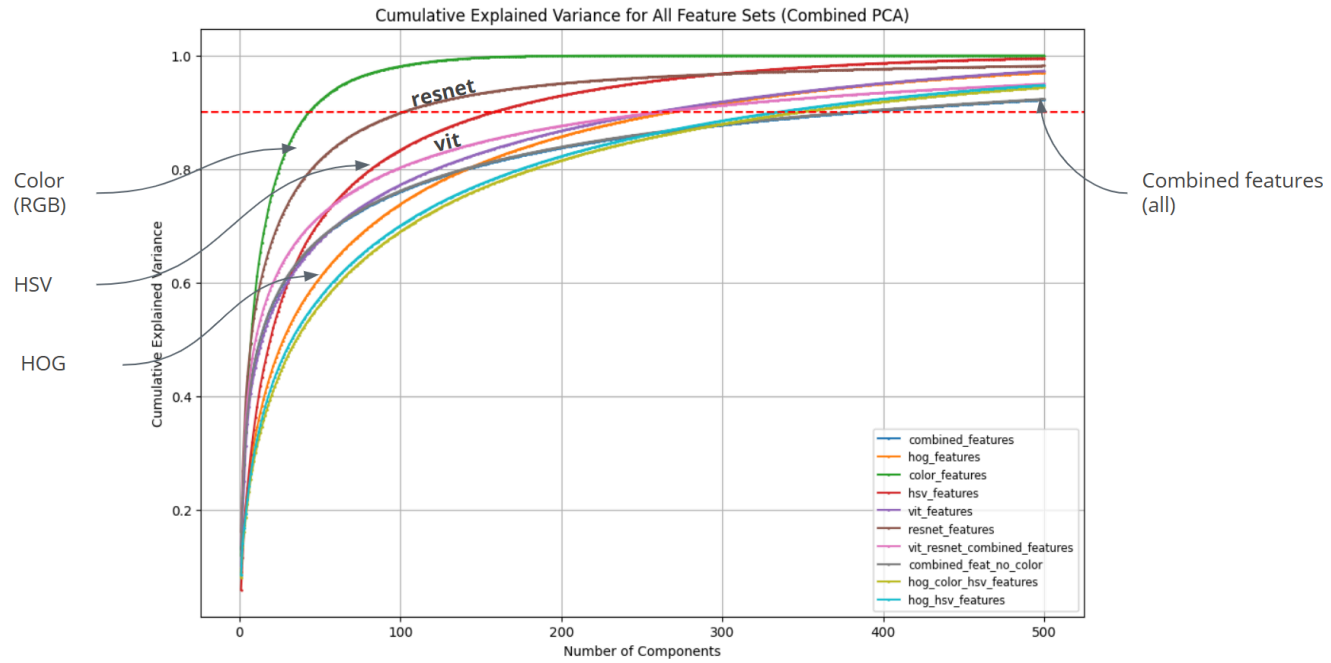

Figure 4.1. Cumulative explained variance curves for all feature sets under PCA. Color and HSV features show rapid variance accumulation due to their low dimensionality and redundancy. HOG features increase more gradually, reflecting higher intrinsic dimensionality. Deep learning embeddings (ResNet, ViT) accumulate variance efficiently due to their compact representational structure. The combined ViT+ResNet embedding exhibits intermediate behavior, indicating partial redundancy between the two models. The combined feature set with and without color displays nearly identical curves, suggesting minimal contribution of color features to overall variance when integrated with texture and deep embeddings.

Color and HSV features show a rapid rise, reaching close to 100% variance with relatively less than 200 and 300 principal components respectively. This is expected, as color information is low-dimensional and highly redundant, making it easy for PCA to compress. HOG features increase more slowly because they capture texture patterns, which introduce higher dimensionality and less redundancy.

ResNet and ViT embeddings show moderately fast increases with over 500 components explaining 90% variance, reflecting that deep features are already compact and optimized representations, though still containing multiple independent directions of variance. However, when combining two deep embeddings (ViT + ResNet), the curve is positioned between the two individual embeddings, suggesting some redundancy due to overlapping information.

To assess the contribution of color features, a combined feature set excluding color was also analyzed. Its cumulative variance curve closely matched that of the full combined set, indicating that color features contribute minimally to overall variance in the combined feature representation.

5. Methodology

To rigorously evaluate the effectiveness of diverse visual feature representations for landmark classification, we implemented a comparative modeling framework incorporating four supervised learning algorithms: Logistic Regression, K-Nearest Neighbours (KNN), Support Vector Machines (SVM), and a ResNet-based Convolutional Neural Network (CNN). Each model was independently trained on distinct feature sets, and hyperparameters were systematically tuned using model-appropriate strategies to ensure optimal performance using grid search for classical models and learning-rate/architecture tuning for the ResNet CNN. Final model configurations were evaluated on the test dataset, and training/inference time recorded for each model.

Model performance was assessed using chosen classification metrics, that is, accuracy, precision, and F1-score to capture both predictive reliability and robustness across classes. Additionally, we computed the Area Under the Curve (AUC) to enable threshold-independent comparison across models and feature groups, particularly useful in assessing discriminative power.

Feature representations were organized into hierarchical categories reflecting their conceptual and computational complexity for effective experimentation:

1. Simple Features: These are the classical, low-level descriptors that capture basic texture and color information
    - Histogram of Oriented Gradients (HOG)(hog_features_pca)
    - RGB color histograms(color_features_pca)
    - HSV color histograms (hsv_features_pca)
2. Combined Simple Features: To assess whether complementary low-level cues improve separability, we concatenated feature vectors:
    - HOG + RGB + HSV (hog_color_hsv_features_pca)
    - HOG + HSV (hog_hsv_features_pca)
3. Complex Features: These high-dimensional, semantically rich embeddings were extracted from pretrained deep vision architectures:
    - ResNet embeddings (resnet_features_pca)
    - Vision Transformer (ViT) embeddings (vit_features_pca)
4. Combined Complex Features: To leverage complementary representational strengths:
    - ViT + ResNet embeddings
5. Fully Combined Feature Sets: To explore the full representational spectrum:
    - Simple + Complex features (combined_features_pca)
    - Combined features excluding RGB color histograms (combined_feat_no_color_pca)

Across all feature groups, dimensionality reduction using Principal Component Analysis (PCA) ensured tractability and mitigated overfitting while preserving discriminative structure. This systematic, multi-model, multi-feature evaluation enables a robust comparison of representational efficacy for landmark classification in varied visual conditions.

6. Results

Logistic Regression

Logistic regression is well suited for landmark classification because it performs strongly when paired with informative feature embeddings, such as those derived from ViT and ResNet50. Its simplicity, computational efficiency, and regularization capabilities allow it to generalize well even in high-dimensional feature spaces. Additionally, it provides stable multiclass performance and interpretable probabilistic outputs, making it an ideal baseline model for evaluating the effectiveness of different handcrafted and deep feature representations in landmark classification.However, techniques like one-vs-rest introduce dependencies which can lead to inconsistent decision boundaries in multi-class landmark tasks thus its effectiveness depends entirely on the quality of handcrafted or pretrained features, making them inherently less flexible

We optimized logistic regression performance across feature sets using grid search over key hyperparameters: regularization strength (C), maximum iterations (max_iter), penalty type (l1 or l2), and solver (saga or liblinear). C balances model fit and regularization, while l1 promotes sparsity and l2 improves stability for correlated features; max_iter was set to 1000 to ensure convergence. Grid search systematically evaluated all combinations, identifying the configuration that yielded robust and generalizable performance across diverse feature extraction methods (Appendix A).

The performance across the different feature categories demonstrates clear distinctions in the classifier fit. Simple features such as HOG, RGB, and HSV achieved only modest accuracy between 58-62%, reflecting their limited ability to capture the full variability and complexity of the dataset. When these simple features were combined, performance improved to over 69%, indicating that texture and color information can complement each other; however, mild overfitting was observed in the test dataset. The precision, recall and f1-score metrics followed similar trends as accuracy in all feature combinations with a 0% or 1% variation(Appendix A) showing the dataset was well balanced.

	Simple Features			Combined Simple Features		Complex Features		Combined Complex Features	Combined features (Simple & Complex)	
Train / Test Results	hog_features_pca	color_features_pca	hsv_features_pca	hog_color_hsv_features_pca	hog_hsv_features_pca	vit_features_pca	resnet_features_pca	vit_resnet_combined_features_pca	combined_features_pca	combined_feat_no_color_pca
Train	64.70%	57.12%	61.52%	75.66%	73.78%	97.66%	98.03%	98.97%	98.88%	99.16%
Test	61.99%	56.01%	58.81%	72.07%	69.82%	97.75%	96.51%	98.14%	98.37%	98.37%
Train Test Gap	2.71%	1.11%	2.71%	3.59%	3.96%	-0.09%	1.52%	0.83%	0.51%	0.79%

Table 5.1. Representation of train test gap on accuracy results for overfitting analysis

Deep-learning features from pretrained models including ResNet and ViT embeddings yielded high performance with accuracy and precision above 96% (Appendix A). These features provided highly discriminative representations, with the train test-gap remaining extremely small consistently below 1%. In contrast, the simple features demonstrated signs of overfitting with test/train differences of HOG, Color, HSV features at 3.59%, 2.71%, and 3.96% respectively. The combined features and advanced features showed resilience with

less than 2% difference in train test accuracy values. The Classification report for the combined features results which has both simple and complex features indicates all individual classes have precision, recall, and F1-scores above 0.95, with most above 0.97. This shows the model is both highly accurate in its positive predictions (precision) and correctly identifies nearly all instances of each class (recall). However, classes like Skopje_Fortress and Eiffel_Tower have slightly lower recall (~0.954 – 0.959) compared to others, suggesting the model occasionally confuses these with other classes that share similar patterns.

Class	Precision	Recall	F1-Score	Support
Niagara_Falls	0.975	0.968	0.971	158
Matka_Canyon	0.964	1	0.982	135
Eiffel_Tower	0.986	0.959	0.972	147
Golden_Gate_Bridge	0.982	1	0.991	165
Skopje_Fortress	0.992	0.955	0.973	133
Edinburgh_Castle	0.97	0.976	0.973	165
Nieuwe_Waterweg	0.995	1	0.998	208
Grand_Canyon	1	1	1	178
Accuracy			0.984	1289
Macro Avg	0.983	0.982	0.983	1289
Weighted Avg	0.984	0.984	0.984	1289

*Table 5.2. Representation of the results of classification per category for all combined features (*combined_features_pca*)*

The ROC/AUC Curves for combined simple features (Macro-average AUC: 0.9355) and combined complex features(Macro-average AUC: 0.9997) show that the combination of the complex features have the ability to distinguish between classes across various thresholds as opposed to the simple features which have multiple false positives across the classes. Combining both simple and complex features further improves performance, achieving a Macro-average AUC of 0.9998, indicative of near-perfect discriminative ability. This is further supported by the confusion matrix which reveals the higher number of misclassified instances when simple features are employed, further highlighting the enhanced predictive power of the complex feature set.

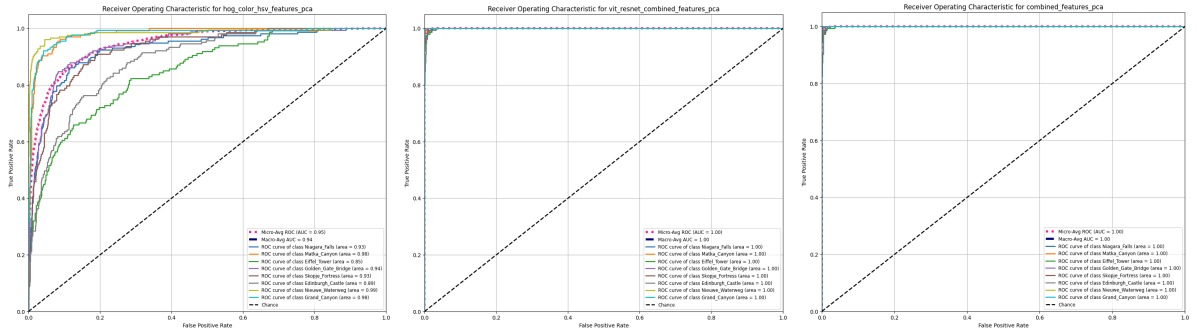

Figure 5.3 The ROC Curves for combined simple features (left), combined complex features (middle) and combined simple & complex features (right) illustrates that complex features outperform simple features in every combination

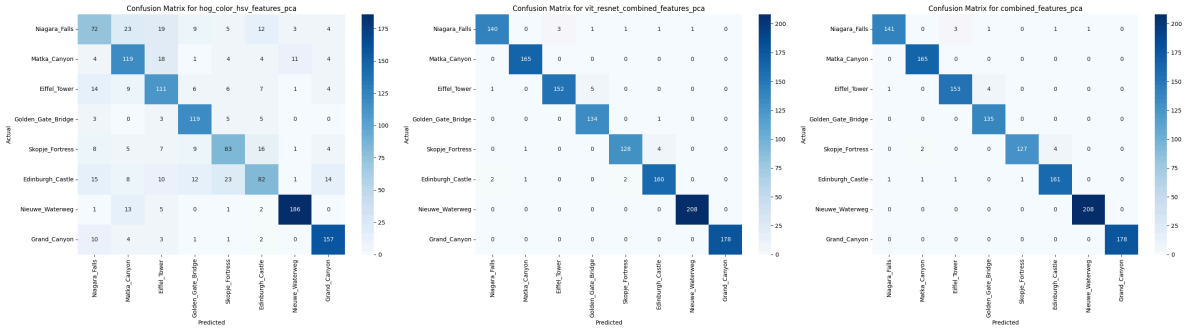

Figure 5.4. The confusion matrix for combined simple features (left), combined complex features (middle) and combined simple & complex features (right)

K - Nearest Neighbor

K-Nearest Neighbors (KNN) is a simple, non-parametric algorithm for classification and regression that assigns a label to a new sample based on the majority class of its K nearest neighbors in feature space. We chose KNN because it relies on distance-based similarity rather than learning an explicit decision boundary, making it well-suited for evaluating how different feature representations such as HOG, color histograms, or deep embeddings cluster semantically similar images. The algorithm relies heavily on distance metrics, making it vulnerable to noisy features, irrelevant dimensions, or poor normalization hence its effectiveness depends on the quality and separability of the features, as well-separated clusters lead to more accurate predictions. Additionally, KNN provides an intuitive measure of similarity between images, allowing us to directly observe how feature embeddings capture landmark-specific patterns.

The model was parameter-tuned by varying the distance metric, weighting scheme, and Minkowski power parameter, while keeping the number of neighbors fixed at eight. Although KNN is a “lazy learning algorithm” the fit method was used to store the training data and organize it for efficient nearest-neighbor searches. Each hyperparameter combination was evaluated on training and test sets to identify the configuration that maximized accuracy while considering computational efficiency. The analysis of the model’s performance across 6 different image feature extraction methods shows a clear dichotomy in results. The highest performance was achieved by the deep learning-derived features, with ViT taking the lead with a test accuracy of 97.75% and an F1 Score of 0.98, closely followed by ResNet at 96.51% test accuracy and 0.96 F1-Score. In stark contrast, the simpler features such as HOG, Color Histograms, and HSV yielded significantly lower performance metrics, with HSV showing the lowest test accuracy at 65.93% and an F1 Score of 0.6616 (Appendix B).

	Simple Features			Combined Simple Features		Complex Features		Combined Complex Features	Combined features (Simple & Complex)	
Train / Test Results	hog_features_pca	hsv_features_pca	color_features_pca	hog_color_hsv_features_pca	hog_hsv_features_pca	vit_features_pca	resnet_features_pca	vit_resnet_combined_features_pca	combined_features_pca	combined_feat_no_color_pca
Train	0.7198	0.6593	0.6979	0.7671	0.75	0.9766	0.9788	0.9858	0.9846	0.9848
Test	0.7198	0.6593	0.6979	0.7671	0.75	0.9766	0.9766	0.9858	0.9846	0.9848
Train Test Gap	0	0	0	0	0	0	0.0022	0	0	0

Table 5.3. Representation of KNN's train test gap on accuracy results

Additionally, the deep learning features did not accrue significant computational time, with ViT performing faster than HSV and HOG. We observe that the combined deep learning features - ViT and ResNet - perform best with an accuracy of 98.58% and an F1-Score of 0.9858. With only the simpler features, we see a significant drop in performance with the best performing being all three for an accuracy of 76.71% and an F1-Score of 0.766. Interestingly, including the simpler features drops performance by approximately 0.1% on accuracy. Ultimately, the data indicates that leveraging deep learning features provides the best predictive accuracy and precision with no significant drawback. As shown in the table below, performance remains robust even for classes with substantial variation in appearance, such as the Eiffel Tower and Golden Gate Bridge, each achieving F1-scores of 97% - 98%. Notably, classes with larger support values, such as Niagara Falls and Grand Canyon, exhibit near-perfect precision and recall, suggesting that the model scales effectively with larger sample sizes.

Class Label	Precision	Recall	F1-Score	Support
0 Grand_Canyon	0.99	0.99	0.99	786
1 Matka_Canyon	1.00	1.00	1.00	528
2 Eiffel_Tower	0.98	0.98	0.98	584
3 Edinburgh_Castle	1.00	1.00	0.98	723
4 Skopje_Fortress	0.96	0.96	0.98	571
5 Golden_Gate_Bridge	0.97	0.97	0.97	569
6 Niagara_Falls	0.99	0.99	0.99	865
7 Nieuwe_Waterweg	1.00	1.00	1.00	710

Table 5.4. Representation of the results of classification per category for all combined features

The ROC/AUC Curves for combined complex features have a high macro-average AUC of 100% as evidence that the combination of the complex features have perfect discrimination capability across all the classes. However, the corresponding confusion matrix reveals a small number of misclassifications, underscoring that despite the model's exceptionally strong overall performance, it does not achieve flawless classification for every class.

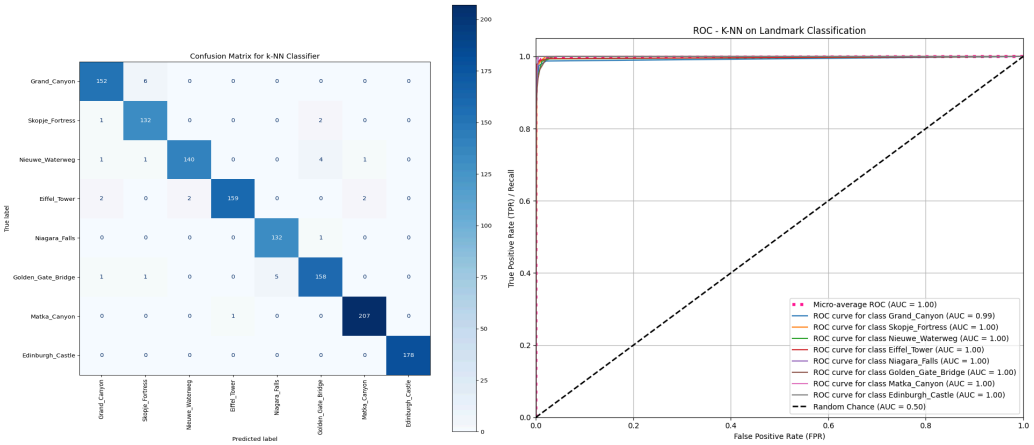

Figure 5.5. KNN's confusion matrix and ROC Curves for combined complex features

Support Vector Machines (SVM)

SVMs work well when features are sparse as provided by HOG, dense (e.g. CNN or ViT embeddings) and nonlinearly separable which is common in real-world images such as landmarks. However, while SVMs can work well with deep features, extremely high-dimensional inputs may lead to longer training times and potential overfitting without proper regularization.

To identify the optimal configuration for the SVM classifier, we conducted an extensive hyperparameter search using a Grid Search strategy with cross-validation. The search systematically explored combinations of the regularization parameter C with values 0.1,1, and 10 to assess the trade-off between margin maximization and misclassification tolerance; along with a range of kernel functions ('rbf', 'linear') and gamma ('scale','auto') to evaluate how each setting affected the model's generalization performance. The grid search revealed that the most effective configuration for the dataset was C = 10, kernel = 'rbf', and gamma = 'scale'. This parameter combination consistently provided the highest cross-validated performance across folds. This indicates that a moderately strong regularization strength was necessary to balance the influence of misclassified samples while still preserving a sufficiently wide margin.The gamma = 'scale' setting further allowed the model to adaptively adjust kernel width based on the variance of the input features, improving stability and reducing the risk of overfitting.

The SVM results below demonstrate the efficiency–accuracy trade-off clearly across different feature representations. Feature sets incorporating pretrained deep models achieve the highest classification accuracy, with the combined and combined_feat_no_color configurations reaching an accuracy of approximately 98.6%. However, these high-performing models come with moderate computational costs, particularly during training, due to the increased complexity of the learned decision boundary. In contrast, ResNet-only features provide an attractive balance between performance and efficiency: although accuracy decreases slightly to 98%, training and inference times are significantly reduced to 2 seconds and 0.3 seconds respectively (up to 9x faster), making this configuration well-suited for scenarios where computational efficiency or real-time inference is important (Appendix C).

	Simple Features			Combined Simple Features		Complex Features		Combined Complex Features	Combined features (Simple & Complex)	
Train / Test Results	hog_features_pca	hsv_features_pca	color_features_pca	hog_color_hsv_features_pca	hog_hsv_features_pca	vit_features_pca	resnet_features_pca	vit_resnet_combined_features_pca	combined_features_pca	combined_feat_no_color_pca
Train	0.74	0.68	0.67	0.82	0.79	0.99	0.98	0.99	0.99	0.99
Test	0.72	0.66	0.63	0.80	0.78	0.99	0.98	0.99	0.99	0.99
Train Test Gap	0.02	0.02	0.04	0.02	0.01	0	0	0	0	0

Table 5.5. Representation of train test gap on accuracy results for SVM

On the other end of the spectrum, models relying solely on hand-crafted features such as HOG, HSV, and color histograms exhibit substantially lower accuracy while incurring the highest training and inference times. Despite similar feature dimensionality after PCA, these features result in poor separability in feature space, causing the SVM to rely on a large number of support vectors and leading to increased computational cost. This combination of lower accuracy and higher runtime makes traditional feature-only models inefficient for this task. Overall, these experiments demonstrate that deep, pretrained features offer the most favorable efficiency–accuracy trade-off, while hand-crafted features may only be justified in extremely resource-constrained settings. Using the all combined feature, the individual class report below shows that the SVM classifier demonstrates consistently strong performance across all landmark classes, with precision, recall, and F1-scores mostly between 0.97 and 1.00. Classes such as Grand Canyon and Nieuwe Waterweg achieved perfect scores, indicating no misclassifications, while slightly lower but still high scores for Edinburgh Castle and Skopje Fortress suggest only minimal confusion between visually similar categories. Overall, the results show that the SVM model effectively discriminates between all landmarks with high reliability and generalization across varying sample sizes.

Class Label	Precision	Recall	F1-Score	Support
0 Grand_Canyon	1.00	1.00	1.00	178
1 Matka_Canyon	0.97	0.99	0.98	135
2 Eiffel_Tower	0.99	0.98	0.99	147
3 Edinburgh_Castle	0.98	0.96	0.97	165
4 Skopje_Fortress	0.98	0.97	0.97	133
5 Golden_Gate_Bridge	0.99	0.99	0.99	165
6 Niagara_Falls	0.97	0.98	0.98	158
7 Nieuwe_Waterweg	1.00	1.00	1.00	208

Table 5.6. Representation of the results of classification per category for all combined features

The ROC curves show near 100% scores with a Macro-average AUC that averages 99.98% across all combinations of complex features, while the combination of simple features stands

at 96.38%. The confusion matrix for combined simple features (left), combined complex features (middle) and combined simple & complex features (right) show progressive increase in classifications across all landmark classes. The simple feature combinations struggled with all classes while the infusion of complex features dramatically increased the classification accuracy. Overall, the large gap in AUC highlights the superior quality and separability provided by complex deep-learning-based features.

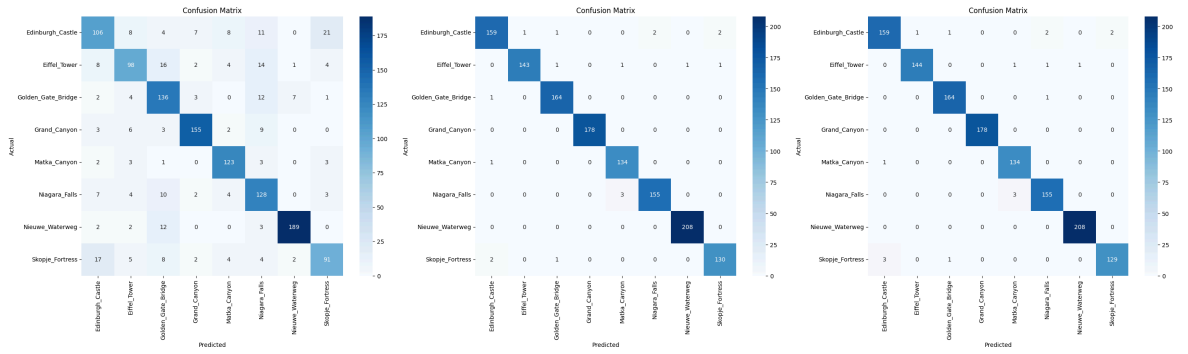

Figure 5.6. The confusion matrix for combined simple features (left), combined complex features (middle) and combined simple & complex features (right)

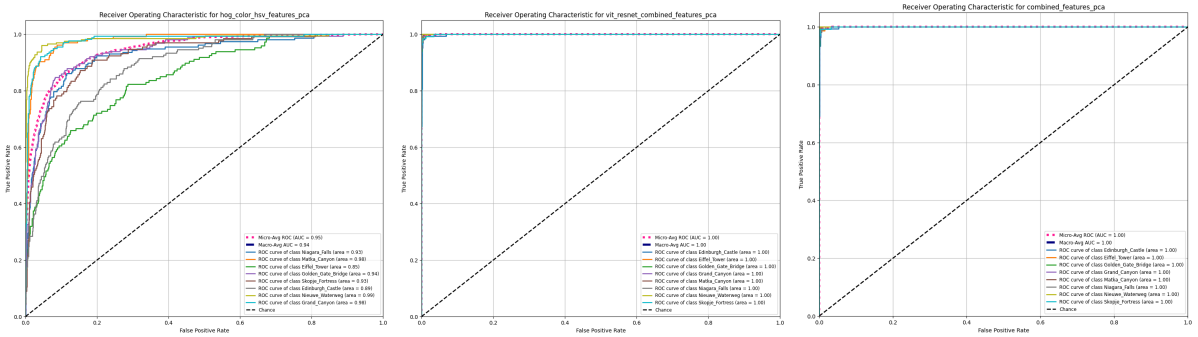

Figure 5.7 The ROC Curves for combined simple features (left), combined complex features (middle) and combined simple & complex features (right) illustrates that complex features outperform simple features in every combination

ResNet-based Convolutional Neural Network (CNN)

For the neural classification component, we trained a lightweight CNN head on the frozen 2048-D scene embeddings from the Places365-pretrained ResNet-50 encoder. All learning occurred in the head, which accepted inputs reshaped to [N, 1, 2048]. The design used two 1-D convolutions (kernel size 3, 64 channels, ReLU) to learn non-linear feature interactions, followed by 30% dropout, Global Average Pooling, and fully connected projections (64 - 128 num_classes) optimized using AdamW at a learning rate of 1e-3 over 20 epochs.

The experiment was restricted to the PCA-ResNet feature set and did not incorporate any of the alternative feature representations as with the other previous models. This is because the model failed to optimize effectively when trained on PCA-ResNet features and improved only marginally per epoch because training was initially implemented using full-batch gradient updates. This optimization setup caused the loss to plateau near random assignment. Testing accuracy remained 22.66%, and performance collapse was evident in the report. Multiple landmarks never formed a usable decision surface and produced F1

scores of zero. Other classes showed inflated recall but near-zero precision, which shows that the network repeatedly predicted a small subset of labels instead of learning full multi-class separation.

Label	precision	recall	f1-score	support
0 Grand_Canyon	0.00	0.00	0.00	157
1 Matka_Canyon	0.07	0.02	0.03	106
2 Eiffel_Tower	0.17	0.80	0.28	117
3 Edinburgh_Castle	0.29	0.71	0.41	145
4 Skopje_Fortress	0.31	0.33	0.32	114
5 Golden_Gate_Bridge	0.00	0.00	0.00	114
6 Niagara_Falls	0.00	0.00	0.00	173
7 Nieuwe_Waterweg	1.00	0.04	0.07	142

Table 5.7. Representation of the results of classification per category for ResNet features

The confusion matrix exhibited off-diagonal concentration and confirmed that most samples were misclassified into the same few predicted buckets. ROC analysis showed only one class with strong separability (AUC 0.95), while the rest stayed far below 0.84. The CNN never reached a converged state capable of shaping meaningful non-linear decision boundaries on the embedding representation.

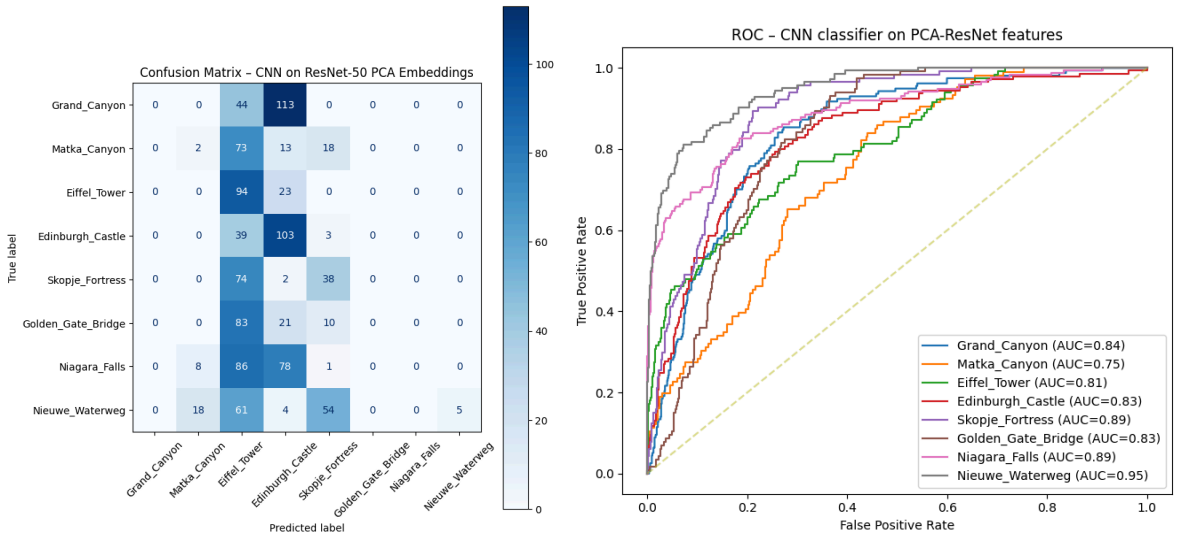

Figure 5.8. The confusion matrix and ROC Curves for ResNet features

The extracted features themselves were highly distinctive as shown by tight clustering in PCA space. Classical ML classifiers also achieved over 95% accuracy on the same 2048-D embeddings. The CNN head underperformed because convolutional architectures impose structural assumptions, including locality and translation-equivariance along the feature axis. These built-in biases are useful for structured signals like pixel grids, but not fit for globally average-pooled embeddings where dimension index does not correspond to an

interchangeable position. Because these embeddings lack spatial locality or meaningful 1-D ordering, weight sharing across convolutional kernels introduced a harmful built-in bias. It reduces optimization stability and prevents the network from forming expressive decision surfaces. This experiment serves as a meaningful negative result and demonstrates that pretrained global scene embeddings are better aligned with classical or MLP classifier heads unless spatial or sequential structure is preserved.

7. Model Evaluation

The different models were hyper parameter tuned and applied to the combined simple and complex features (`combined_features_pca)` to provide best configurations for the classification models. The following are the results:

Model	**Parameters**	**Search Values**	**Best Value**	**Accuracy**	**AUC/ROC**	**Training Time in seconds(s)**
Logistic Regression	C	[1,0.1]	1	0.9837	0.998	290.4242
	max_iter	[1000, 2000]	1000			
	penalty	['l1', 'l2']	l2			
	solver	['liblinear', 'saga']	saga			
K-NN	Weight	['uniform', 'distance']	uniform	0.9858	1.0	0.02s
	Metric	['euclidean', 'manhattan']	euclidean			
	P	[1, 2]	1			
SVM	C	[0.1,1,10]	10	0.9860	1.0	4.134235
	Gamma	['scale','auto']	scale			
	Kernel	:['rbf', 'linear']	rbf			
CNN	Learning_rate, Dropout, Kernel_size, Epochs	N/A	N/A	0.2266	N/A	5.40s

Table 5.13. Representation of the tuned hyperparameter and results of each model

Logistic regression provides high accuracy at 98.37% but takes a long time to train compared to the other models. K-NN has higher accuracy measures at 98.58% and runs extremely quickly at 0.02s. Good for quick predictions. SVM has marginally higher accuracy compared to KNN, however, with much longer training time. CNN performs poorly on this dataset, possibly because the data format and compression of spatial data hence it was not suitable for the experiment. Logistic Regression, KNN, and SVM demonstrated high AUC scores, demonstrating their strong ability to discriminate between landmark classes across the different feature representations.

8. Discussions

Our results demonstrate the significance of the discriminative power of feature representations for image classification. Simple features such as HOG, RGB, and HSV achieved only moderate accuracy levels, ranging from 56% to 75% across Logistic Regression, KNN, and SVM models. In comparison, complex features extracted from pretrained deep networks consistently surpassed 97% accuracy, illustrating the superior representational capacity of deep embeddings. Interestingly, combining simple and complex features resulted in a modest improvement of approximately 1%, pushing accuracy above 98%. This suggests that simple features provide complementary information that enhances the discriminative capacity of deep features, albeit marginally. Specifically, the addition of RGB color features consistently improved separability over and above HSV_HOG feature combinations across all classical models, contributing to a 2–4% increase in both accuracy and precision suggesting that the raw RGB representation preserves critical chromatic details that are partially lost in HSV transformations, leading to improved class separability.

The performance gains observed with complex embeddings highlight a crucial insight: the choice of feature representation exerts a greater influence on classification performance than the choice of classifier. Lightweight classifiers, including Logistic Regression, KNN, and SVM, achieved excellent results when fed high-quality deep features, emphasizing the effectiveness of leveraging pretrained networks without necessitating heavy model architectures. Among the complex feature sets, combinations such as combined_features_pca and vit_resnet_combined_features_pca consistently delivered the best performance across multiple metrics accuracy, precision, recall, F1-score, and AUC while simple features lagged significantly.

An additional consideration is computational efficiency. Surprisingly, simple features like HOG, RGB, and HSV required longer training times despite having less dense information than individual complex embeddings. While combined complex features slightly increased computation time compared to single embeddings, the marginal gains in performance do not always justify this additional cost, suggesting that using individual deep features may represent the most practical tradeoff between accuracy and efficiency (Appendix A, B, C).

The performance of a CNN trained on ResNet-50 embeddings provided further insights into model-feature alignment. Despite the embeddings being highly informative, the CNN performed poorly, producing collapsed predictions with lower accuracy. This outcome can be attributed to a mismatch between the model's architectural assumptions and the feature structure: convolutional layers rely on spatial or sequential relationships, which are absent in globally pooled embeddings. Consequently, CNN could not extract additional discriminative patterns. This result underscores the importance of aligning model architecture with feature characteristics; classical methods proved better suited for classification when using pretrained deep embeddings.

Overall, our findings reinforce the central role of feature quality in image classification. Pretrained deep networks offer robust and highly discriminative representations that can significantly enhance even simple classifiers, while simple features play a supportive, complementary role. Furthermore, aligning model assumptions with feature structure is

essential for maximizing performance, as demonstrated by the contrasting behavior of the CNN and classical classifiers.

A key limitation of this study is its reliance on a limited set of feature types, particularly the use of specific pretrained embeddings (ResNet-50 and ViT) and a narrow selection of simple features, which may not fully represent the broader landscape of classical or learned descriptors. The experiments did not explore end-to-end deep learning models or fine-tuning, restricting our ability to assess how feature representations might improve under task-specific optimization. Additionally, robustness to real-world variations such as lighting, occlusion, or domain shifts was not evaluated, limiting generalizability beyond the curated dataset. Finally, the CNN analysis was conducted only on PCA-reduced ResNet embeddings, which constrains conclusions about CNN performance on alternative feature structures or raw image data.

9. Generalizability

Model selection and generalizability was conducted after the hyperparameter tuning process conducted exclusively on the validation set. This hyperparameter process was conducted on a separate validation set to prevent overfitting and select configurations optimized for generalizable performance. We ultimately achieved high generalizability with all classical models leveraging deep-learning features, confirming that the quality of the ViT and ResNet embeddings was the primary driver for successful learning and generalization.

The success of this feature-based approach is quantified by the results on the held-out test set. Both the Logistic Regression (Test Accuracy: 98.37%) and K-NN (Test Accuracy: 98.58%) models demonstrated exceptional generalization, maintaining a train-test accuracy gap consistently below 1%. This minimal difference is strong evidence that these models learned highly generalizable decision boundaries in the feature space, effectively avoiding overfitting to the training data. Conversely, the CNN demonstrated a profound failure to generalize, achieving a test accuracy of only 22.66%. This failure was attributed not to the feature quality, but to the model's architectural misalignment; its convolutional layers could not learn meaningful patterns from the non-spatial, compressed ResNet embeddings, leading to catastrophic failure on unseen test data.

While the performance on the benchmark was high, the external generalizability of the overall system remains limited by the narrow scope of the study, which was restricted to eight specific landmark categories. To substantially improve the training process for real-world application, two key areas must be addressed. First, the current process must be extended to validate robustness against real-world noise -including image obstruction, extreme lighting, and novel camera viewpoints - which means integrating more sophisticated data augmentation or specialized adversarial training into the feature-based pipeline. Second, scaling the model to large-scale datasets (like the full Google Landmarks dataset) introduces significant challenges such as extreme class imbalance; overcoming this requires adopting advanced training strategies like fine-tuning the deep feature encoders or implementing hierarchical classification to manage the vast label space.

10. Accuracy vs Efficiency

The results across three feature vectors (Simple, Deep, and Combined) and the classifiers established a clear trade-off between accuracy and computational cost (training and inference time). The accuracy-optimizated solution, achieved by the SVM classifier with the Combined Feature Vector, delivered the highest overall accuracy (e.g., 98.6%) but this high performance came at the expense of a longer training duration and a higher computational burden. Conversely, the efficiency-optimized solution - the K-NN classifier with the Deep Feature Vector - offered a superior computational profile, reducing training time to approximately > 1 second (where SVM ranged from 3 seconds to 14 seconds) while maintaining near-maximal performance (only a marginal 0.05% to 1.0% loss in accuracy). Additionally, we see K-NN inference time significantly outperformed SVM - where SVM consistently performed within 2-3 seconds while K-NN was always > 1 seconds. This relative trade-off illustrates a diminishing return: the small incremental gain in accuracy provided by the most resource-intensive combination is disproportionate to its substantial computational cost, confirming the K-NN + Deep Feature configuration as the most practical choice for deployment where speed and efficiency are critical.

11. Conclusion

This project demonstrates that feature representation is the dominant factor in successful landmark image classification, outweighing the choice of classifier itself. Across Logistic Regression, SVM, and K-Nearest Neighbors, models leveraging pretrained deep feature embeddings from ViT and ResNet consistently achieved near-perfect generalization, with test accuracies approaching 99% and negligible train–test gaps. These embeddings provide compact yet semantically rich representations that remain robust to variations in viewpoint, lighting, and scale, enabling even simple classifiers to perform exceptionally well. In contrast, traditional handcrafted features such as HOG, HSV, and color histograms consistently underperformed, confirming that low-level visual cues alone are insufficient for capturing the complexity of real-world landmark imagery. From an efficiency perspective, although the SVM achieved the highest absolute accuracy, its inference cost was notably higher due to its complex decision boundary. The KNN model, when paired with deep embeddings, delivered virtually identical accuracy while offering significantly faster inference, making it the most practical solution for real-world deployment. Overall, our results highlight that modern deep representations paired with appropriately matched classifiers enable both high accuracy and efficient inference, validating the effectiveness of transfer learning for landmark recognition tasks.

12. Future Work

While the models demonstrated excellent performance within the scope of this project, a key limitation lies in the scale of classification. The original dataset contains over 200,000 labeled landmark categories, whereas this study focused on only eight carefully selected classes. A natural and impactful extension of this work would be to scale the system to handle the full dataset, transforming the problem into a large-scale, fine-grained landmark classification task. This would introduce new challenges related to class imbalance, visual similarity between landmarks, and computational scalability. Additionally, future work should evaluate model robustness under real-world conditions such as occlusion, extreme lighting,

and novel viewpoints. Further improvements could be achieved by fine-tuning pretrained models for landmark-specific representations and optimizing inference through approximate nearest neighbor methods or lightweight architectures, enabling efficient and scalable real-world deployment.

13. References

CSAILVision. (2025). GitHub - CSAILVision/places365: The Places365-CNNs for Scene Classification. GitHub. https://github.com/CSAILVision/places365

Lin, Y., Cai, Y., Gong, Y., Kang, M., & Li, L. (2019). Extracting urban landmarks from geographical datasets using a random forests classifier. International Journal of Geographical Information Science, 33(12), 2406–2423. https://doi.org/10.1080/13658816.2019.1620238

Tobias Weyand, Andre Araujo, Bingyi Cao, Jack Sim. (2020). Google Landmarks Dataset v2 -- A Large-Scale Benchmark for Instance-Level Recognition and Retrieval. arXiv:2004.01804

Melendez, P. [pemujo]. (2023). GLDv2_Top_51_Categories [Data set]. Hugging Face. https://huggingface.co/datasets/pemujo/GLDv2_Top_51_Categories

Geographic Landmark Based Visual Localization (2025), Github - hngondoki/geographic_landmark_based_visual_localization: W281 Project. Github. https://github.com/hngondoki/geographic_landmark_based_visual_localization

14. Authors Contribution

Bosen: EDA, Data Preprocessing, Simple Feature Extraction - HSV Histogram, SVM model/pipeline; Written -EDA, Data Preprocessing, HSV Histogram, SVM model/pipeline, Conclusion, Future Work
Rick: Simple Feature Extraction - HOG (including visuals), Complex Feature Extraction - ViT (including visuals), 2D PCA visualization, K-NN model implementation, hyperparameter-tuning, and evaluation; Written - Abstract, Introduction, HOG, VIT, K-NN results, Generalizability, Accuracy vs Efficiency Tradeoff.
Brandon: Complex feature extraction - ResNet-50, 2D PCA, t-SNE, extracted feature visualization; CNN model implementation, experiment, and evaluation; Written - Feature embeddings from ResNet-50, CNN, CNN results (visuals).
Hildah: RGB color feature engineering, conducted Principal Component Analysis (PCA), and implemented logistic regression models. Visualizations including PCN, t-SNE and Confusion matrices.; Written -Abstract, Introduction, Colour Histograms, Logistic Regression, Discussion. Additionally, project management, overseeing progress, reviewing work, and ensuring alignment of final delivery with rubric requirements.

15. Appendices

A. Logistic Regression - Train/Test Experiment Results

Features	Parameters	Train / Test Results	accuracy	precision	recall	f1-score	Training/ Inference time in sec
combined_features_pca	{'C': 1, 'max_iter': 1000, 'penalty': 'l2', 'solver': 'saga'}	Train	0.9888	0.99	0.99	0.99	290.4242
		Test	0.9837	0.98	0.98	0.98	0.0056
hog_features_pca	{'C': 0.1, 'max_iter': 1000, 'penalty': 'l2', 'solver': 'saga'}	Train	0.6470	0.63	0.63	0.63	33.6372
		Test	0.6199	0.60	0.61	0.60	0.0081
color_features_pca	{'C': 1, 'max_iter': 1000, 'penalty': 'l1', 'solver': 'saga'}	Train	0.5712	0.56	0.57	0.56	2.1382
		Test	0.5601	0.55	0.55	0.55	0.0023
hsv_features_pca	{'C': 1, 'max_iter': 1000, 'penalty': 'l2', 'solver': 'saga'}	Train	0.6152	0.61	0.62	0.61	7.1278
		Test	0.5881	0.58	0.58	0.58	0.0039
vit_features_pca	{'C': 1, 'max_iter': 1000, 'penalty': 'l2', 'solver': 'liblinear'}	Train	0.9766	0.99	0.99	0.99	63.9464
		Test	0.9775	0.98	0.98	0.98	0.0048
resnet_features_pca	{'C': 0.1, 'max_iter': 1000, 'penalty': 'l2', 'solver': 'saga'}	Train	0.9803	0.98	0.98	0.98	70.0831
		Test	0.9651	0.96	0.96	0.96	0.0036
vit_resnet_combined_features_pca	{'C': 1, 'max_iter': 1000, 'penalty': 'l2', 'solver': 'liblinear'}	Train	0.9897	0.99	0.99	0.99	193.9438
		Test	0.9814	0.98	0.98	0.98	0.0062
combined_feat_no_color_pca	{'C': 0.1, 'max_iter': 1000, 'penalty': 'l2', 'solver': 'saga'}	Train	0.9916	0.99	0.99	0.99	274.8424
		Test	0.9837	0.98	0.98	0.98	0.0069
hog_color_hsv_features_pca	{'C': 1, 'max_iter': 1000, 'penalty': 'l2', 'solver': 'liblinear'}	Train	0.7566	0.75	0.75	0.75	53.4027
		Test	0.7207	0.71	0.71	0.71	0.0061
hog_hsv_features_pca	{'C': 1, 'max_iter': 1000, 'penalty': 'l2', 'solver': 'liblinear'}	Train	0.7378	0.72	0.73	0.72	48.9584
		Test	0.6982	0.68	0.68	0.68	0.0069

B. K-NN - Train/Test Experiment Results

Features	Parameters	Train / Test Results	accuracy	precision	recall	f1-score	Training/ Inference time in sec
combined_features_pca	{weight: uniform, metric: manhattan, p: 1}	Train	0.9846	0.9848	0.9846	0.9846	0.72
		Test	0.9846	0.9848	0.9846	0.9846	0.73
hog_features_pca	{weight: uniform, metric: manhattan, p: 1}	Train	0.7198	0.7372	0.7198	0.7183	1
		Test	0.7198	0.7372	0.7198	0.7183	0.48
color_features_pca	{weight: uniform, metric: manhattan, p: 1}	Train	0.6979	0.718	0.6979	0.6976	0.03
		Test	0.6979	0.718	0.6979	0.6976	0.03

hsv_features_pca	{weight: uniform, metric: manhattan, p: 1}	Train	0.6593	0.7112	0.6593	0.6616	0.06
		Test	0.6593	0.7112	0.6593	0.6616	0.06
vit_features_pca	{weight: uniform, metric: manhattan, p: 1}	Train	0.9766	0.9768	0.9766	0.9765	0.04
		Test	0.9766	0.9768	0.9766	0.9765	0.04
resnet_features_pca	{weight: uniform, metric: euclidean, p: 1}	Train	0.9788	0.979	0.9788	0.9788	0.22
		Test	0.9766	0.9768	0.9766	0.9765	0.22
vit_resnet_combined_features_pca	{weight: uniform, metric: euclidean, p: 1}	Train	0.9858	0.9859	0.9858	0.9858	0.3
		Test	0.9858	0.9859	0.9858	0.9858	0.3
combined_feat_no_color_pca	{weight: uniform, metric: manhattan, p: 1}	Train	0.9848	0.985	0.9848	0.9848	0.7
		Test	0.9848	0.985	0.9848	0.9848	0.7
hog_color_hsv_features_pca	{weight: uniform, metric: manhattan, p: 1}	Train	0.7671	0.7761	0.7671	0.766	0.47
		Test	0.7671	0.7761	0.7671	0.766	0.47
hog_hsv_features_pca	{weight: uniform, metric: manhattan, p: 1}	Train	0.75	0.7613	0.75	0.7486	0.38
		Test	0.75	0.7613	0.75	0.7486	0.38

C. SVM - Train/Test Experiment Results

Features Used	Parameters		Accuracy	Precision	Recall	F1-Score	Training/ InferenceTime
combined_feat_no_color	{'C': 10, 'gamma': 'scale', 'kernel': 'rbf'}	Train	0.986036	0.986099	0.986036	0.986030	4.448490
		Test	0.99	0.99	0.98	0.99	0.772963
combined_features_pca	{'C': 10, 'gamma': 'scale', 'kernel': 'rbf'}	Train	0.986036	0.986063	0.986036	0.986018	4.134235
		Test	0.99	0.99	0.99	0.99	0.935904
vit_resnet_combined	{'C': 10, 'gamma': 'scale', 'kernel': 'rbf'}	Train	0.986036	0.986071	0.986036	0.986017	3.151890
		Test	0.99	0.99	0.99	0.99	2.714556
vit	{'C': 10, 'gamma': 'scale', 'kernel': 'rbf'}	Train	0.985260	0.985331	0.985260	0.985235	3.444416
		Test	0.99	0.99	0.98	0.98	1.341401
resnet	{'C': 10, 'gamma': 'scale', 'kernel': 'rbf'}	Train	0.979829	0.979849	0.979829	0.979800	2.057819
		Test	0.98	0.98	0.98	0.98	0.315311
hog_color_hsv	{'C': 10, 'gamma': 'scale', 'kernel': 'rbf'}	Train	0.795966	0.798776	0.795966	0.795401	13.503856
		Test	0.80	0.79	0.79	0.79	3.103231
hog_hsv	{'C': 10, 'gamma': 'scale', 'kernel': 'rbf'}	Train	0.775795	0.777884	0.775795	0.774957	9.569369
		Test	0.79	0.78	0.78	0.78	2.015547
hog	{'C': 10, 'gamma': 'scale', 'kernel': 'rbf'}	Train	0.716059	0.719954	0.716059	0.716503	12.142179
		Test	0.72	0.71	0.71	0.71	3.233675
hsv	{'C': 10, 'gamma': 'scale', 'kernel': 'rbf'}	Train	0.658650	0.661897	0.658650	0.658331	9.257810
		Test	0.66	0.66	0.66	0.66	2.232065
color	{'C': 10, 'gamma': 'scale', 'kernel': 'rbf'}	Train	0.626843	0.629875	0.626843	0.626826	3.979828
		Test	0.63	0.63	0.62	0.62	0.740289

D. CNN - Training Experiment Results

Experiment	Learning rate	Epochs	Batch size	Dropout	Training accuracy	Training time (s)
1	1e-3	50	Mini-batch	0.3	0.2234	67.2
2	1e-3	20	Mini-batch	0.3	0.2195	28.8
3	1e-2	20	Full	0.3	0.1924	5.4
4	5e-4	20	Full	0.3	0.2078	5.4
5	1e-3	40	Full	0.3	0.2141	10.2
6	1e-3	20	Full	0.3	0.2266	5.4