Terrain Classification Using Satellite Imagery

Spencer Morris , Maiya Caldwell , Muthumayan Madhayyan
Advisor: Rachel Brown Teaching Assistant: Albert Yu
MIDS W281 | Spring 2024 | Final Project

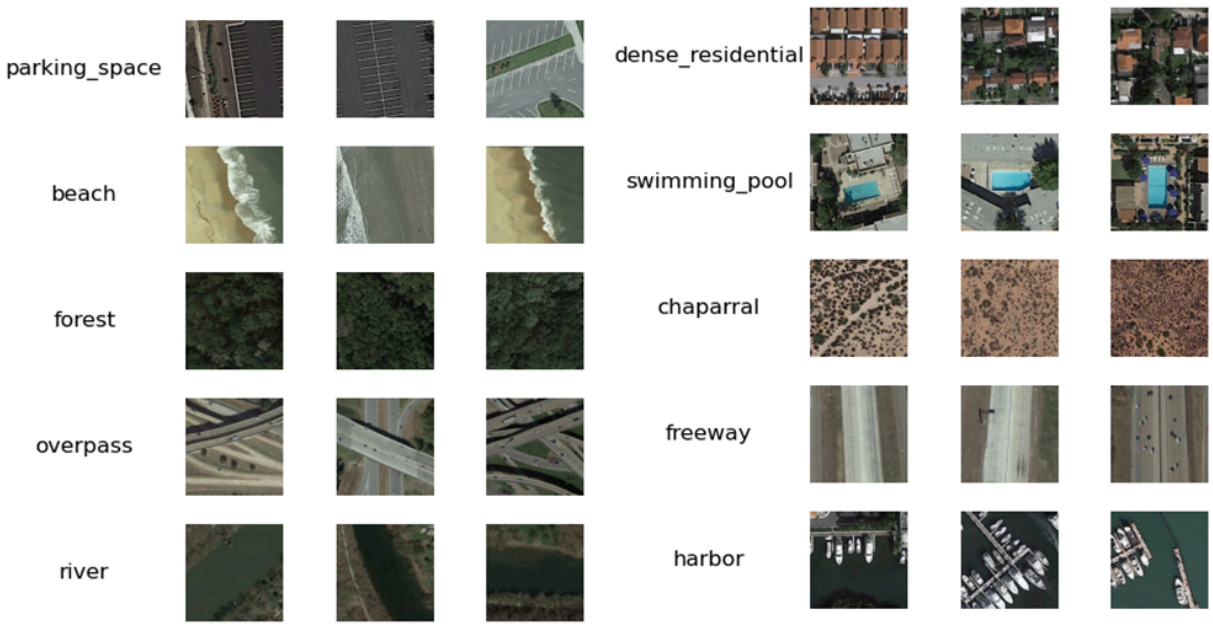
Confusion matrices for HSV feature-set (validation)
Confusion matrices for GLCM+HSV (validation)
Confusion matrices for full feature-set (validation)
Confusion matrices for HSV & GLCM feature-set (test)

Abstract

Remote sensing satellite images are useful in several domains, including but not limited to agriculture, planning, climate studies, navigation, mapping, defense, and more. Our project aims to analyze several satellite images and subject them to well-established image processing techniques in order to classify them into 10 different classes. Our approach is to evaluate each processing technique independently and in concert to figure the best possible method. Herein we present the efficiency and accuracy of these classification models in relation to the feature-vector combination used.

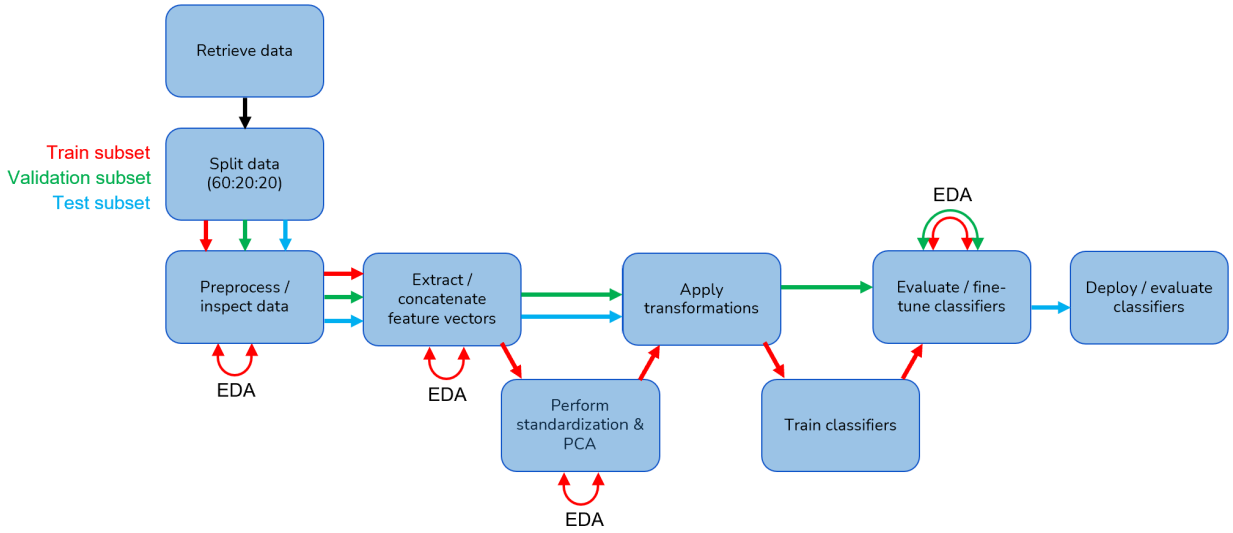
Introduction

PatternNet is a large-scale high-resolution remote sensing dataset collected for remote sensing image retrieval. It includes 38 classes, with each 800 images of size 256×256 pixels associated with each class. The images in *PatternNet* are collected from Google Earth imagery. All the images are manually labeled with a class name. The figure below shows several example images for each class:



Approach

We follow an approach that employs classic image classification methods i.e., without the use of convolutional or deep learning neural networks. Below is a schematic of the full data pipeline used, along with a brief description.



- **Retrieval:** Since each class has about 800 images and there are 38 classes, the number of images included in PatternNet is prohibitively costly (in time) to train, validate and test. We decided to limit the number of classes to 10, choosing a diverse classification criteria that includes a mix of man-made structures vs. natural landscapes and water vs. land-based scenes.
- **Data splitting:** Train/validation/test subsets were immediately defined at a 60:20:20 ratio following data retrieval, thus reserving 160 images per class for training. The training and validation subsets were used for training and validation of the classifier models.
- **Preparation:** A single image was identified as corrupt and removed from consideration. Images were inspected and confirmed to have consistent shape, datatype, and labels. As such, no additional major preparation work was necessary.
- **Feature extraction:** Several classical feature extraction techniques were explored to identify candidate feature vectors. Exploratory Data Analysis (EDA) was performed on the train subset for these features to identify candidate features for inclusion in our model. Color channel information, texture, orientation, and frequency spectrum analysis were all evaluated. We have also incorporated an advanced feature extraction technique such as using SIFT descriptors along with a

Bag of Visual Words approach. Each technique yielded a feature vector of varying dimensions.

- **Dimensionality reduction:** Once a vector was defined incorporating selected features, we performed z-standardization and principal component analysis on the train subset to compress the feature vector for improved computational performance.
- **Classifier training:** A subset of feature vectors were used on the train subset to train Random Forest (RF) and Support Vector Machine (SVM) classifiers. The validation subset was used to judge the predictive performance of the models and evaluate overfitting. A grid-search approach was used to fine-tune the classifiers, and EDA was performed to manually evaluate misclassifications via confusion matrices and visual inspection of images.
- **Deployment:** The tuned models judged to be most performant were deployed on the test subset to measure true model performance.

Subsequent sections in this document will elaborate on further details, and the full set of implementations are available in a github repo, with key notebooks identified in the associated readme and citations section.

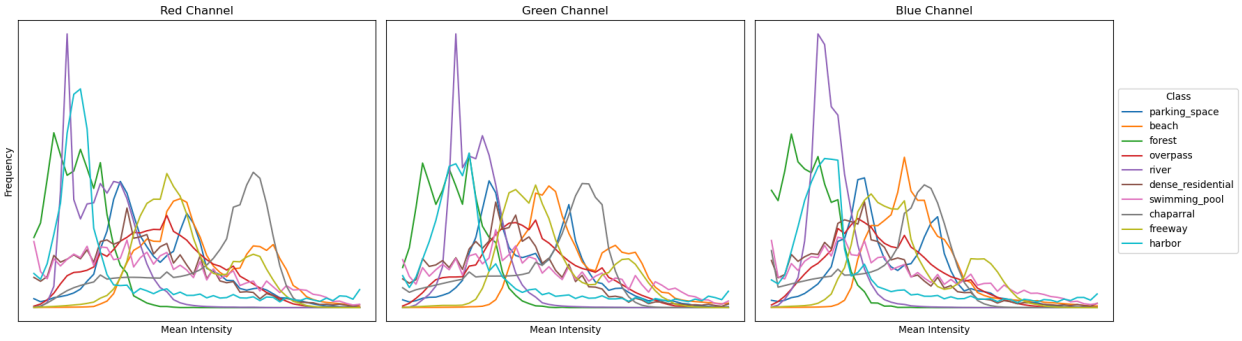
Feature Extraction & Selection

Feature extraction for classification of terrain images involves identifying and extracting relevant visual patterns or attributes from images, such as edges, textures, colors, and shapes. These extracted features serve as descriptive, numerical representations of the images, enabling machine learning models to recognize and classify objects or patterns within images accurately. We chose RGB and HSV color spaces alongside GLCM, HOG, spatial frequency spectrum, and SIFT with Bag of Visual Words (BoVW) for feature extraction in order to capture diverse characteristics such as color, texture, spatial relationships, and local keypoint descriptors, enhancing the robustness and generalizability of our classifiers.

RGB

We chose RGB, representing colors as combinations of red, green, and blue light intensities, for our terrain surveillance imagery due to our understanding of its ability to capture fundamental color information, important for distinguishing our 10 classes, which all have distinct color characteristics. RGB's simplicity also makes it an accessible starting

point for exploring feature extraction across these classes. By analyzing the individual contributions of red, green, and blue channels, RGB facilitates identification of color variations in an image and is helpful in terrain classification tasks where distinct color signatures play an important role in distinguishing terrain.



With RGB as the only feature vector, we perform PCA to yield the 2 principal components contributing to over 97% of the total variance. These principal components were then used for training the Random Forest and SVM classifiers. Please refer to RGB feature-set PCA and tSNE plot.

Using GridSearch, we identified the best hyperparameters to use for training.

```
Best parameters for Random Forest Classifier: {'max_depth': 10, 'n_estimators': 200}
Training time for Random Forest Classifier: 0:00:43.066633
Best parameters for Support Vector Machine: {'C': 10, 'kernel': 'rbf'}
Training time for Support Vector Machine: 0:00:10.782773
```

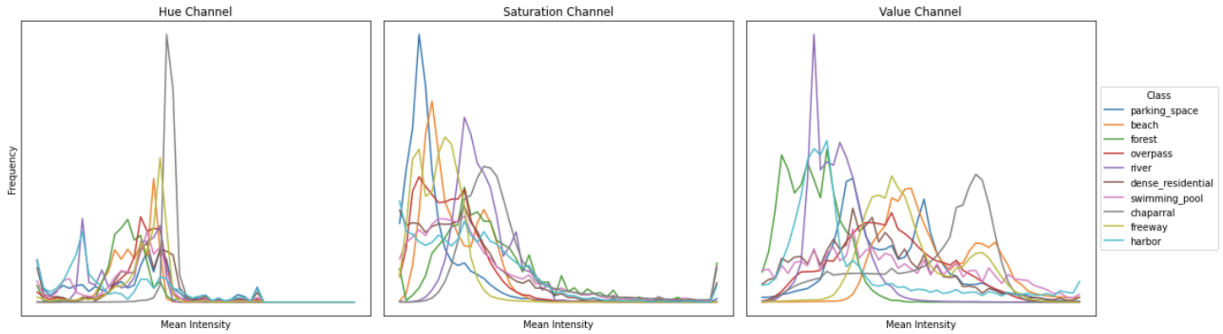
The classifiers were trained and evaluated using just the RGB feature vector, yielding the following accuracy metrics:

```
Random Forest Classifier:
Train Accuracy: 0.8160033340279225
Validation Accuracy: 0.6823014383989994

Support Vector Machine:
Train Accuracy: 0.7034798916440925
Validation Accuracy: 0.6772983114446529
```

Please refer to Confusion matrix for RGB feature-set (validation) for a view of the mismatched classifications. When relying solely on RGB for classification, we had greater than 67% accuracy on the validation set.

Hue, Saturation and Value (HSV)

HSV (hue, saturation, and value) is a color representation model that separates color information from brightness in an image, facilitating better differentiation between various hues and intensities. By isolating hue and saturation, HSV enables better distinction between different colors. This distinction is particularly helpful in terrain classification where unique color patterns, i.e. green for vegetation or blue for water, serve as key indicators for identifying different types of terrain, enhancing the performance of our classifiers.



With HSV as the only feature vectors, we perform PCA to yield the 15 principal components contributing to over 80% of the total variance. These principal components were then used for training the Random Forest and SVM classifiers. Please refer to [HSV feature-set PCA and tSNE plots].

Using GridSearch, we identified the best hyperparameters to use for training.

```
Best parameters for Random Forest Classifier: {'max_depth': None, 'n_estimators': 100}
Training time for Random Forest Classifier: 0:01:21.791120
Best parameters for Support Vector Machine: {'C': 10, 'kernel': 'rbf'}
Training time for Support Vector Machine: 0:00:07.592407
```

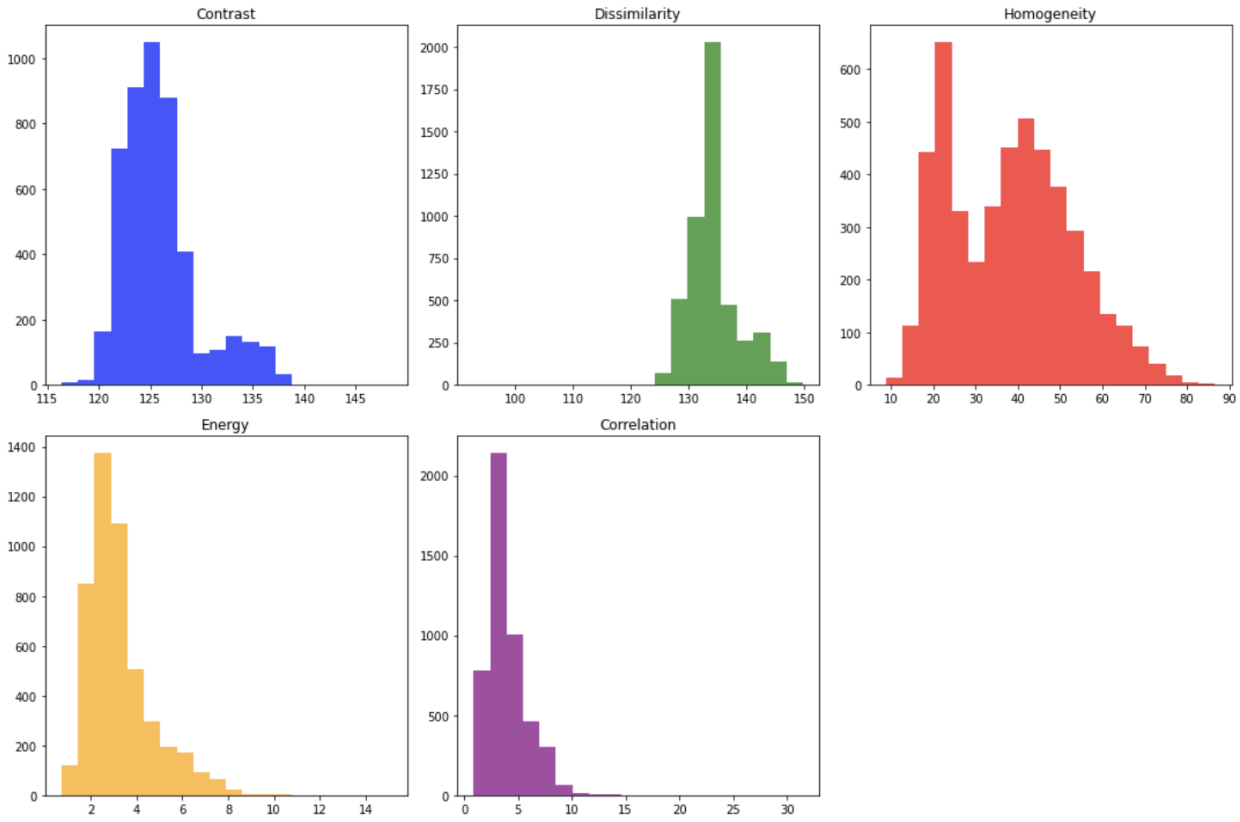
The classifiers were trained and evaluated using just the HSV feature vector, yielding the following accuracy metrics:

```
Random Forest Classifier:
Train Accuracy: 1.0
Validation Accuracy: 0.9681050656660413

Support Vector Machine:
Train Accuracy: 0.9952073348614294
Validation Accuracy: 0.9843652282676673
```

Please refer to Confusion matrices for HSV feature-set (validation) for a view of the mismatched classifications. When relying solely on HSV for classification, we had greater than 96% accuracy on the validation set.

Gray-level Co-occurrence Matrix (GLCM)

GLCM is a method used to describe the texture of an image by analyzing the spatial relationships between pixel intensity values. By estimating how often pairs of pixel values occur together at a particular distance & angle, GLCM helps capture textural patterns in an image. Examples of the histograms for contrast, dissimilarity, homogeneity, energy, and correlation below help depict the distribution of these texture features across an image, demonstrating variations, uniformity, patterns, and relationships between pixel intensities. These are useful in terrain classification where capturing unique textures such as smooth roads or ragged forests can help discern different terrains.



With GLCM as the only feature vector, we perform PCA to yield the 3 principal components contributing to over 97% of the total variance. These principal components were then used

for training the Random Forest and SVM classifiers. Please refer to the GLCM feature-set PCA and tSNE plot.

Using GridSearch, we identified the best hyperparameters to use for training.

```
Best parameters for Random Forest Classifier: {'max_depth': None, 'n_estimators': 100}
Training time for Random Forest Classifier: 0:00:40.109018
Best parameters for Support Vector Machine: {'C': 10, 'kernel': 'rbf'}
Training time for Support Vector Machine: 0:00:06.930776
```

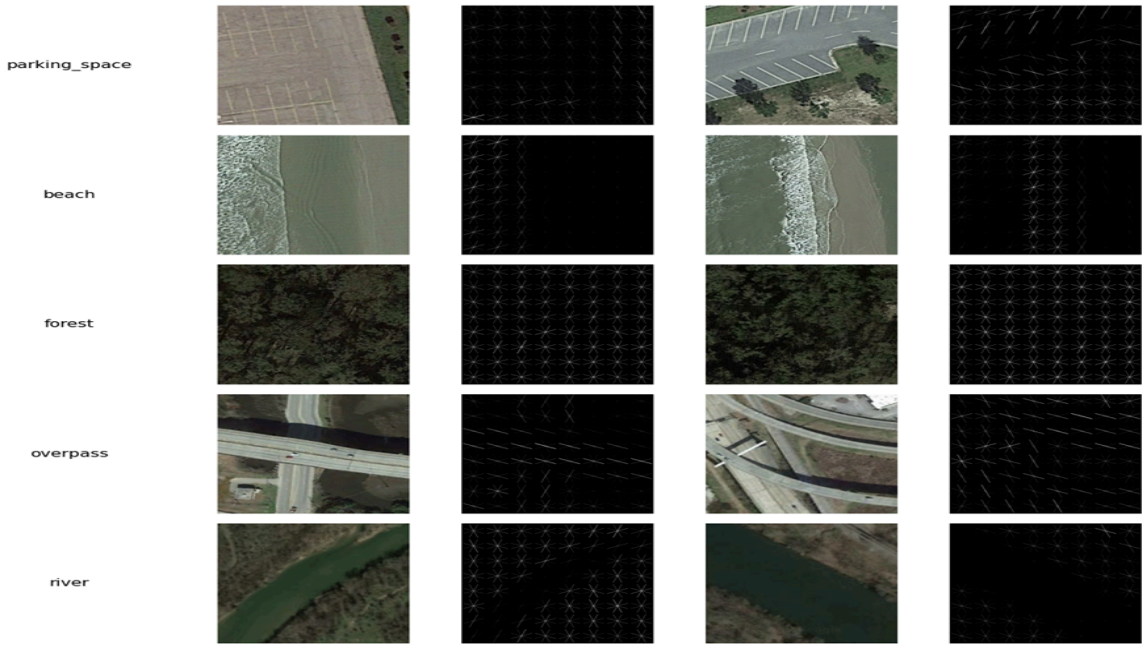
The classifiers were trained and evaluated using just the HSV feature vector, yielding the following accuracy metrics:

```
Random Forest Classifier:
Train Accuracy: 1.0
Validation Accuracy: 0.8855534709193246

Support Vector Machine:
Train Accuracy: 0.8887268180871015
Validation Accuracy: 0.8824265165728581
```

Please refer to Confusion matrix for GLCM feature-set (validation) for a view of the mismatched classifications. When relying solely on GLCM for classification, we had greater than 88% accuracy on the validation set.

Histogram of Oriented Gradients (HOG)

HOG is a technique used to describe the local gradient of info in an image. HOG captures shape and edge information by computing the gradient orientations in localized regions which helps in detecting object boundaries and texture patterns. These features are valuable for identifying distinctive shapes and structures in terrain, such as the distinctive shapes of shorelines on beaches or the linear patterns of roads and traffic lanes.



With HOG as the only feature vector, we perform PCA to yield 165 principal components contributing to over 94% of the total variance. These principal components were then used for training the Random Forest and SVM classifiers. Please refer to the HOG feature-set PCA and tSNE plot.

Using GridSearch, we identified the best hyperparameters to use for training.

```
Best parameters for Random Forest Classifier: {'max_depth': 20, 'n_estimators': 300}
Training time for Random Forest Classifier: 0:05:57.542895
Best parameters for Support Vector Machine: {'C': 10, 'kernel': 'rbf'}
Training time for Support Vector Machine: 0:10:38.536004
```

The classifiers were trained and evaluated using just the HOG feature vector, yielding the following accuracy metrics:

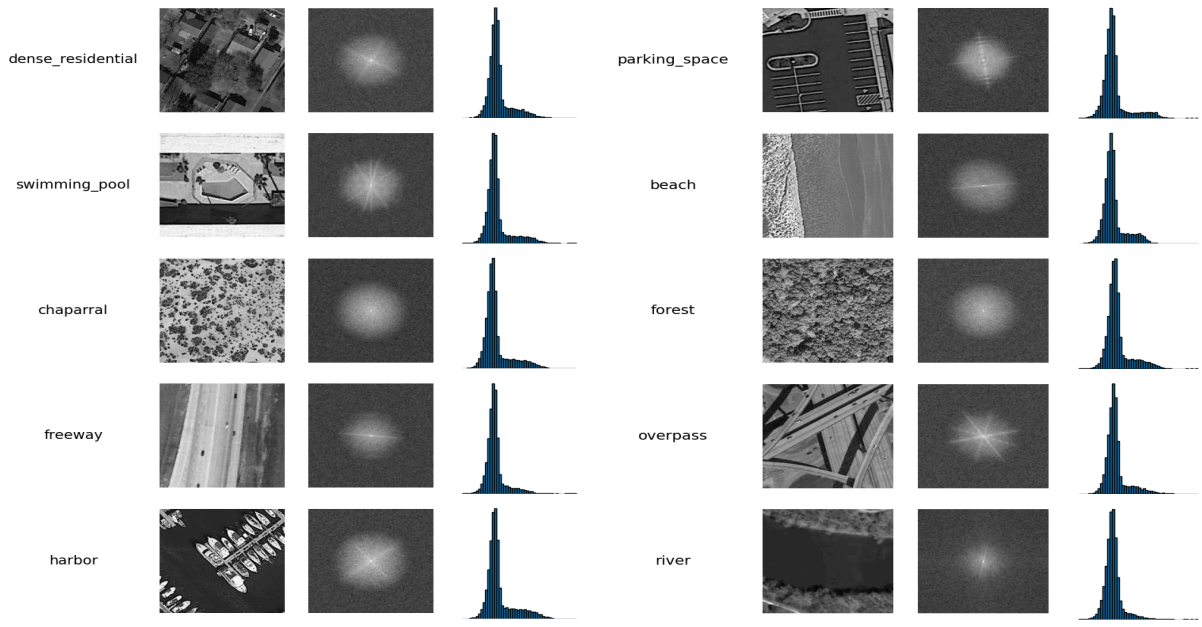
```
Random Forest Classifier:
Train Accuracy: 1.0
Validation Accuracy: 0.8624140087554721

Support Vector Machine:
Train Accuracy: 0.9881225255261513
Validation Accuracy: 0.9199499687304565
```

Please refer to Confusion matrix for HOG Frequency feature-set (validation) for a view of the mismatched classifications. When relying solely on HOG for classification, we had greater than 86% accuracy on the validation set.

Spatial Frequency Spectrum

Since these satellite images have a combination of natural landscapes and human-made structures, we anticipated that the spatial frequency spectrum of these images would contain distinct patterns suitable as feature vectors. We converted each image to grayscale, performed a fast fourier transform, then applied a low-pass gaussian filter to remove high-frequency content. The premise of applying a low pass filter was to retain the gradual variations of pixel intensities while filtering out the rapid changes. Although the low pass filter did NOT filter out much of the high frequency content, it seemed to have a positive effect on the classification accuracy. So, we retained this additional step of filtering.

The frequency spectrum output of the transform was then broken into 20 bands (in logarithmic scale). With about 20 frequency bands we achieved a separable distribution (as seen on in the following histograms) between the various classes of images. These distributions were then used as candidate feature vectors.



Using the frequency distribution as feature vectors, we subject them through PCA to yield the 7 principal components contributing to over 90% of the total variance. These principal

components were then used for training the Random Forest and SVM classifiers. Please refer to Spatial Frequency feature-set PCAand tSNE plots.

Using GridSearch, we identified the best hyperparameters to use for training.

```
Best parameters for Random Forest Classifier: {'max_depth': None, 'n_estimators': 100}
Training time for Random Forest Classifier: 0:01:07.164655
Best parameters for Support Vector Machine: {'C': 10, 'kernel': 'rbf'}
Training time for Support Vector Machine: 0:00:09.667977
```

The classifiers were trained and evaluated using just the spatial frequency vector, yielding the following accuracy metrics:

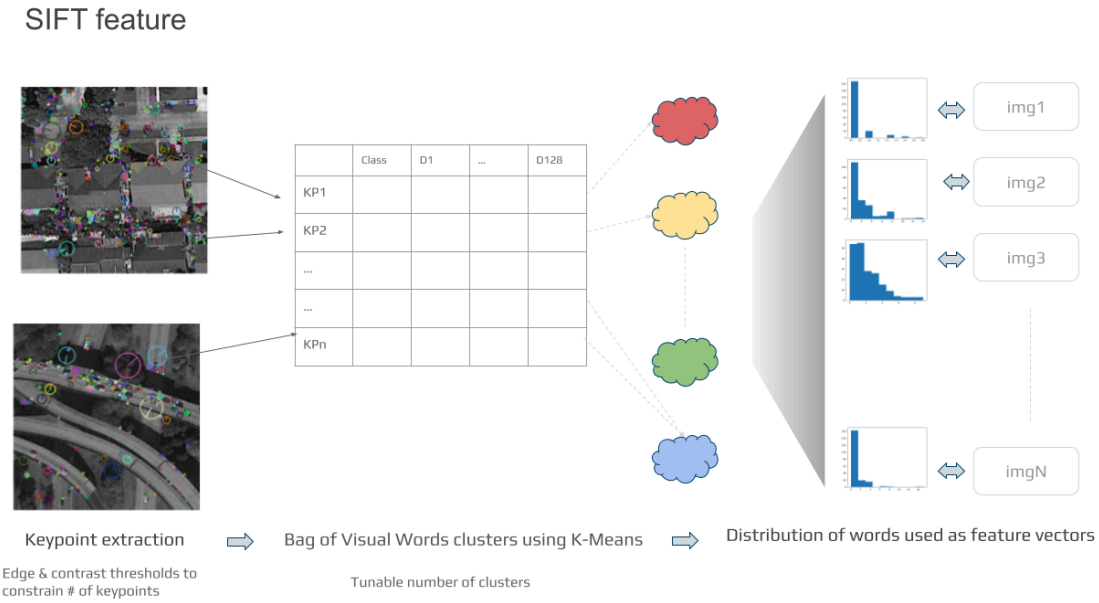
```
Random Forest Classifier:
Train Accuracy: 1.0
Validation Accuracy: 0.8549093183239524

Support Vector Machine:
Train Accuracy: 0.9124817670347989
Validation Accuracy: 0.8749218261413383
```

Please refer to Confusion matrix for Spatial Frequency feature-set for a view of the mismatched classifications. When relying solely on the spatial frequencies for classification, only natural landscapes like rivers and forests are accurately identified. The rest of them have a good percentage of mismatches.

SIFT + BoVW

SIFT (Scale-Invariant Feature Transform) combined with Bag of Visual Words (BoVW) is a method used for detecting and describing local features in an image and representing them in a more compact way. SIFT detects distinctive keypoints and extracts feature components that are invariant to scale & rotation. BoVW aggregates these components into a vocabulary of visual words. For terrain classification, SIFT features can capture distinct visual patterns like the arrangement of trees, buildings, bodies of water, and BoVW enables efficient representation of these patterns, helping in identifying different types of terrain.

The distribution of the visual words for a given image was captured as a feature vector. A brief overview of the process is captured below:



We used a k-cluster count to 20 for feature extraction. Then, we took this through the same process of PCA before training the classifiers. The PCA reduced the number of principal components to 5, which captured over 92% of total feature variation. Those principal components were then used for training the Random Forest and SVM classifiers. Please refer to SIFT feature-set PCAand tSNE plots.

Using GridSearch, we identified the best hyperparameters to use for training.

```
Best parameters for Random Forest Classifier: {'max_depth': 20, 'n_estimators': 300}
Training time for Random Forest Classifier: 0:01:06.800111
Best parameters for Support Vector Machine: {'C': 10, 'kernel': 'rbf'}
Training time for Support Vector Machine: 0:00:08.553979
```

The following accuracy metrics summarize the predictive performance of our classifiers when trained and validated using only SIFT/BoVW feature vectors.

```
Random Forest Classifier:
Train Accuracy: 0.9981246092936028
Validation Accuracy: 0.32020012507817386

Support Vector Machine:
Train Accuracy: 0.8437174411335695
Validation Accuracy: 0.3595997498436523
```

Please refer to the Confusion matrix for SIFT feature-set for a view of the mismatched classifications. Although the model fared reasonably well with certain classes (namely

freeway, overpass, river, and swimming pool), this feature was clearly a poor one (at least in isolation) for our classification task.

Best combination (GLCM and HSV)

We experimented with a combination of feature-sets. HOG had a good standalone accuracy (>97%) but the number of principal components was pretty high (182). We noticed that both HSV and GLCM had reasonable accuracy while the number of principal components was about 17.

Using the combination of HSV and GLCM feature vectors, we subject them through PCA to yield the 17 principal components contributing to over 82% of the total variance. These principal components were then used for training the Random Forest and SVM classifiers. Please refer to GLCM+HSV feature-set PCA and tSNE plot.

Using GridSearch, we identified the best hyperparameters to use for training.

```
Best parameters for Random Forest Classifier: {'max_depth': None, 'n_estimators': 100}
Training time for Random Forest Classifier: 0:01:39.016715
Best parameters for Support Vector Machine: {'C': 10, 'kernel': 'rbf'}
Training time for Support Vector Machine: 0:00:06.575821
```

The classifiers were trained and evaluated using GLCM and HSV feature vectors, yielding the following accuracy metrics:

```
Random Forest Classifier:
Train Accuracy: 1.0
Validation Accuracy: 0.9781113195747342

Support Vector Machine:
Train Accuracy: 0.9972911023129819
Validation Accuracy: 0.9912445278298937
```

Please refer to Confusion matrices for GLCM+HSV (validation) for a view of the mismatched classifications.

Both the classifiers had a remarkable accuracy - 97.8 and 99.1 percent for the validation set.

With these two features, the mismatches are mostly around the parking space being misclassified as an overpass. Visual inspection shows that the overpass seems to have some similarity to parking space in terms of texture.

Principal Component Analysis

Given that images are inherently high-dimensional data, extracting selected features into vectors is a crucial step for making classification problems tractable. Even so, feature vectors themselves can often be prohibitively large, especially for features such as HOG or when joined together, and as datasets grow in number.

Principal Component Analysis (PCA) is a method for reducing the dimensionality of complex datasets (including those represented as feature vectors). PCA is the process by which the covariance matrix of a given dataset is diagonalized by a new orthonormal basis. This new orthonormal basis comprises the so-called principal components of the original dataset, and can be represented as a linear combination of the original features.

The first principal component captures the maximum variance in the data, followed by subsequent (orthogonal) principal components that capture diminishing variance. Once the PCA is completed, a cutoff criterion can be established to choose the optimum number of components that capture some threshold of the cumulative variance in the original dataset. The cutoff criterion used herein is to plot the cumulative variance against the principal components and identify the inflection point (so-called elbow-plot method). Using this approach, most principal components chosen captured at least 90-95% of the total dataset variation.

t-SNE

t-Distributed Stochastic Neighbor Embedding (t-SNE), like PCA, is a method for compressing a dataset, but it is different in key ways. Like PCA, it projects a dataset to a lower-dimensional space (almost always 2-dimensions or 3-dimensions), but it is a nonlinear technique. Its aim is to preserve local similarities between data points (in contrast to PCA, which captures global variance). As such, it's an excellent tool for visualizing a high-dimensional dataset in a plot to observe clustering of data points by class.

Importantly, t-SNE is not a suitable transformation for subsequent modeling, since the number of factors is fixed at 2 or 3 and point-wise distances are computed for a specific set

of data in order to minimize local and maximize global structure. Furthermore, a lack of obvious structure in a tSNE plot is not necessarily indicative of a poor dataset representation. All t-SNE plots generated therein were computed on selected principal components, were visualized in two dimensions, and were used for illustration only.

Classification

Classifiers

Random Forest

We chose the Random Forest classifier due to its suitability for handling multiclass problems using an ensemble approach, where diverse terrain features need to be accurately categorized. Additionally, its ability to mitigate overfitting by combining predictions from multiple trees ensures reliable performance, ideal for our project using diverse feature sets.

Support Vector Machine

Support Vector Machine (SVM) was selected for its strength in capturing complex and intricate relationships in high-dimensional feature spaces, making it suitable for our task of discerning distinct terrain classes with potentially nonlinear boundaries, as terrains often appear to have complex spatial relationships that require a classifier capable of capturing such subtle variations.

Hyperparameter Tuning

To improve the performance of each of our classifiers, we utilized GridSearch, a hyperparameter optimization technique that searches through a provided grid of hyperparameter values, to find optimal hyperparameter combinations to maximize their accuracy. We found these, shown below and bolded, amongst our search spaces, for our models using the feature vectors of all explored feature types, as well as for our feature vectors with HSV and GLCM features only, since they help our classifiers perform best.

In Random Forest, the “estimators” hyperparameter controls the number of decision trees in the ensemble, influencing the complexity of the model and against overfitting, while “max depth” limits the depth of each tree in the ensemble, mitigating the risk of capturing noise in the data.

In SVM, “C” helps to regulate the tradeoff between maximizing the margin and minimizing classification errors: larger values prioritize minimization of these errors, smaller prioritizes margin maximization. “Kernel” determines the type of decision boundary, which allows the model to capture relationships in the data through various function kernels, such as linear, polynomial, or radial basis function kernels.

Feature	Random Forest Classifier		SVM Classifier	
	max_depth	#estimators	kernel	C
RGB	10	200	RBF	10
HSV	None	100	RBF	10
HOG	20	300	RBF	10
GLCM	None	100	RBF	10
Spatial Freq	None	100	RBF	10
SIFT	20	300	RBF	10
All the above	20	300	RBF	10
HSV + GLCM	None	100	RBF	10

Analysis

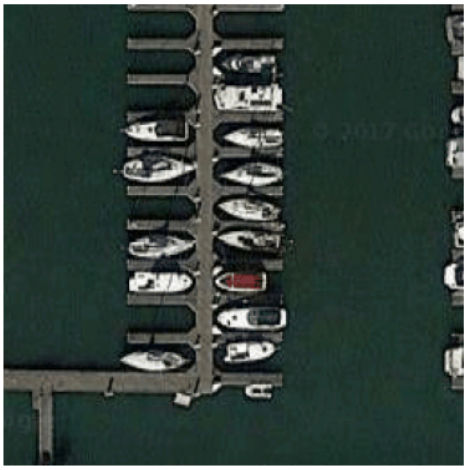
In our analysis of confusion matrices using the validation set of all features, both Random Forest and SVM classifiers exhibited some misclassifications, particularly between chaparrals and swimming pools, as well as between beach and freeway classes. These errors may come from similarities in certain visual characteristics, as we mention below in our misclassification analysis.

However, when using just HSV features or when combining HSV & GLCM features, classification performance improved for both models. This upgrade in performance suggests that the subset of features selected for our classifiers effectively captured distinctive characteristics of the terrain classes, resulting in reduced ambiguity during classification.

These observations highlight the importance of feature selection in improving classifier performance as well as the limitations of relying on specific features, like RGB or HOG, which may not effectively capture the distinctions between terrain classes.

Misclassifications

Understanding misclassifications is important in assessing the limitations of our classifiers, providing insights into factors influencing erroneous predictions and guiding improvements in model performance. For example, the misclassification of a harbor as a parking space highlights the tendency of each classifier to interpret spatial arrangement, like clusters of objects and gaps between, resembling cars and potentially leading to confusion.



Similarly, the misclassification of swimming pools as dense residential neighborhoods may indicate the influence of contextual cues, i.e. where the presence of pools within similar neighborhoods may create visual similarities that challenge accurate classification.

These examples show the complexity of terrain classification tasks and highlight the importance of considering surrounding information for improving accuracy. Moving forward, refining feature extraction methods or incorporating other environmental cues would likely enhance each classifier's ability to identify subtle differences between the different terrain classes.

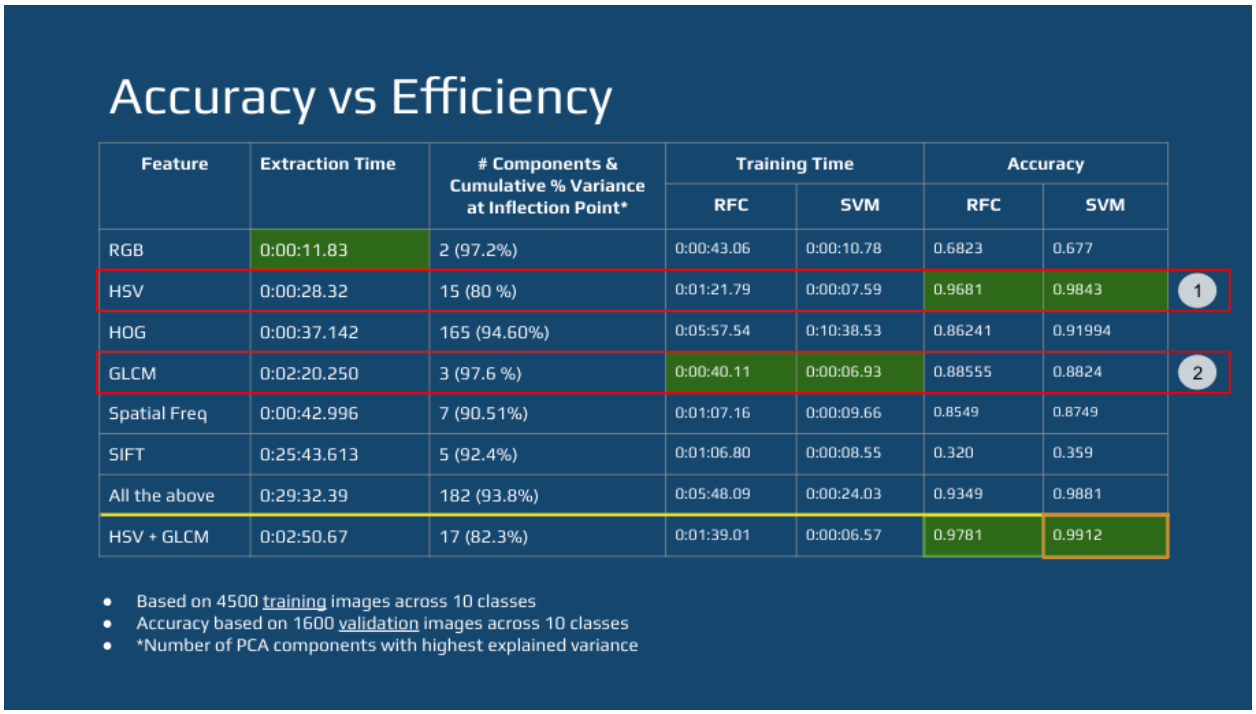
Generalizability

Performance evaluation on the test set revealed that the SVM classifier consistently outperformed the Random Forest classifier, indicating a robust generalizability of our trained models. The uniform accuracy across all classes in our chosen subset when only using HSV and GLCM features suggests that the classifiers are capable of effectively distinguishing between diverse terrain types without particular bias towards specific classes. To further enhance generalizability, future training processes could potentially explore additional feature extraction and hyperparameter optimization techniques, ensuring the classifiers perform well across various datasets. Data augmentation techniques such as rotating, flipping, or adding noise to the images could also further help mitigate any potential overfitting as well as the generalizability of our models on unseen data.

Efficiency vs Accuracy

In comparing accuracy across different feature sets and classifiers, it's important to note that the baseline accuracy would have been 10% due to chance, given the dataset's ten classes. Given all feature types independently as well as altogether, we found that the accuracy of our models were highest just using the HSV features, but faster to train just using the GLCM features. We found that using all feature types with 182 specific components did pretty well, but with just 17 we were able to achieve our highest accuracy using just GLCM and HSV features, taking about one-quarter of the training time. SVM outperformed Random Forest in both training time and accuracy.

In discussing efficiency vs. accuracy, it is worth mentioning the significant difference in extraction time, particularly with SIFT features, which took almost 26 minutes, or all features together, which took about 30 minutes, as opposed to the much quicker extraction times using other feature sets. This emphasizes the role of extraction time in the evaluation of classifier efficiency.



Test Dataset Validation

Our dataset contains 1600 images across the subset of 10 classes; we had isolated about 20% of images at the start of the project for testing. We chose the two best feature types that have the best combination of efficiency and accuracy i.e., GLCM and HSV. Using the test dataset, we evaluated the Random Forest and SVM classifiers with the chosen feature types. The results are summarized below:

```
Random Forest Classification:
Train Accuracy: 1.0
Test Accuracy: 0.9781113

Support Vector Machines:
Train Accuracy: 0.9972
Test Accuracy: 0.9912
```

Please refer to Confusion matrices for HSV & GLCM feature-set (test) for a view of the mismatched classifications.

Conclusion

Image processing is one of the most important use cases for machine learning. However, signal processing techniques developed in the past decades are fundamental to these advances in machine learning. In this project, we experiment with these classical image processing techniques to extract features from images. The efficacy of these techniques vary with the type of images processed and the features that are relevant to the task on hand. In this project, the images are satellite images and our task is to build a classifier that can segregate images based on the classical image processing techniques (as opposed to deep learning techniques).

The project is a culmination of all the various image processing techniques that were covered in the course asynchronous material, live session, and the homework assignments. All those lessons were instrumental in preparing us for this project.

We were pleasantly surprised that simple feature extraction techniques like HSV and GLCM have that much accuracy in discriminating between classes. Perhaps this is an artifact of the mix of classes we chose or just that the dataset contains images of large natural bodies and man-made structures. Nevertheless, it is an affirmation of the power of these techniques. We were able to discern that more complex features like SIFT were not necessarily a good fit for this dataset. While HOG had a good accuracy, it also had higher principal components. The final choice of using HSV and GLCM proved ideal in pruning the number of components with pretty efficient training and accurate classification.

Given the time constraints of this project, we did not have time to implement all of our ideas. Overfitting was observed for most Random Forest models as well as training using the complex feature vectors, indicating that further tuning may have improved the generalization of those models. We were also interested in exploring additional complex feature vectors, in particular exploiting transfer learning from public image classification models such as ResNet. Finally, extending the model to all classes within the dataset could be demonstrative to assessing the generalizability of our best models.

We were concerned about the overfitting experienced during training, but were relieved to see the test accuracy was close to the training accuracy. But generalization is a concern. We believe the methods suggested in the above section of Generalization should be able to address some of the concerns.

We believe that this project gave a better understanding of real-world image processing problems and the confidence to apply them.

References

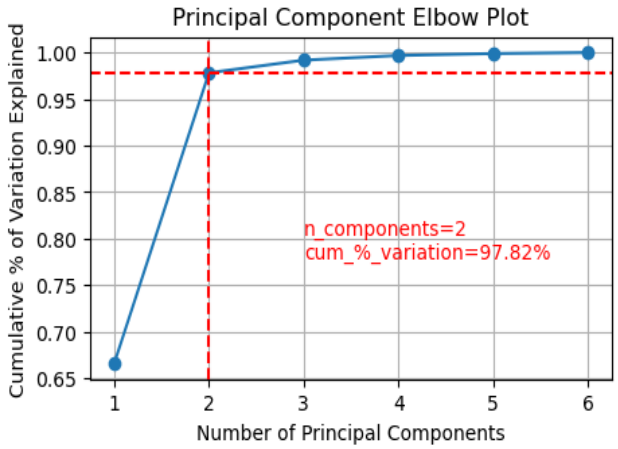
1. Project GitHub Repository
    a. Notebooks for Feature Selection and Extraction
    b. Notebook for Transformations, PCA, and Visualizations
    c. Notebook for Model Grid Search, Evaluation, and Deployment
2. Zhou, W., Newsam, S., Li, C., & Shao, Z. (2018). PatternNet: A benchmark dataset for performance evaluation of remote sensing image retrieval. *ISPRS Journal of Photogrammetry and Remote Sensing, 145,* 197-209.
3. https://medium.com/analytics-vidhya/fast-cnn-substitution-of-convolution-layers-with-fft-layers-a9ed3bfdc99a

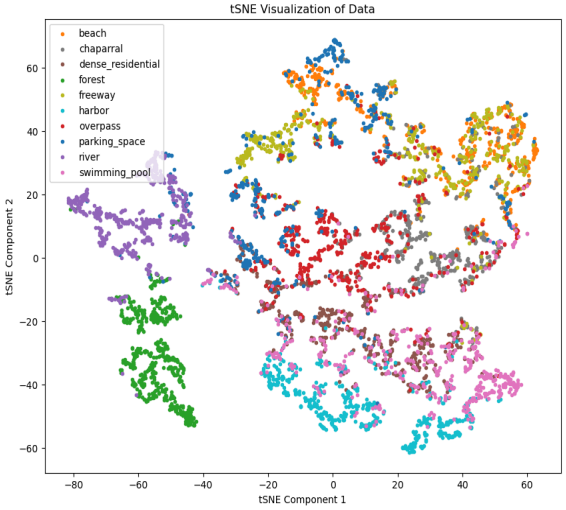
4. https://www.mdpi.com/2072-4292/14/3/574
5. https://www.kdnuggets.com/2023/05/principal-component-analysis-pca-scikitlearn.html

Appendix

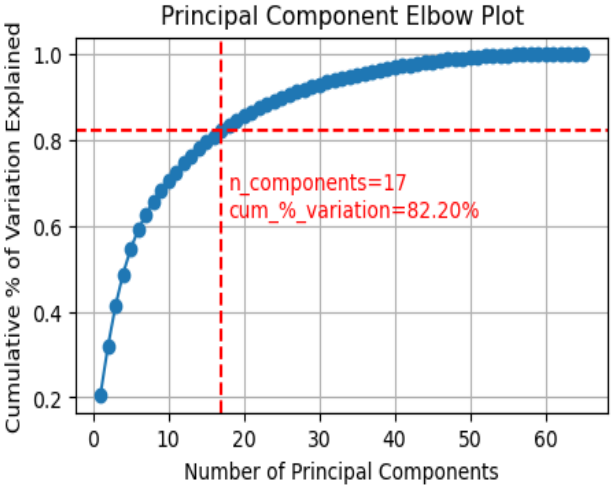
PCA and tSNE plots

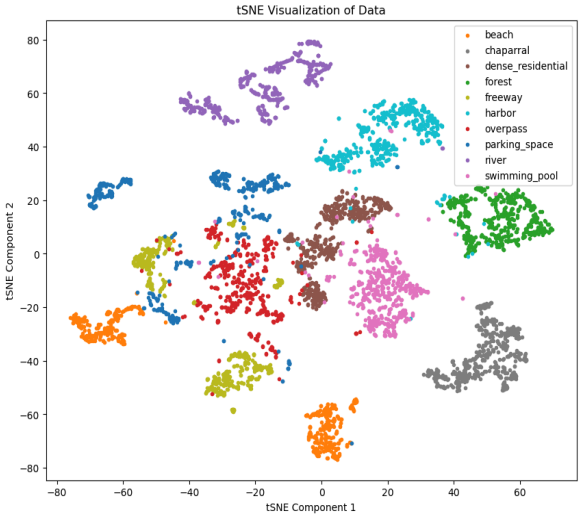
RGB feature-set PCA and tSNE plot



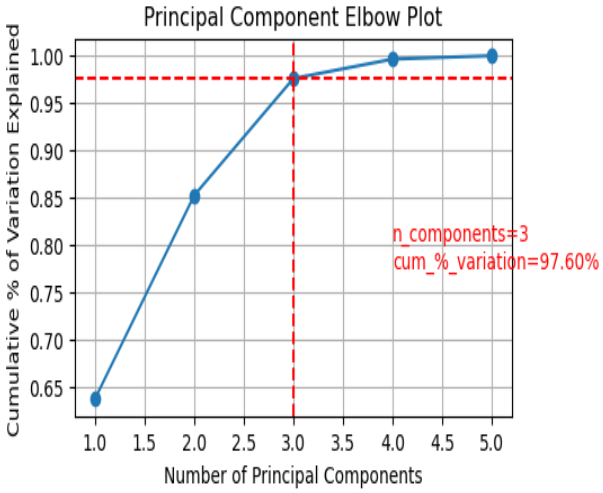


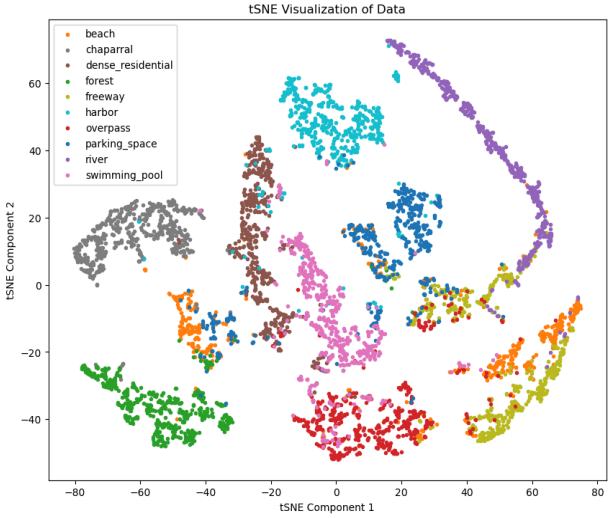
GLCM+HSV feature-set PCA and tSNE plot


Principal Component Elbow Plot
Cumulative % of Variation Explained
Number of Principal Components
n_components=17
cum_%_variation=82.20%
1.0
0.8
0.6
0.4
0.2
0
10
20
30
40
50
60

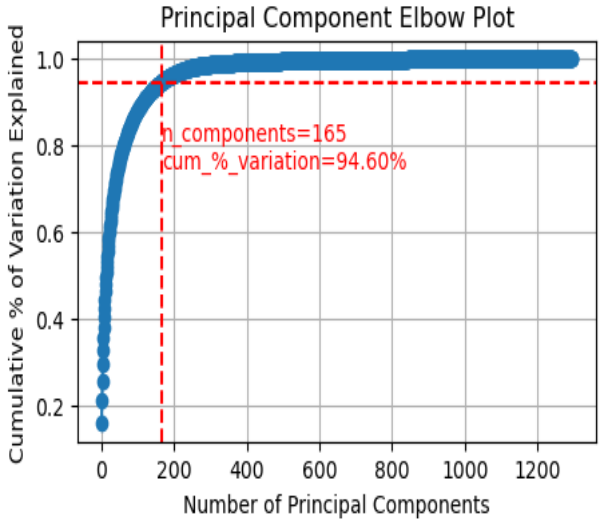

tSNE Visualization of Data
tSNE Component 1
tSNE Component 2
beach
chaparral
dense_residential
forest
freeway
harbor
overpass
parking_space
river
swimming_pool

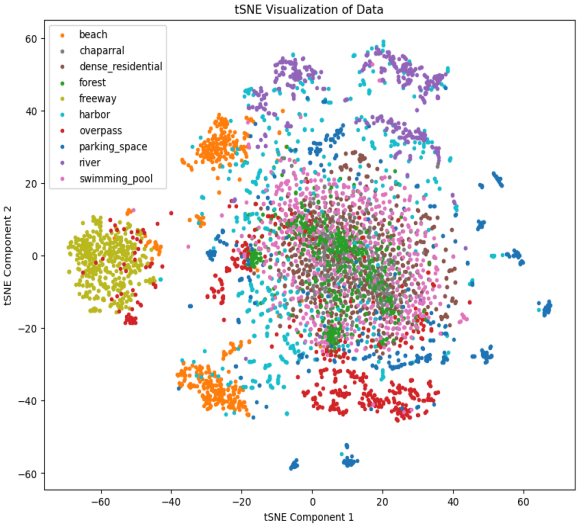
GLCM feature-set PCA and tSNE plot



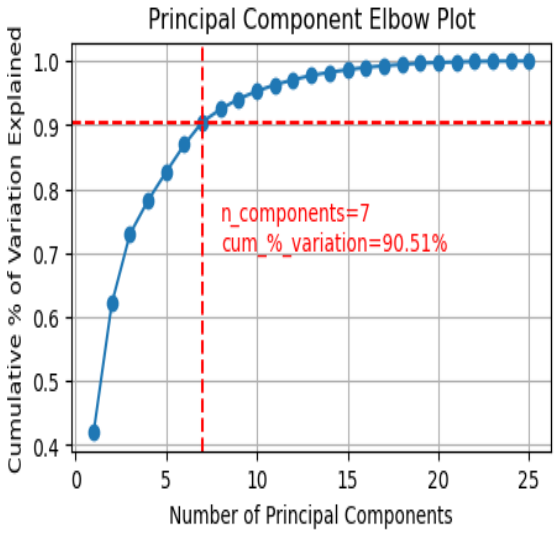


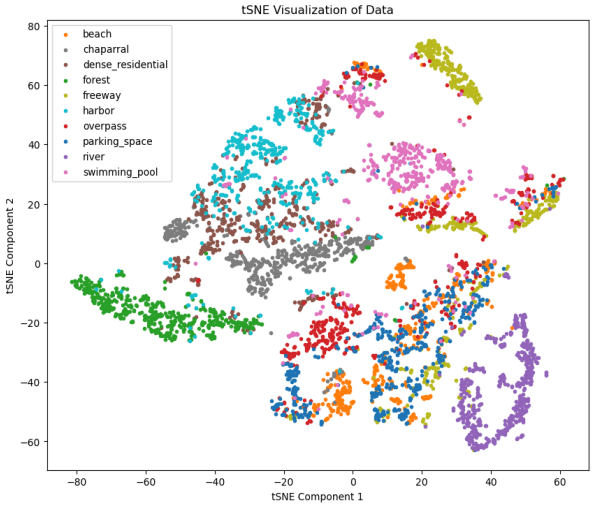
HOG feature-set PCA and tSNE plot



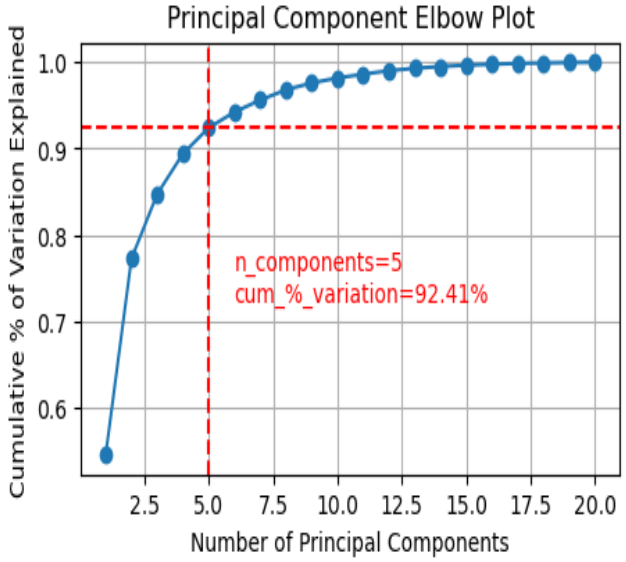


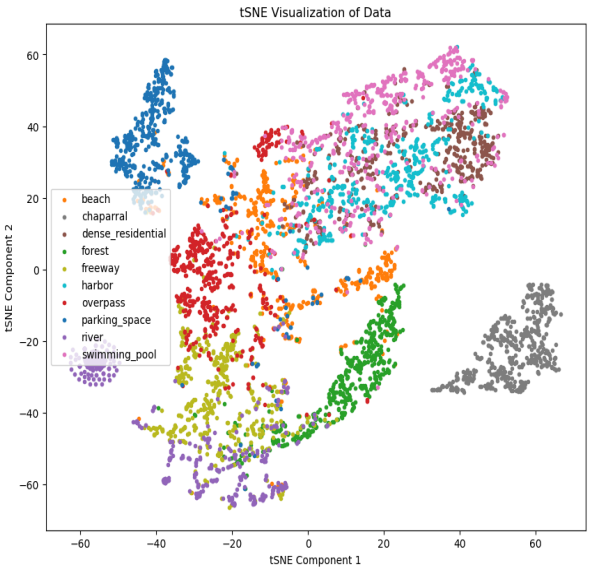
Spatial Frequency feature-set PCA and tSNE plots



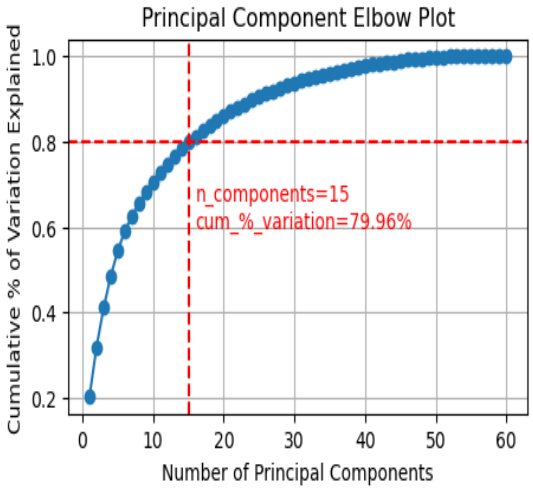


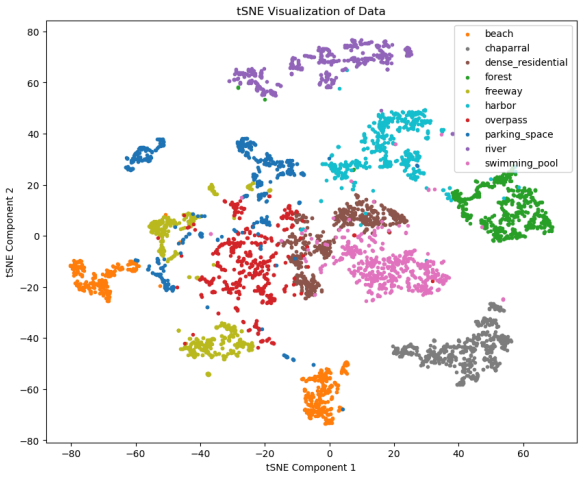
SIFT feature-set PCAand tSNE plots



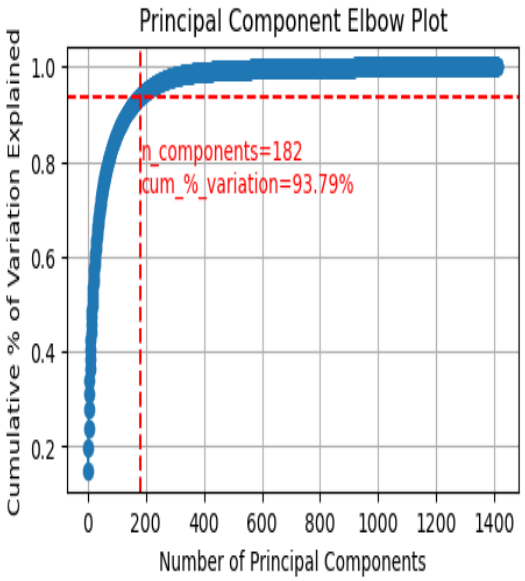


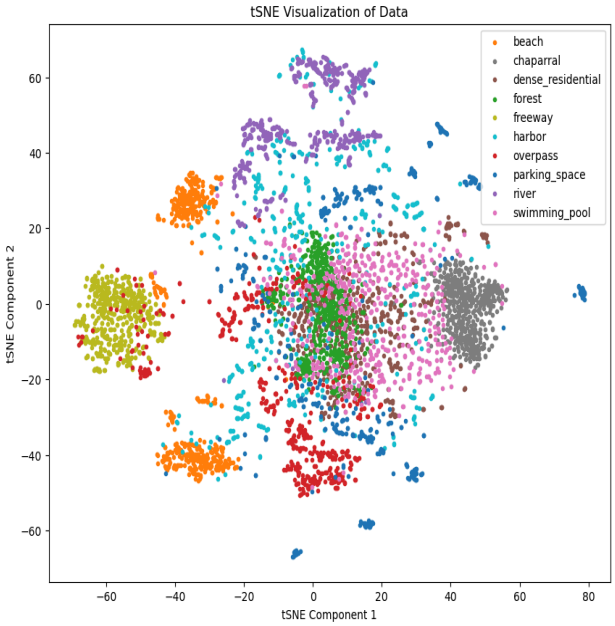
HSV feature-set PCA and tSNE plots





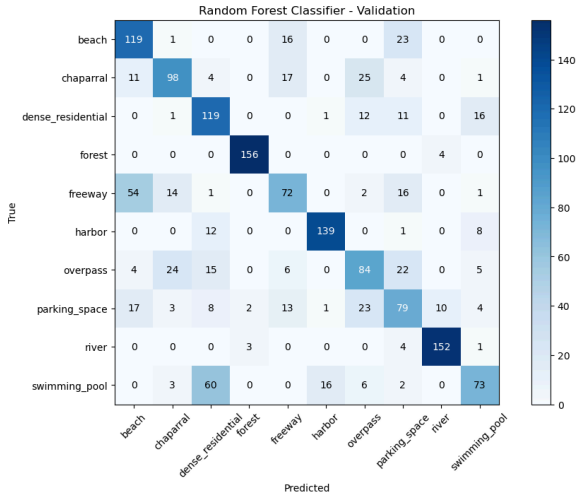
All feature-sets PCA and tSNE plots

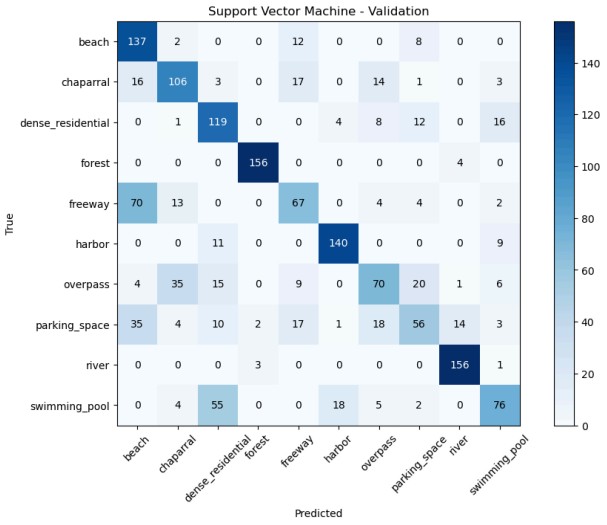




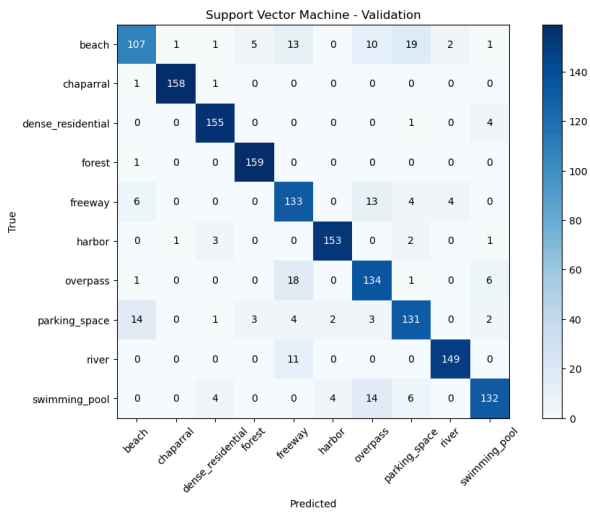
Confusion Matrices

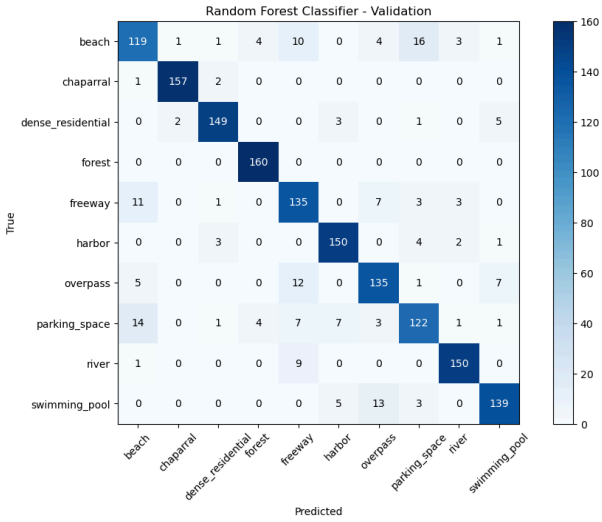
Confusion matrix for RGB feature-set (validation)



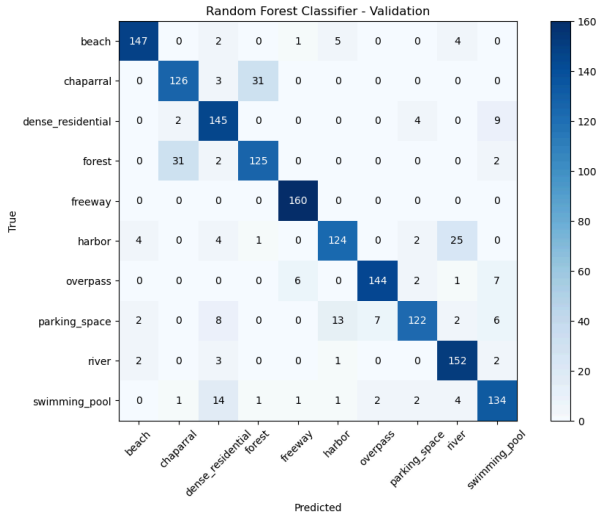


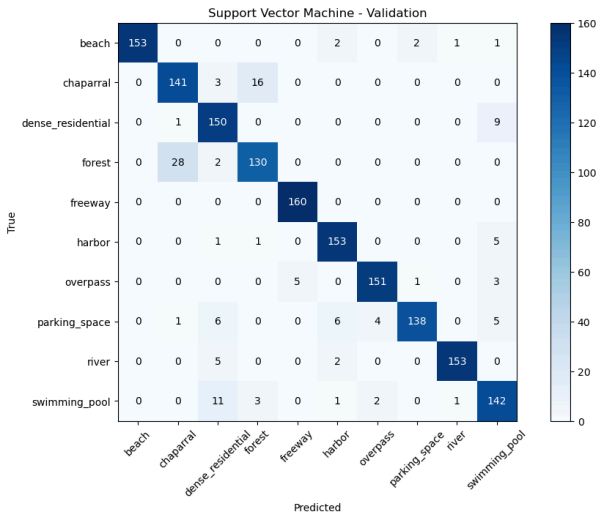
Confusion matrix for GLCM feature-set (validation)



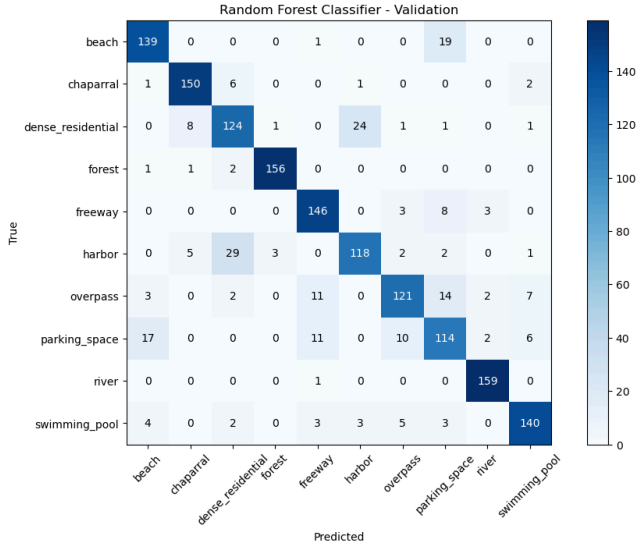


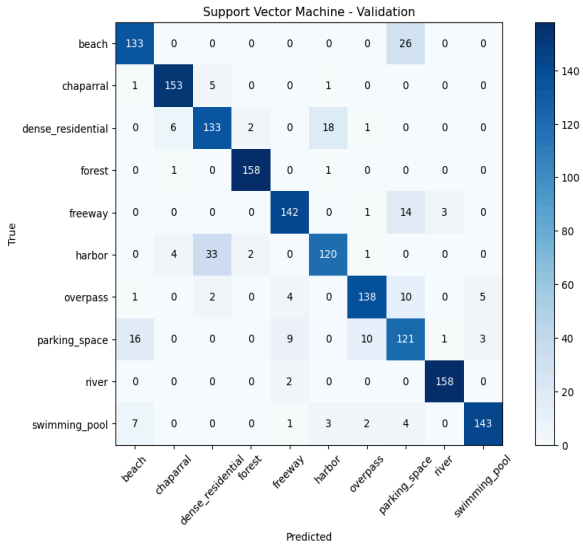
Confusion matrix for HOG Frequency feature-set (validation)



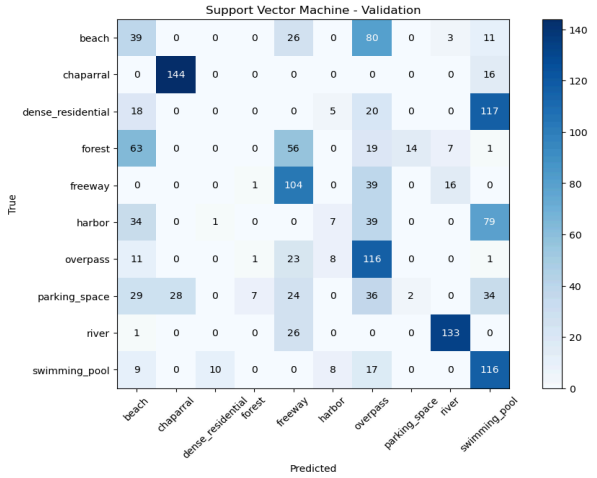


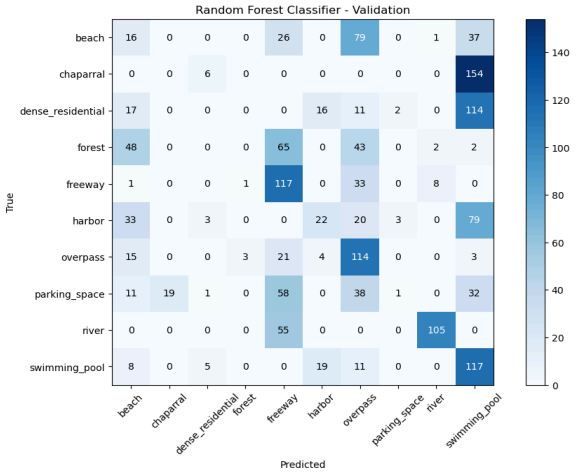
Confusion matrix for spatial frequency feature-set (validation)



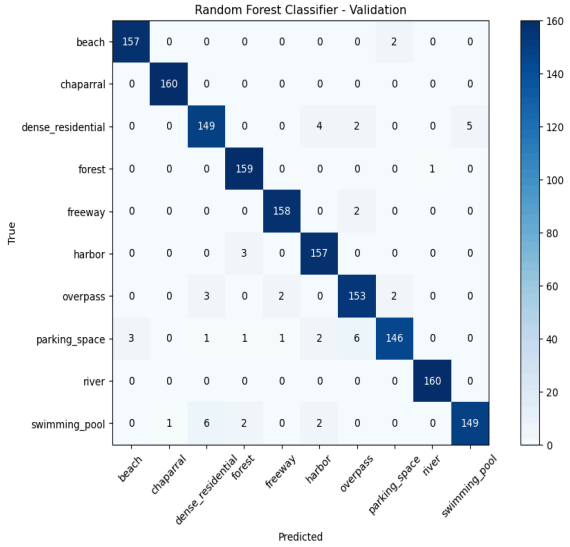


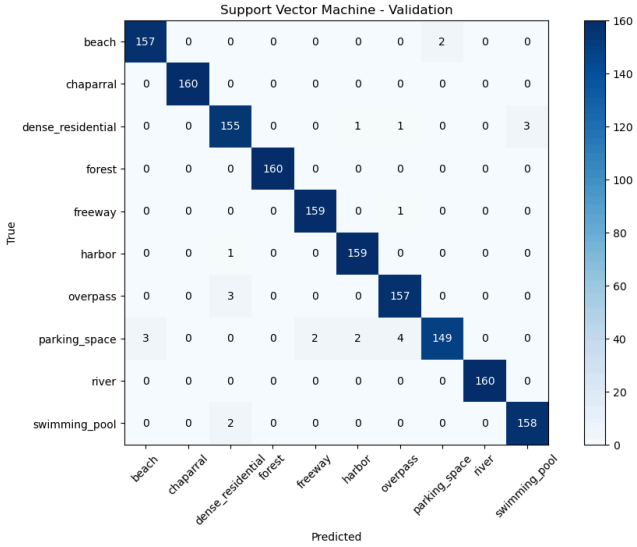
Confusion matrix for SIFT feature-set (validation)



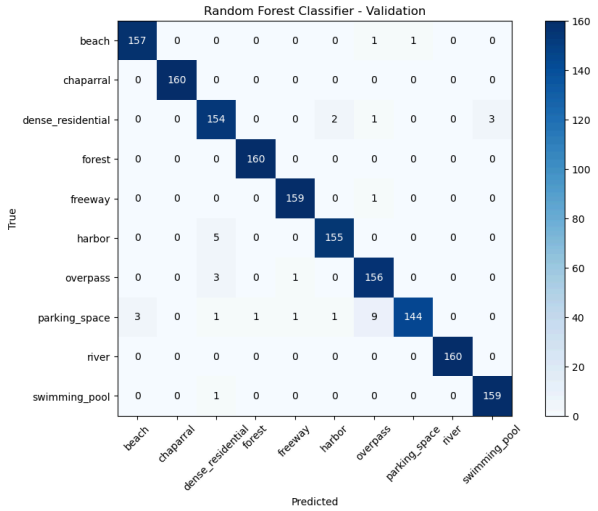


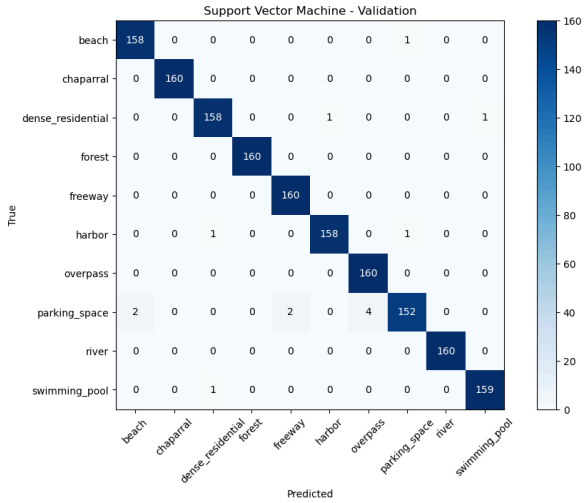
Confusion matrices for HSV feature-set (validation)



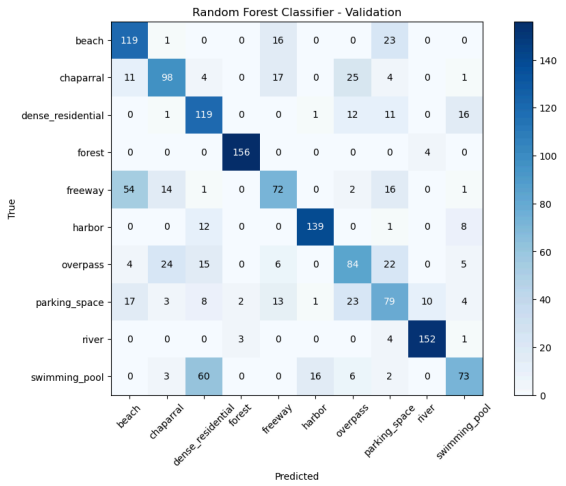


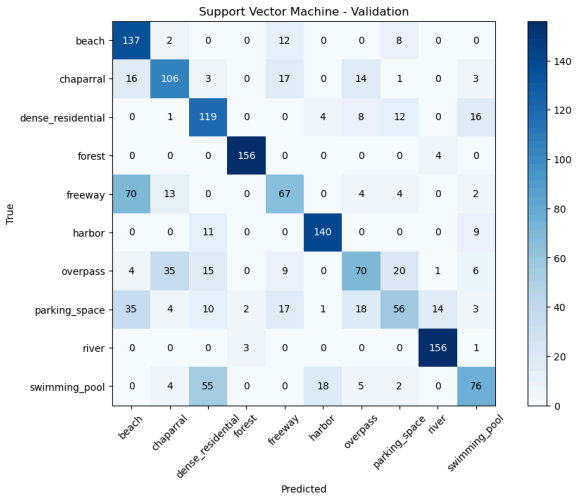
Confusion matrices for GLCM+HSV (validation)





Confusion matrices for full feature-set (validation)





Confusion matrices for HSV & GLCM feature-set (test)