P.A.N.I.C.: Pattern Analysis in Numeric and Image Contexts

Georgios Skoulidis (s6050786)[1]

Group [06]
Pattern Recognition (WMAI021-05) 2024-2025.1A
[1] Artificial Intelligence Department, University of Groningen

December 23, 2024

1 Introduction

Analyzing patterns in data can reveal valuable insights, such as detecting groups and identifying typical qualities of categories or classes. This knowledge can then be applied to new, unseen data.
The dataset is composed of images of large cats, each belonging to one of five classes (from the Big Cats Image Dataset). For dataset classification, we will compare various supervised learning methods: Support Vector Machine, Random Forest, and K-Nearest Neighbors. For clustering, we will evaluate K-Means, and Fuzzy C-Means.
These methods were selected from a table of established clustering and classification techniques to compare their performance on specific types of data.

2 Methods

2.1 Data Analysis

We first resized the images to a uniform square size, ensuring that both height and width were equal. This simplifies processing and maintains consistency.
Duplicate images were identified using an average hash function from Python's `imagehash`. If the two duplicates belong to different classes, we remove both, as we are not certain about their correct class. If they belong to the same class, we simply remove only one; 9 pairs of duplicates were found, and 14 images were removed from the dataset, resulting in 156 remaining instances.

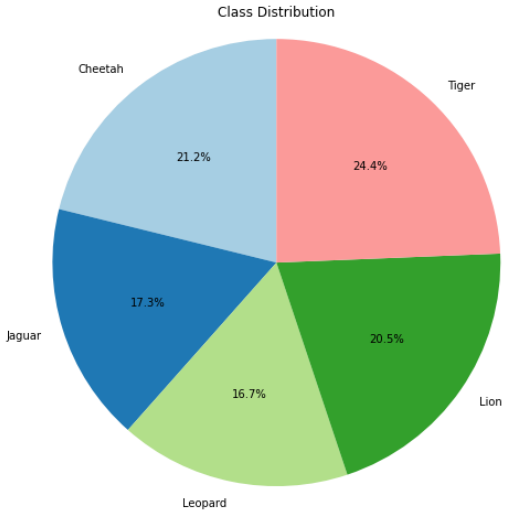


Figure 1: Class distribution after handling duplicates.

Another key aspect of our image analysis was identifying the original image sizes before they were resized in the first step.

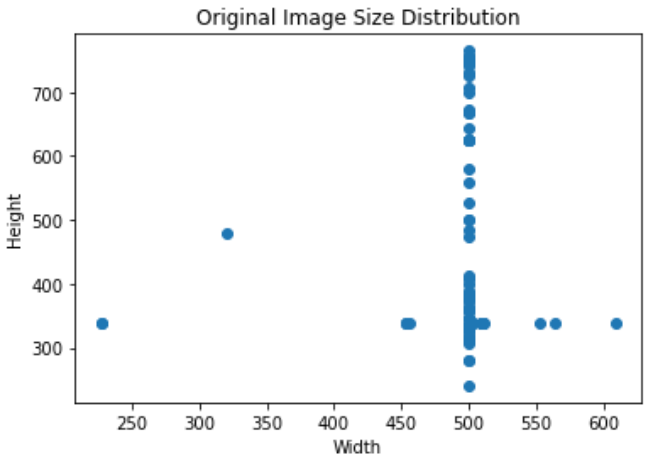
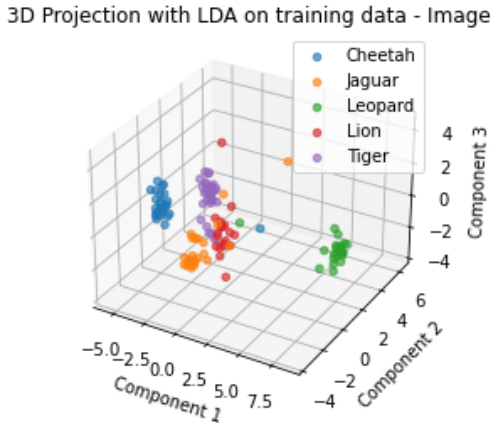
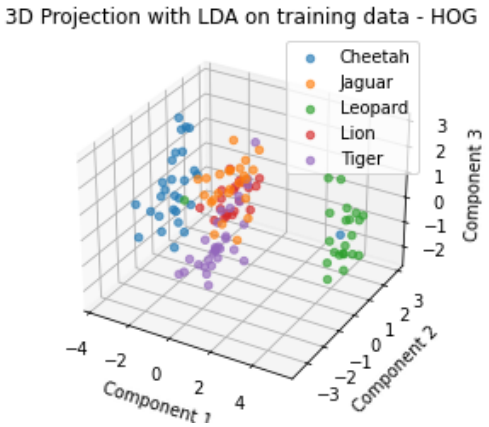


Figure 2: Original Image Size Distribution.

After we have performed data cleaning, we split into training and test sets (`80%/20%`). The test set was set aside, and we focused on the training data. A great method to visualize linear combinations of features that separate or characterize two or more classes is **Linear Discriminant Analysis (LDA)**. LDA returns **number of classes** − 1 features, making it not only useful for visualization but also a supervised approach for feature reduction and even classification.



(a) 3 LDA components in image training data.



(b) 3 LDA components in HOG training data.

Figure 3: Comparison of 3 LDA components in image and HOG feature training data.

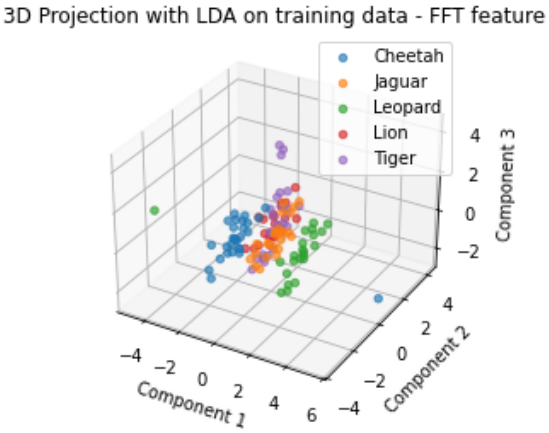


Figure 4: 3 LDA components in (non-cropped) FFT feature training data.

A potential issue with using LDA for feature reduction is the risk of overfitting. While LDA was part of our tuning process, we also employed more specific methods for image feature extraction. First, we computed the magnitude of the **Fast Fourier Transform (FFT)** with low-pass cropping, reducing dimensionality by retaining lower frequencies, which carry essential information. Second, we used the **Histogram of Oriented Gradients (HOG)**, a feature descriptor based on gradient orientation histograms in localized image regions. HOG effectively reduces the number of features. It also requires grayscale conversion of each image.

In Figures 3a, 3b, and 4, we observe that all data points are sufficiently separable, with the original image data appearing slightly more promising than the transformed data. This may be due to the original images retaining full information or because three LDA components capture limited variance in transformed data. Figure 5 confirms this, as the three LDA components for original images explain slightly more variance than those for transformed data. However, LDA indicates that all five classes remain separable across all three training sets.

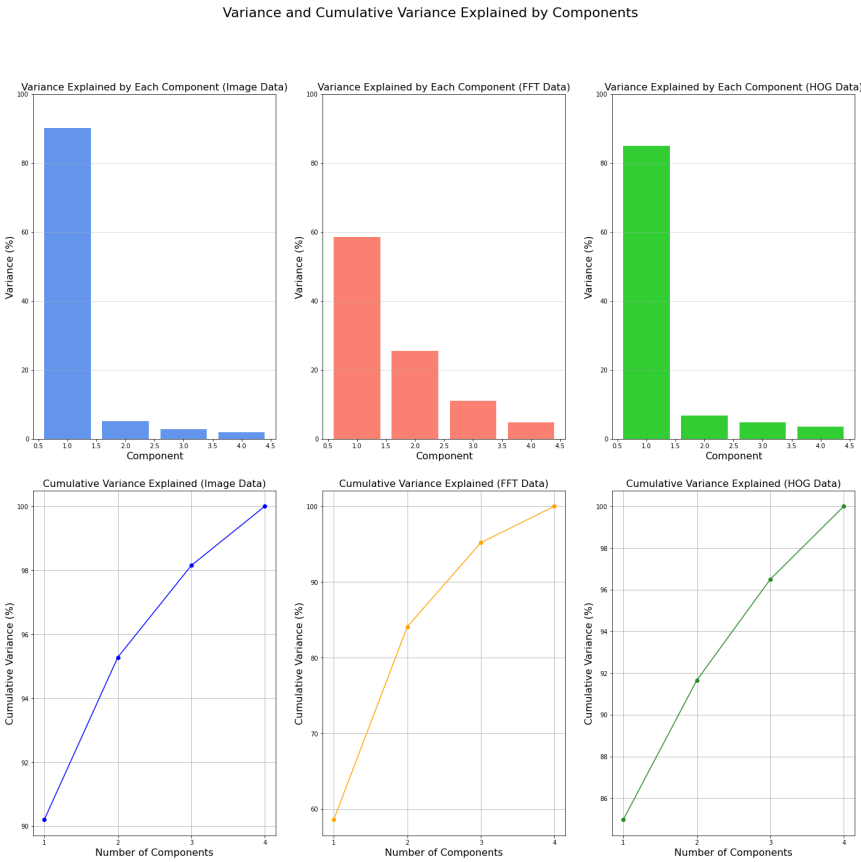


Figure 5: Variance and Cumulative Variance Explained by LDA Components.

2.2 Data Augmentation

As shown in Figure 1, we can visualize our class ratios. The class imbalance is clear, and the accuracy metric can be proved misleading when assessing a classifier. Assuming a perfectly balanced real-world class distribution, with an equal likelihood of an image belonging to any of the five classes, we aim to oversample for most of the classes with data augmentation. Applying transformations to images creates new instances, potentially enhancing performance. We apply these transformations only after splitting the training set into sub-training and validation sets within each cross-validation fold. This data augmentation pipeline is used exclusively for classification, not for clustering.
The transformations we applied for image augmentation and their specific queue are as follows:

1. **Random Horizontal Flip** ($p = 0.25$): Randomly flips the image horizontally with a probability of 25%. It adds variability to the objects' horizontal orientation in the image.

2. **Random Rotation** (degrees = 20): Randomly rotates the image by an angle from -20 to 20 degrees, helping the model become orientation-invariant.

3. **Color Jitter** (Brightness, Contrast, Saturation): Randomly adjusts brightness, contrast, and saturation by up to 25%, introducing color variability to help the model generalize to different lighting and color conditions.

2.3 Clustering

For clustering, we followed two approaches: a supervised and an unsupervised learning approach. In a real-world scenario, we typically would not have access to the labels. Since we now have them, we can also implement a supervised approach to assess our clustering algorithm based on the ground truth. The dataset will not be divided into training and test sets. We will utilize **Fuzzy C-Means (FCM)**, which assigns a probability vector to instances indicating their degree of belonging to each cluster. This is especially useful for images that are hard to distinguish, where animals can be easily confused.
For Python, we used the `fuzz.cluster.cmeans` function from `skfuzzy`. The function is called with arguments `m=2`, `error=0.005`, `maxiter=1000`, and `init=None`.

Supervised Clustering Approach To evaluate the clustering's performance, we can utilize the given labels. Therefore, we use the **Normalized Mutual Information (NMI)** metric. By passing the true labels along with the assigned cluster for each instance (the cluster with the highest probability), we can obtain a score that shows the quality of the clustering compared to the ground truth. NMI values of 0 show independence from true labels, while values of 1 indicate perfect agreement. We can tune various parameters of the FCM algorithm, such as the number of clusters k, and the input features (HOG, FFT, or Flattened Images), to identify the configurations that maximize the NMI score.

Unsupervised Clustering Approach We leave the labels aside to simulate a real-world scenario where they are not available. We will analyze the **Fuzzy Partition Coefficient (FPC)** returned from the FCM algorithm, which measures the fuzziness in the clustering predictions.
Furthermore, to assess the quality of the clustering without labels, we can compute the difference between the FPC and $\frac{1}{k}$, which represents the lowest possible coefficient for k clusters. A smaller difference indicates that the clusters are poorly defined, while a larger difference suggests that the clusters are more distinct.

2.4 Classification

The original image sizes in the dataset vary. It would be beneficial to experiment with different resizing options to determine which dimensions achieve the best results for the classification. Figure 2 shows the distribution of original image sizes in the dataset. As we can see, there is no specific height and width combination that significantly outnumbers the others.
For tuning, we use each set of features with different processing methods. We defined a validation technique to maximize generalization: Stratified 5-fold cross-validation. We chose 5 folds to avoid small validation sets due to limited data. Stratified cross-validation ensures a balanced distribution of instances per class across all folds.

For each sub-training set in each fold, we performed data augmentation by applying a series of random transformations to selected images. We experimented with two different augmentation approaches. The first involved **oversampling each class until all classes had the same number of instances as the class with the maximum instances (thus, this class was not oversampled)**. The second involved **oversampling each class until it reached 60 instances**, which was three to four times the number of existing instances in that class. This type of augmentation not only increases the number of instances but also balances the dataset, allowing us to use accuracy as our primary evaluation metric. We would like to discuss the pros and cons of applying this type

of data augmentation within each cross-validation fold. First, it is essential to apply these transformations to the image data before computing features like HOG or FFT. This step though, results in a significant computational load for each fold. Additionally, the randomness of the transformations may introduce unwanted noise, potentially hindering our ability to select the optimal model. However, applying data augmentation to the training set before cross-validation risks compromising the validation process. If an image and its transformed version end up in different sets (one in the sub-training set and the other in the validation set), they could remain highly correlated, effectively validating with training data and leading to bias. We prefer augmented images to be included only in the training set, not the validation process. Therefore, we can accept a lower validation accuracy to avoid a misleading high score that does not reflect true model performance. Our evaluation data must always consist of new, unseen samples to ensure an unbiased estimation of the model's generalization ability.

After augmenting the sub-training data, we have two options for training the classifier: we can either use the flattened images directly or extract features using HOG and FFT. For the FFT-transformed features, we validate across different **crop sizes**. Additionally, we decided to apply **ANOVA feature selection** for both the HOG features and the image data. We conducted a grid search over various crop sizes and number of features based on the data format, and we explored smaller grids around the top-performing parameter configurations. In our grid search, we also considered the hyperparameters associated with the classifiers we utilized.

We used three classifiers known for their effectiveness in image data applications, experimenting with specific parameters for each. **K-Nearest Neighbors (KNN)** was tuned for the number of neighbors (`k`), **Random Forest (RF)** was tuned for the number of estimators (`n_estimators`), and **Support Vector Machines (SVM)** was tuned for **kernel types** and **degrees** when the kernel is polynomial. For the RF classifier, we did not apply feature selection techniques like ANOVA or LDA to the HOG features or images, as the algorithm already performs feature selection.

For splitting the dataset into training and test sets, and further dividing the training set into sub-training and validation sets for cross-validation, we consistently set a specific random seed. This ensures that we obtain the same random pseudo-sequence, resulting in identical splits across different runs, which helps avoid bias and ensures consistency. The main metric for evaluating our models is the average validation accuracy across all folds.

We also repeated the tuning process for different image resize dimensions: 128x128, 256x256, and 512x512.

2.5 Ensemble

For the ensemble, we combined the three classifiers for both the numeric and image datasets, using the hyperparameters that achieved the best performance for each. For the image data, we ensured diversity by training the least accurate classifier, KNN, with FFT features instead of HOG, despite HOG yielding better accuracy. This choice aimed to enhance model diversity and complement the strengths of the other classifiers. The three models were then combined into an ensemble, using the majority decision rule for the final classification output. With each model expected to perform above chance, the ensemble technique was anticipated to improve overall results.

3 Results

3.1 Clustering

When we assessed clustering quality using the FCP, we found that the FFT feature data yielded more fuzziness compared to HOG features or raw images. When we assessed clustering performance using NMI with known labels, we observed that HOG features produced the highest NMI scores.

Number of Clusters (k)	**Optimal Crop Size**	**FPC**	**Increase % from** $1/k$
2	50	0.9894	97.89%
3	5	0.5849	75.48%
4	5	0.3952	58.07%
5	5	0.3885	94.26%
6	5	0.2918	75.07%
7	5	0.2429	70.05%
8	30	0.2256	80.51%
9	5	0.1824	64.20%
10	25	0.1950	95.02%
11	15	0.1524	67.60%
12	5	0.1449	73.92%

Table 1: Best models for clustering with fuzzy c-means (FCM) based on FPC values for different k, using FFT features for crop sizes up to 50 with a step size of 5. The percentage increase represents the difference from the lowest possible FPC $1/k$.

For $k = 2$, the configuration yields the highest FPC increase of 97.89%. The corresponding clusters are distributed as follows:

- **Cluster 0:** 3 images
- **Cluster 1:** 153 images

It is clear that the clustering is highly unbalanced, with Cluster 0 containing only 3 images. This uneven distribution makes us doubt the usefulness of this configuration. We have to consider exploring more.
For $k = 10$, the second-best configuration yields a percentage increase of 95.02%. The cluster distribution is as follows:

- **Cluster 0:** 0 images
- **Cluster 1:** 23 images
- **Cluster 2:** 82 images
- **Cluster 3:** 0 images
- **Cluster 4:** 2 images
- **Cluster 5:** 2 images
- **Cluster 6:** 0 images
- **Cluster 7:** 23 images

- **Cluster 8:** 24 images
- **Cluster 9:** 0 images

Several clusters remain empty. This indicates that while the fuzzy c-means algorithm does distribute images across multiple clusters, it fails to make use of all the available ones.
The third-best, for $k = 5$, presents a more balanced distribution of images across clusters, with an FPC increase of 94.26%. The clusters are as follows:

- **Cluster 0:** 38 images
- **Cluster 1:** 14 images
- **Cluster 2:** 3 images
- **Cluster 3:** 69 images
- **Cluster 4:** 32 images

This configuration successfully uses all 5 clusters. Almost all clusters contain a significant number of images except Cluster 2. The relative balance across clusters makes the $k = 5$ configuration the strongest candidate.
We also used the labeled data to check on the results of the configuration with $k = 5$.

	Cluster 1	Cluster 2	Cluster 3	Cluster 4	Cluster 5
Cheetah	9	3	0	14	7
Jaguar	5	1	0	14	7
Leopard	9	0	0	11	6
Lion	10	1	1	15	5
Tiger	5	9	2	15	7

Table 2: Cluster results with actual categories.

When using labels with NMI, we again tuned for different values of k. We found that our best model corresponds to 300 selected HOG features with $k = 12$, yielding an NMI of 0.2269.
We can see the **cluster distribution** below:

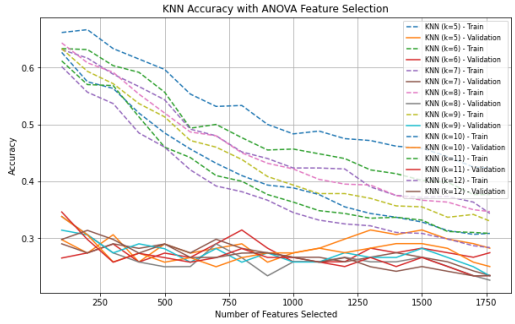
- Cluster 0: 13 images
- Cluster 1: 0 images
- Cluster 2: 55 images
- Cluster 3: 0 images
- Cluster 4: 7 images
- Cluster 5: 0 images
- Cluster 6: 0 images
- Cluster 7: 6 images
- Cluster 8: 63 images

- Cluster 9: 0 images
- Cluster 10: 12 images
- Cluster 11: 0 images

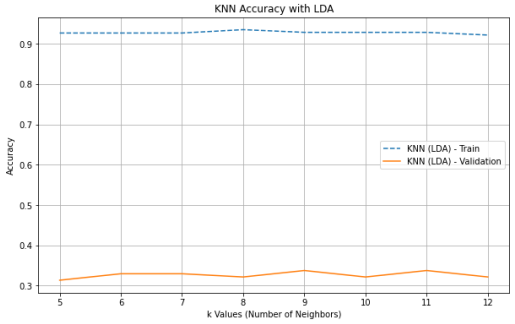
Six clusters have remained empty. This suggests that k may be too high and clustering is too difficult for the dataset.

3.2 Classification

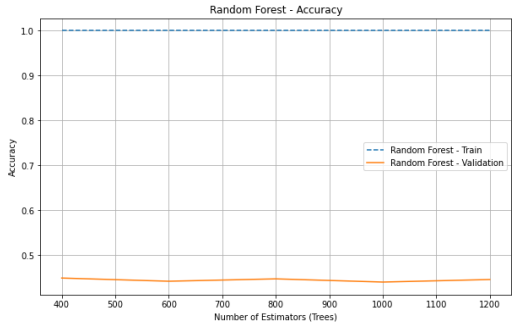
We observed no significant performance differences across the three validated image sizes, so we use 128x128 images to reduce computational complexity. Flattened images and FFT features performed worse than HOG features. As for data augmentation, oversampling to 60 images per class was less effective than oversampling until the number of instances matched those of the majority class. Applying LDA to HOG [1] features and image data across various classifiers did not yield satisfactory results either. The classifier that demonstrated the best performance was the SVM [2]. Among the different kernel types, the polynomial kernel outperformed the others, with a polynomial degree of 10 achieving the highest validation accuracy.
Figure 6 illustrates some of the models we trained and validated during the tuning process using HOG features extracted from 128x128 images. SVMs outperform both KNN and RF classifiers.



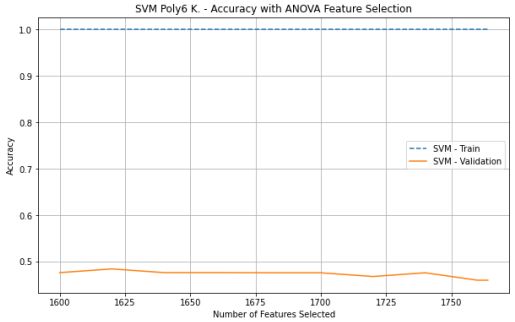
(a) Average Cross-Validation Accuracy for KNN for different k values with LDA applied in HOG features.



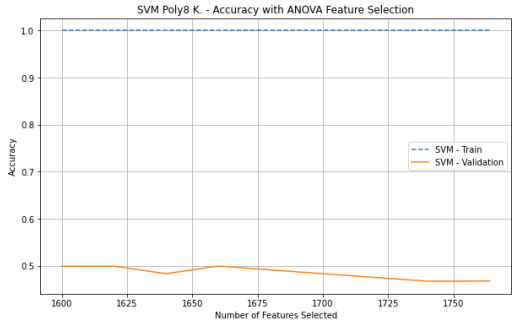
(b) Average Cross-Validation Accuracy for KNN for different HOG selected features with ANOVA.



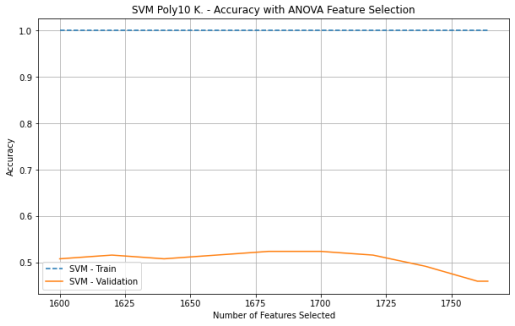
(c) Average Cross-Validation Accuracy for RF for different number of estimators using HOG features.



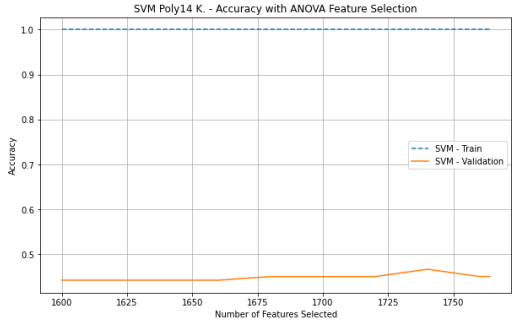
(d) Average Cross-Validation Accuracy for SVM with polynomial kernel of degree 6, for HOG selected features with ANOVA.



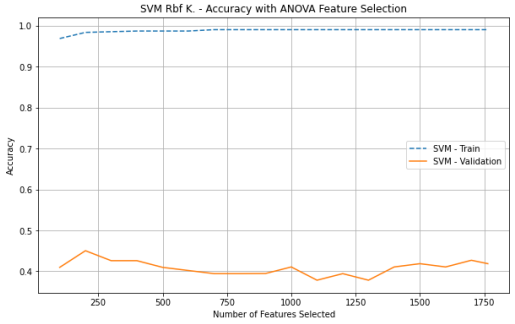
(e) Average Cross-Validation Accuracy for SVM with polynomial kernel of degree 8, for HOG selected features with ANOVA.



(f) Average Cross-Validation Accuracy for SVM with polynomial kernel of degree 10, for HOG selected features with ANOVA.



(g) Average Cross-Validation Accuracy for SVM with polynomial kernel of degree 14, for HOG selected features with ANOVA.



(h) Average Cross-Validation Accuracy for SVM with Radial Basis Function kernel, for HOG selected features with ANOVA.

Figure 6: Model Optimization; each stands for HOG features, extracted from 128x128 images. Data augmentation is performed and we plot the Average Accuracy for all folds.

During the last phase of the tuning process for HOG data, we focused on SVMs with a polynomial kernel and degree of 10, on an interval of interest between 1650 and 1710 features. We trained and evaluated the models using steps of 10 within this interval.

Features	Training Accuracy	Validation Accuracy
1650	1.0	0.465
1660	1.0	0.46472
1670	1.0	0.46486
1680	1.0	0.46513
1690	1.0	0.46612
1700	1.0	0.46902
1710	1.0	0.46918
Full HOG (1764)	1.0	0.46227

Table 3: Training and Validation Results for SVM with Polynomial Kernel of degree 10.

We also tuned for FFT features later on and got poor results. The whole tuning process can be seen in Section A. Finally, we have our best pipeline, which is in Table 4. We can now train on the entire dataset and evaluate our final model using the test set, which we have not used yet.

Pipeline Steps
Resize images to 128x128
Oversample with data augmentation until all class instances match the instances of the majority class
Apply HOG to extract features
Select 1710 out of the 1764 extracted features using ANOVA
Train an SVM with a polynomial kernel and a degree of 10

Table 4: Optimal Pipeline Steps

We found that the final **Test Accuracy** is **0.2812**.

3.3 Ensemble

We ensemble an SVM with a polynomial kernel of degree 10 using 1,710 selected HOG features (optimal model), a Random Forest with 250 estimators using the full set of HOG features, and a KNN using reduced FFT features with a crop size of 50. Figures 7 and 8 provide the accuracies yielded over 20 cross-validation runs.

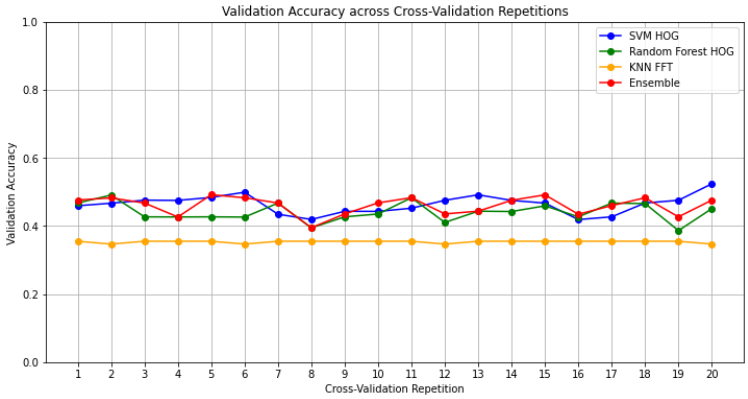


Figure 7: Line Plot of Validation Accuracy for SVM, RF, KNN, and their Ensemble Model across 20 Stratified 5-Fold Cross-Validation Runs.

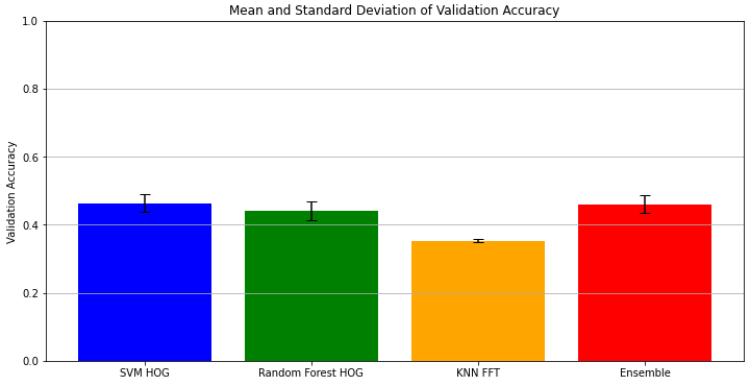


Figure 8: Average Validation Accuracy for SVM, RF, KNN, and their Ensemble Model over 20 Stratified 5-Fold Cross-Validation Runs.

The results in Table 5 show that the SVM with a mean accuracy of 0.4635 remains our top-performing model. The ensemble, while close, did not outperform the SVM, possibly due to differences in feature sets and the fact that combining models with varying performance levels (e.g., KNN with lower accuracy) may not have effectively leveraged their strengths.

Model	**Mean**	**Std. Dev.**
RF	0.4410	0.0271
KNN	0.3533	0.0033
SVM	**0.4635**	**0.0264**
Ensemble	0.4599	0.0261

Table 5: Means and Standard Deviations for SVM, RF, KNN, and Ensemble Models.

4 Discussions and/or conclusions

Clustering The analysis revealed notable differences in clustering quality based on feature types. FFT features introduced more *fuzziness* in clustering when assessed with the Fuzzy Partition Coefficient (FPC) compared to HOG features and raw image data. However, when evaluated using Normalized Mutual Information (NMI), HOG features yielded consistently higher scores, indicating that they better preserve structural information aligned with the ground truth.
$k = 5$ with FFT features (crop size of 5) yields a balanced distribution, with most clusters containing a substantial number of images, while fuzziness remains high. Therefore, it is our preferred choice for the best clustering model.

$k = 12$ with HOG features had the best NMI score, but several clusters remained empty. This makes us believe that there are plenty of challenges for clustering this image dataset.

Classification The final **Test Accuracy** of **0.2812** is higher than the random classification accuracy for 5 classes (0.2), but our model still struggles to distinguish between them. The difference between the low test accuracy and the higher cross-validation accuracy could be due to the stochastic nature of the data augmentation, which may have introduced considerable noise into the model training for each validation fold.
We should also highlight the importance of having a sufficient number of instances. Our dataset contains only 156 images, resulting in limited instances for each subset (training, validation, and test). This lack of data means our model does not train adequately, and we have insufficient samples to validate its performance. It is even more challenging given that we perform classification for five categories. Even the test accuracy might be misleading when we have such a tiny testset.

References

[1] Seung-Hyun Lee, MinSuk Bang, Kyeong-Hoon Jung, and Kang Yi. An efficient selection of hog feature for svm classification of vehicle. pages 1–2, 06 2015.

[2] Mohammadreza Sheykhmousa, Masoud Mahdianpari, Hamid Ghanbari, Fariba Mohammadimanesh, Pedram Ghamisi, and Saeid Homayouni. Support vector machine versus random forest for remote sensing image classification: A meta-analysis and systematic review. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 13:6308–6325, 2020.

A Supplementary materials

The supplementary materials contain additional images relevant to the fine-tuning process of the image data classifier. A lot of effort was put into trying to increase the performance for classification, so we felt it was good to provide the results of this process even if it is not a part of the main report.

A.1 First Tuning Phase for Big Cats Classification; KNN Vs SVM

A.1.1 128x128 Images

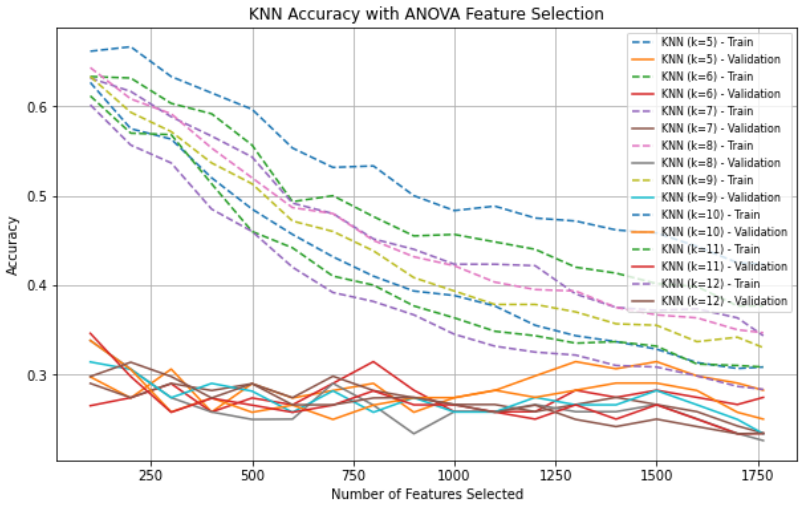


Figure 9: Accuracy with respect to the number of HOG-selected features for KNN, showing curves for different values of K. Images used were 128x128. The classifier was trained with data augmentation, adjusting class instances to match the majority class.

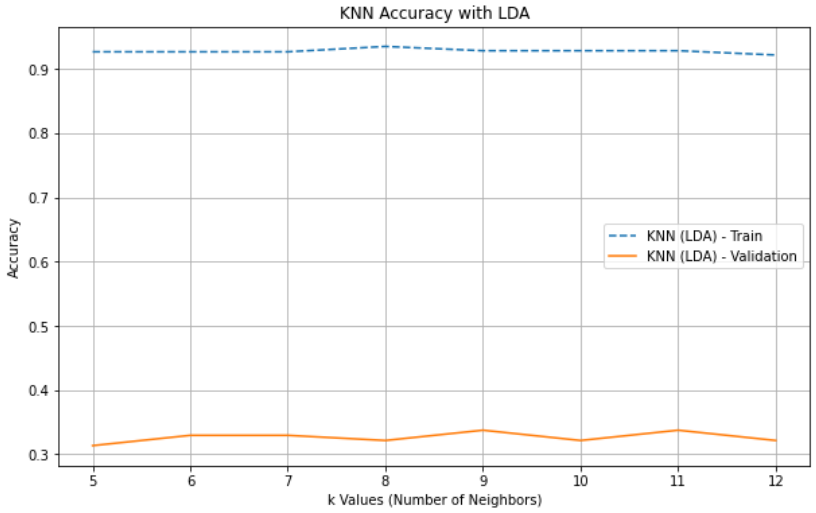


Figure 10: Accuracy with respect to KNN clusters for LDA on HOG features. Images used were 128x128. The classifier was trained with data augmentation, adjusting class instances to match the majority class.

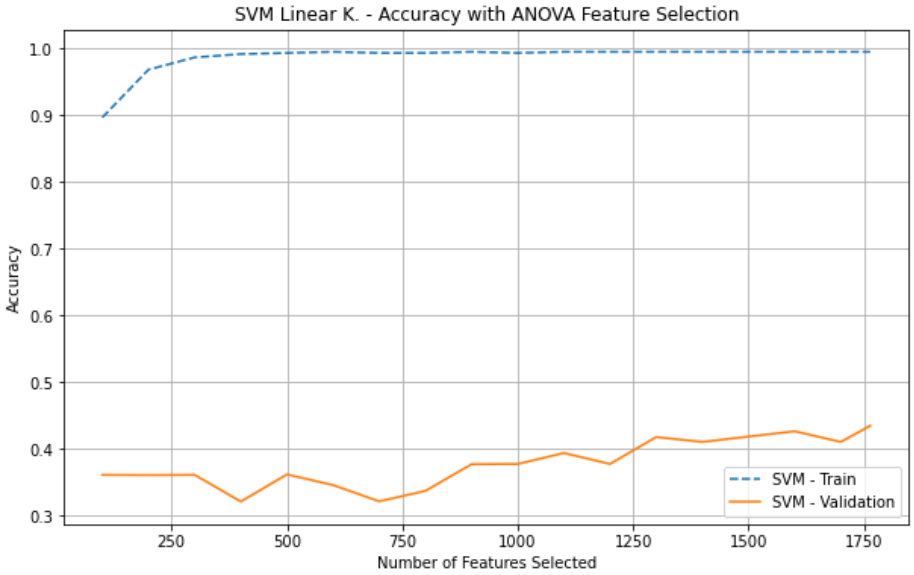


Figure 11: Accuracy with respect to the number of HOG-selected features for SVM with a linear kernel. Images used were 128x128. The classifier was trained with data augmentation, adjusting class instances to match the majority class.

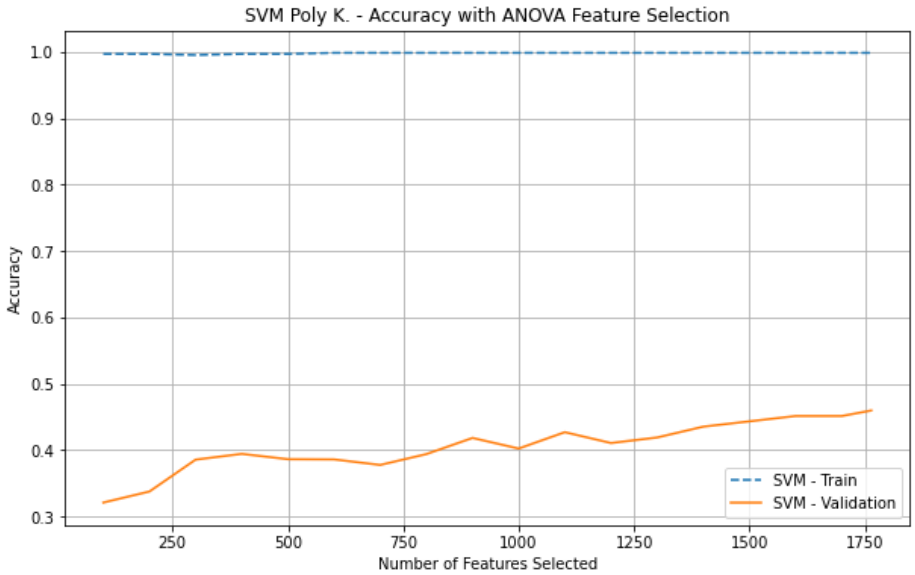


Figure 12: Accuracy with respect to the number of HOG-selected features for SVM with a polynomial kernel of degree 3. Images used were 128x128. The classifier was trained with data augmentation, adjusting class instances to match the majority class.

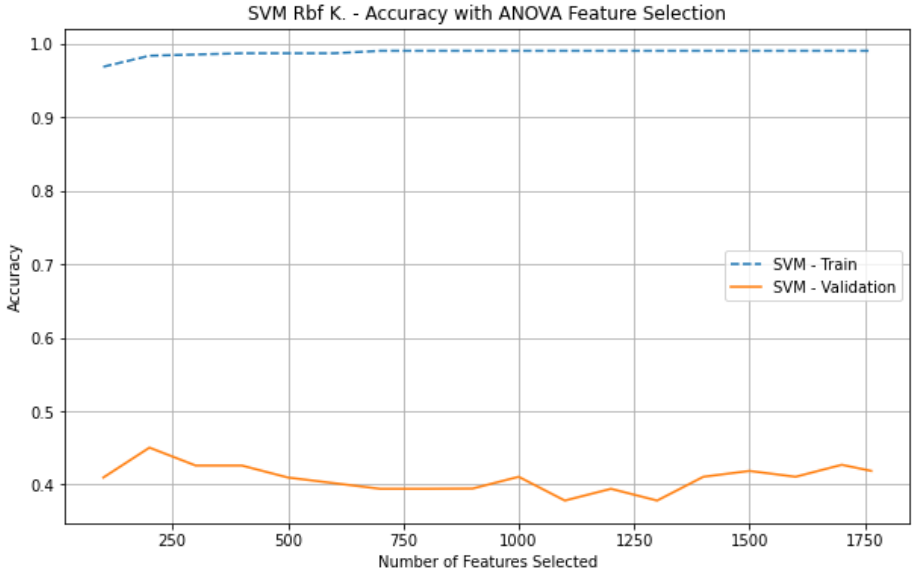


Figure 13: Accuracy with respect to the number of HOG-selected features for SVM with a Radial basis function kernel. Images used were 128x128. The classifier was trained with data augmentation, adjusting class instances to match the majority class.

A.1.2 256x256 Images

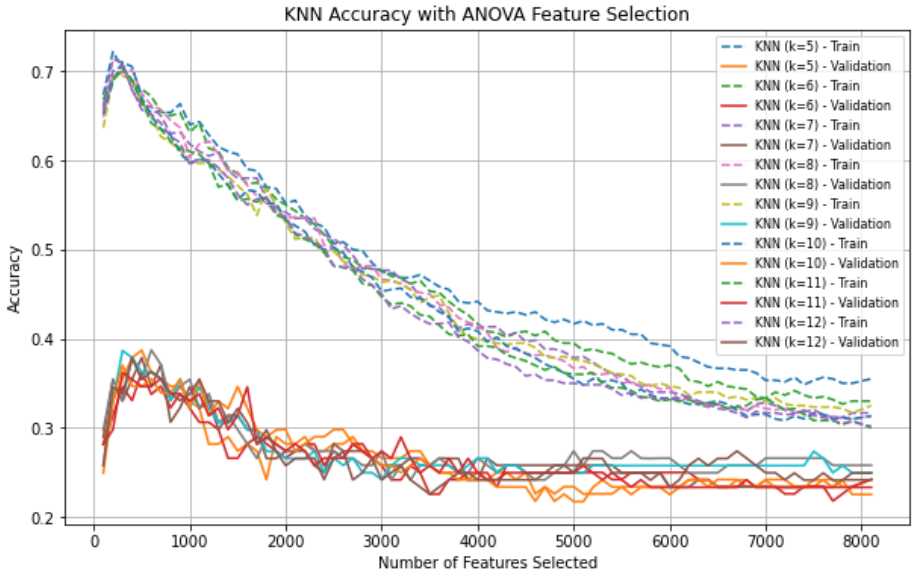


Figure 14: Accuracy with respect to the number of HOG-selected features for KNN, showing curves for different values of K. Images used were 256x256. The classifier was trained with data augmentation, adjusting class instances to match the majority class.

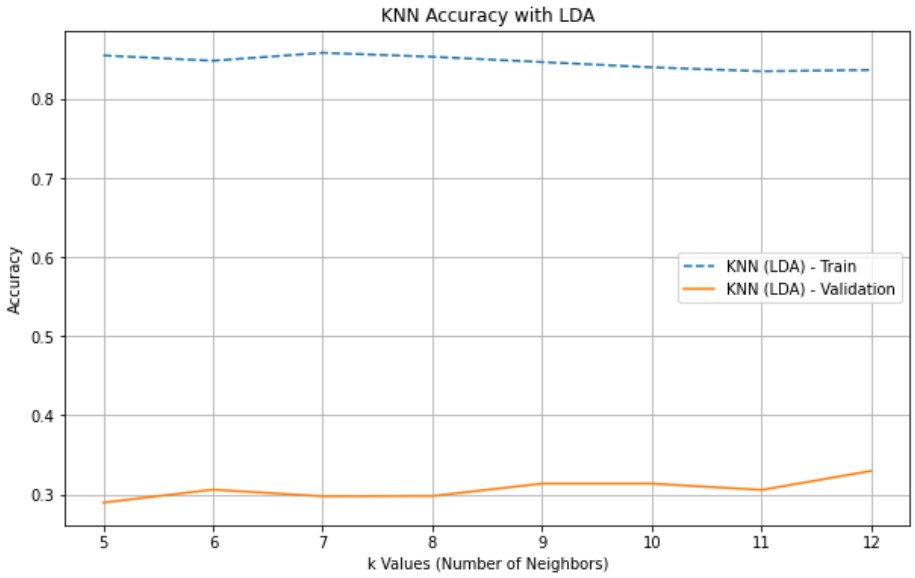


Figure 15: Accuracy with respect to KNN clusters for LDA on HOG features. Images used were 256x256. The classifier was trained with data augmentation, adjusting class instances to match the majority class.

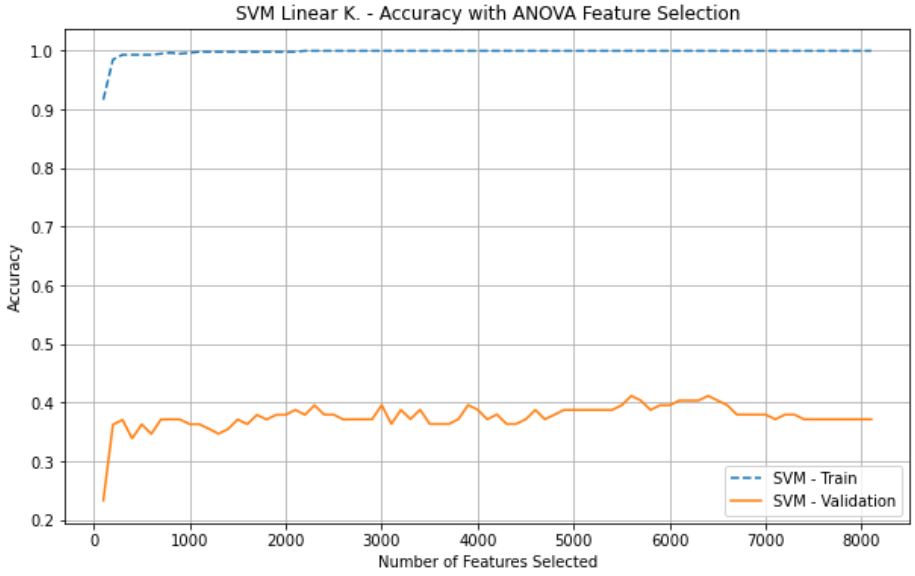


Figure 16: Accuracy with respect to the number of HOG-selected features for SVM with a linear kernel. Images used were 256x256. The classifier was trained with data augmentation, adjusting class instances to match the majority class.

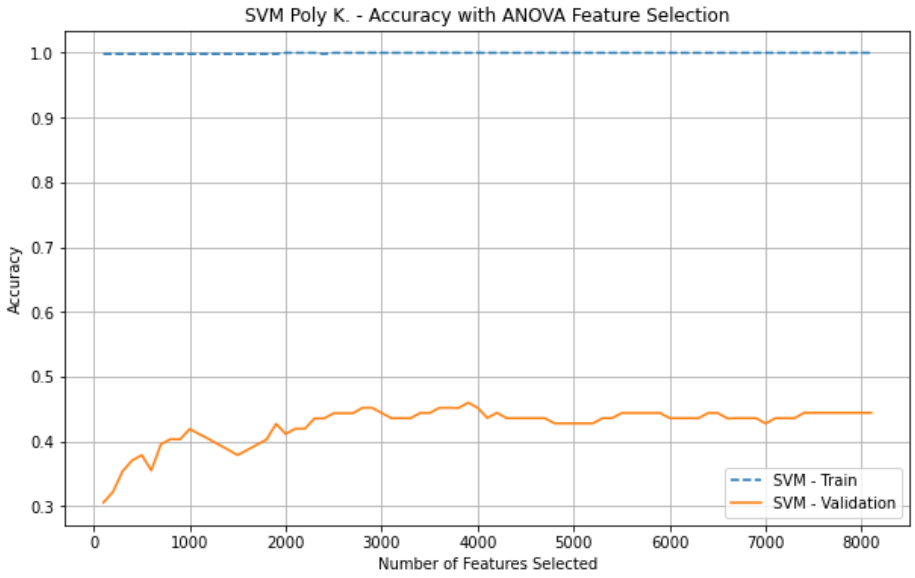


Figure 17: Accuracy with respect to the number of HOG-selected features for SVM with a polynomial kernel of degree 3. Images used were 256x256. The classifier was trained with data augmentation, adjusting class instances to match the majority class.

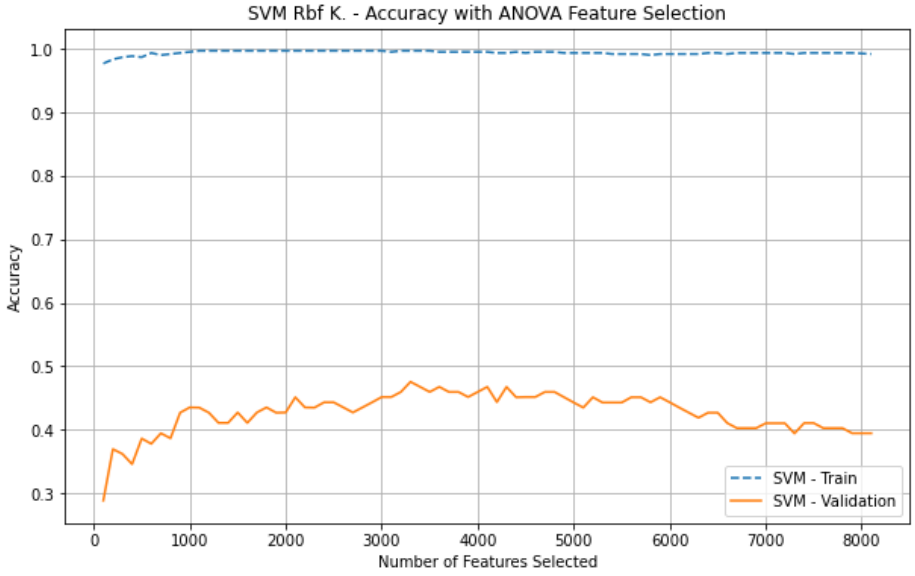


Figure 18: Accuracy with respect to the number of HOG-selected features for SVM with a Radial basis function kernel. Images used were 128x128. The classifier was trained with data augmentation, adjusting class instances to match the majority class.

A.1.3 512x512

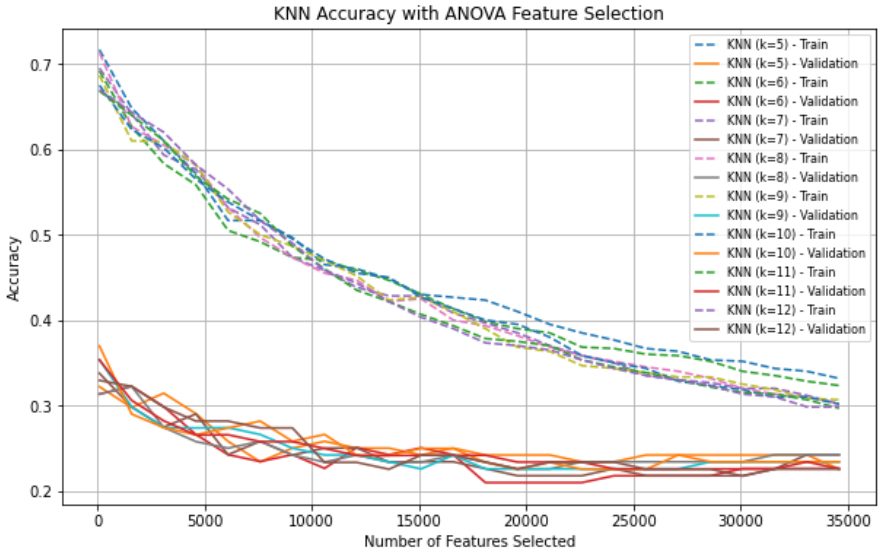


Figure 19: Accuracy with respect to the number of HOG-selected features for KNN, showing curves for different values of K. Images used were 512x512. The classifier was trained with data augmentation, adjusting class instances to match the majority class.

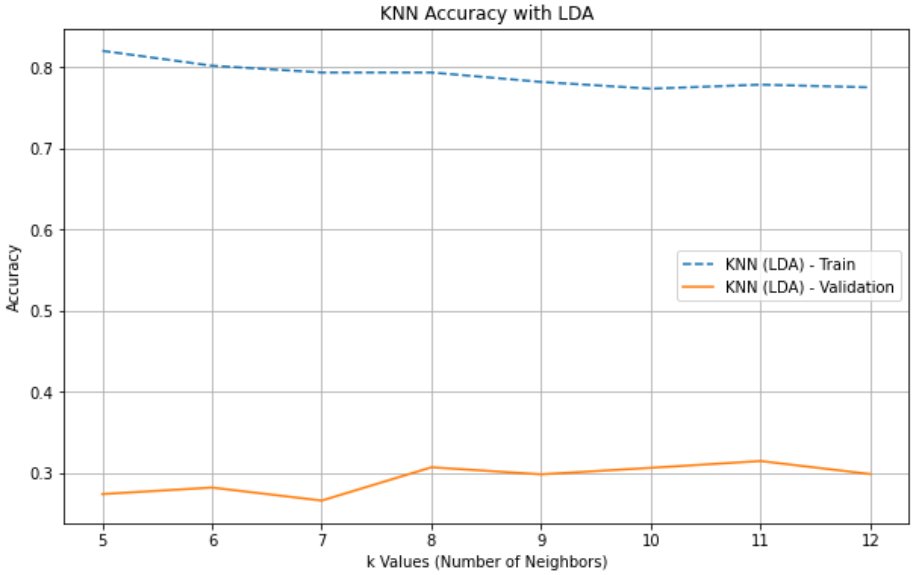


Figure 20: Accuracy with respect to KNN clusters for LDA on HOG features. Images used were 512x512. The classifier was trained with data augmentation, adjusting class instances to match the majority class.

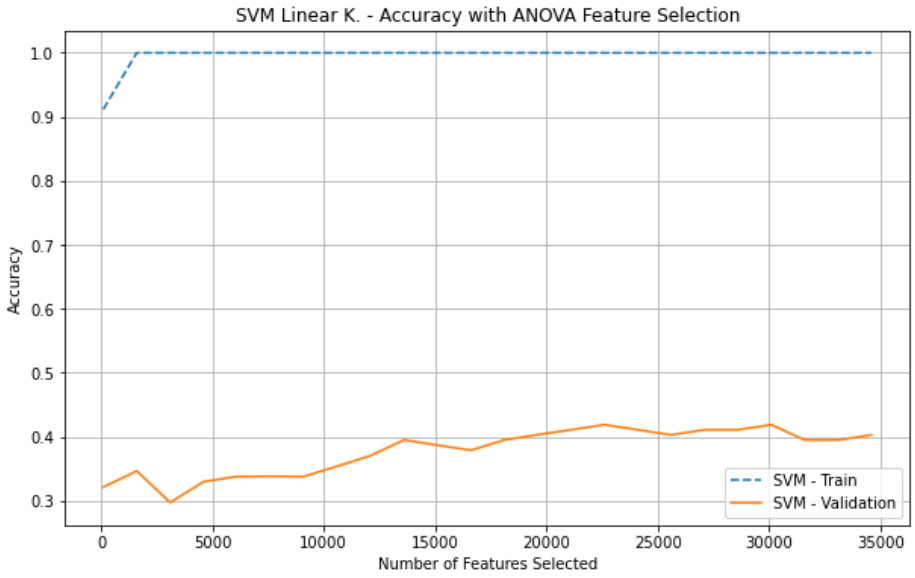


Figure 21: Accuracy with respect to the number of HOG-selected features for SVM with a linear kernel. Images used were 512x512. The classifier was trained with data augmentation, adjusting class instances to match the majority class.

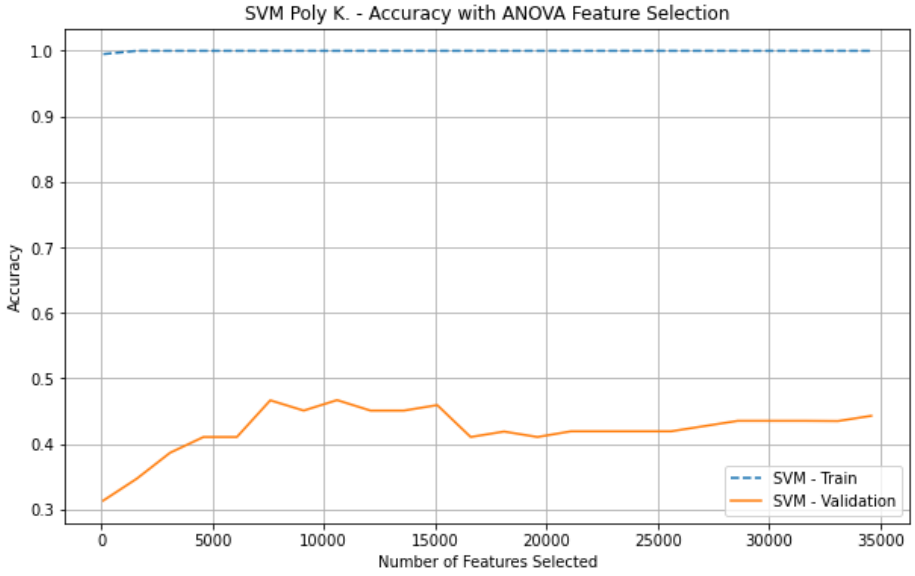


Figure 22: Accuracy with respect to the number of HOG-selected features for SVM with a polynomial kernel of degree 3. Images used were 512x512. The classifier was trained with data augmentation, adjusting class instances to match the majority class.

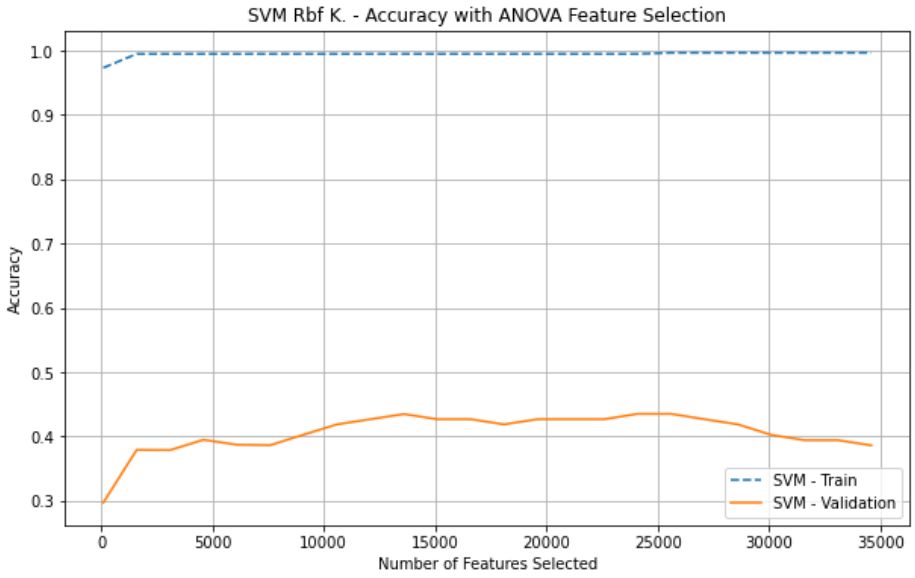


Figure 23: Accuracy with respect to the number of HOG-selected features for SVM with a Radial basis function kernel. Images used were 512x512. The classifier was trained with data augmentation, adjusting class instances to match the majority class.

A.2 Second Tuning Phase for Big Cats Classification, SVM (Low-degree Polynomial Kernels) Vs RF

A.2.1 128x128

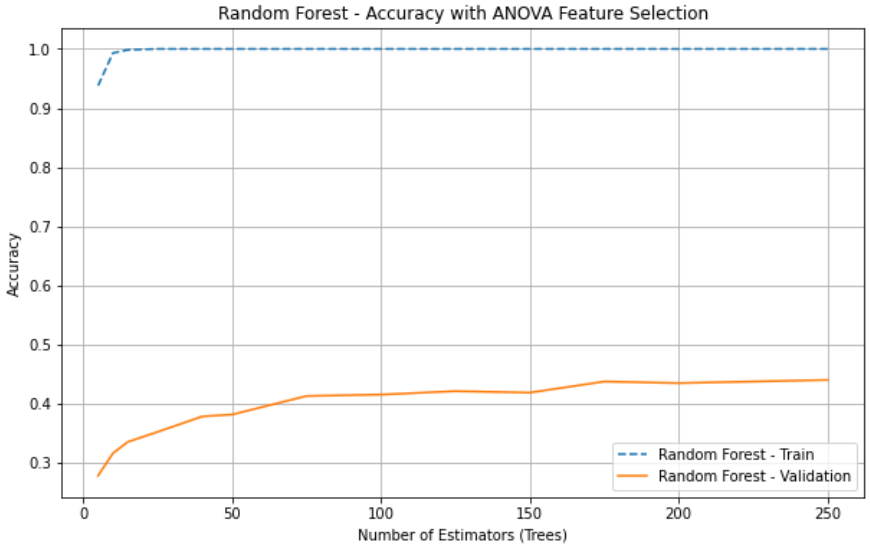


Figure 24: Accuracy with respect to the number of estimators for Random Forest on HOG features. Images used were 128x128. The classifier was trained with data augmentation, adjusting class instances to match the majority class.

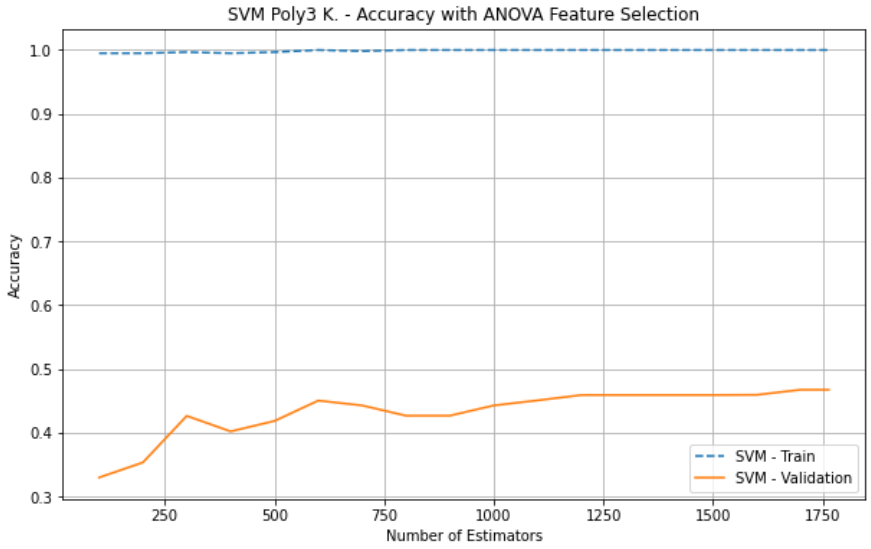


Figure 25: Accuracy with respect to the number of HOG-selected features for SVM with a polynomial kernel of degree 3. Images used were 128x128. The classifier was trained with data augmentation, adjusting class instances to match the majority class.

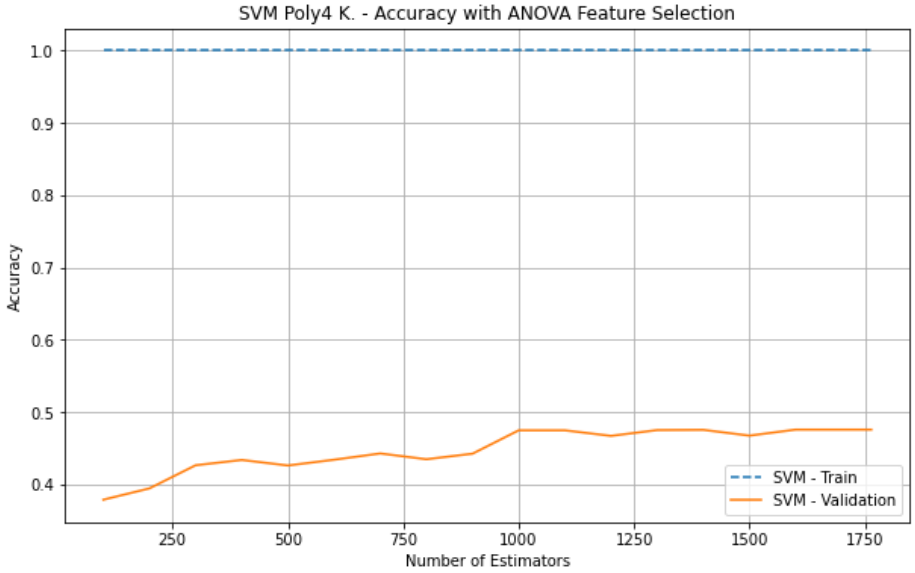


Figure 26: Accuracy with respect to the number of HOG-selected features for SVM with a polynomial kernel of degree 4. Images used were 128x128. The classifier was trained with data augmentation, adjusting class instances to match the majority class.

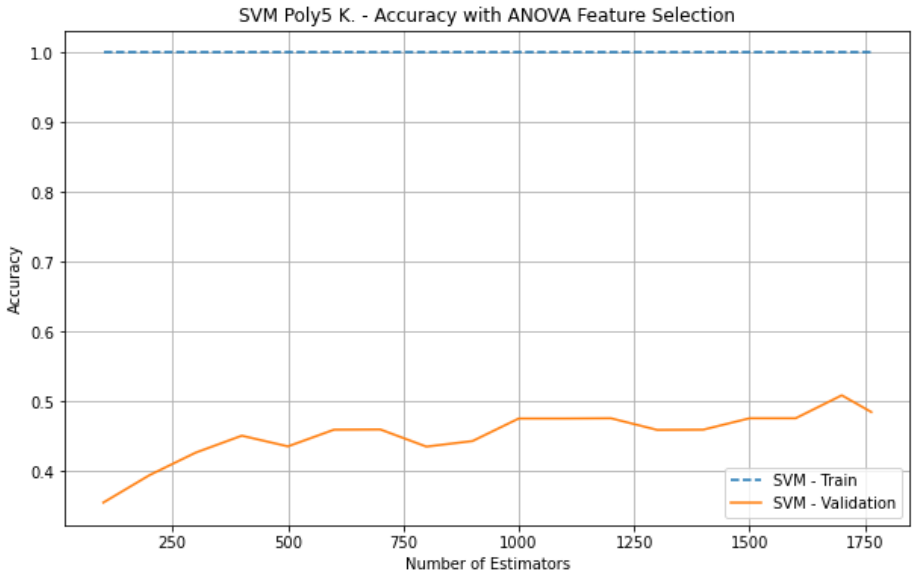


Figure 27: Accuracy with respect to the number of HOG-selected features for SVM with a polynomial kernel of degree 5. Images used were 128x128. The classifier was trained with data augmentation, adjusting class instances to match the majority class.

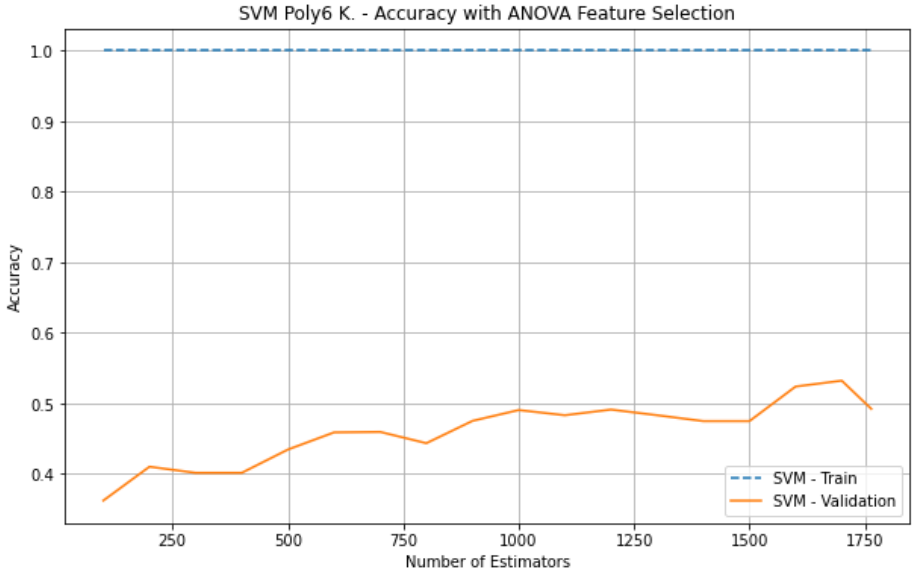


Figure 28: Accuracy with respect to the number of HOG-selected features for SVM with a polynomial kernel of degree 6. Images used were 128x128. The classifier was trained with data augmentation, adjusting class instances to match the majority class.

A.2.2 256x256

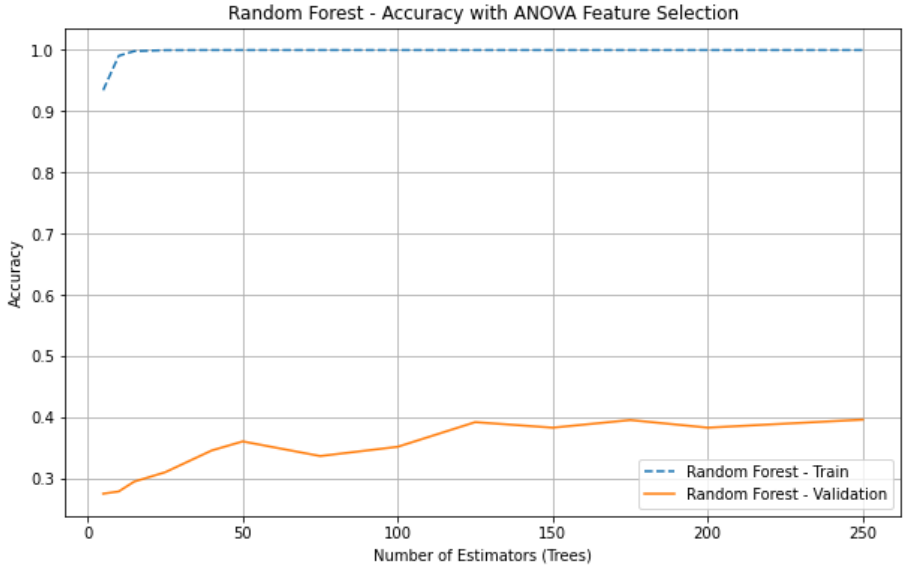


Figure 29: Accuracy with respect to the number of estimators for Random Forest on HOG features. Images used were 256x256. The classifier was trained with data augmentation, adjusting class instances to match the majority class.

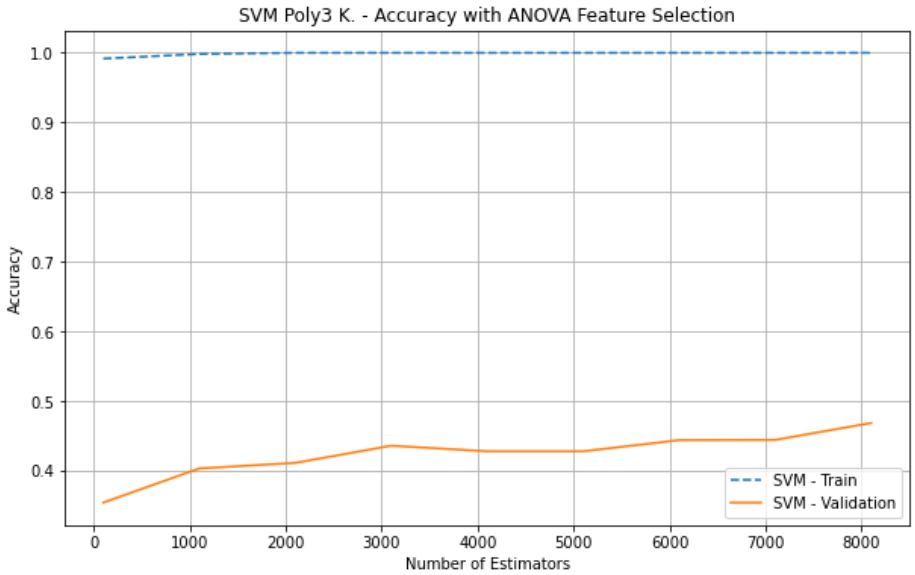


Figure 30: Accuracy with respect to the number of HOG-selected features for SVM with a polynomial kernel of degree 3. Images used were 256x256. The classifier was trained with data augmentation, adjusting class instances to match the majority class.

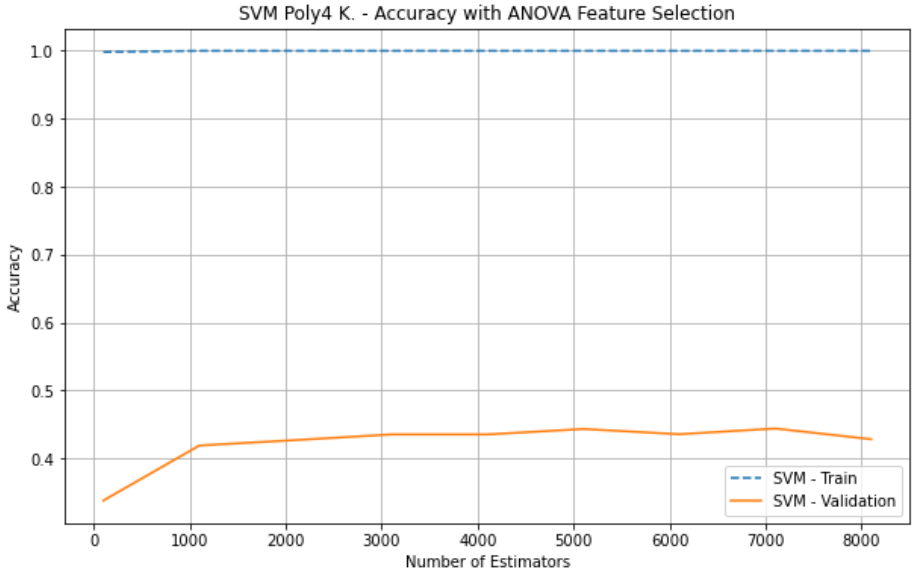


Figure 31: Accuracy with respect to the number of HOG-selected features for SVM with a polynomial kernel of degree 4. Images used were 256x256. The classifier was trained with data augmentation, adjusting class instances to match the majority class.

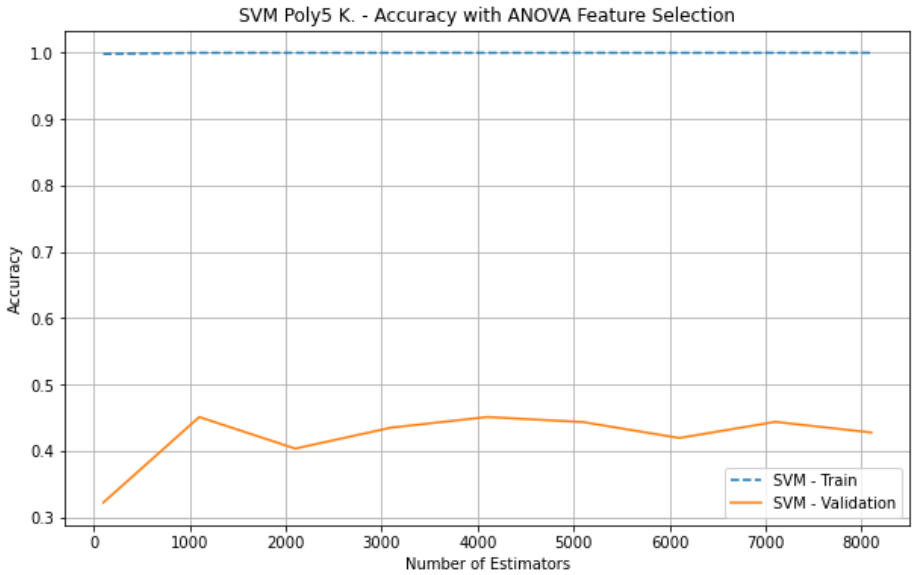


Figure 32: Accuracy with respect to the number of HOG-selected features for SVM with a polynomial kernel of degree 5. Images used were 256x256. The classifier was trained with data augmentation, adjusting class instances to match the majority class.

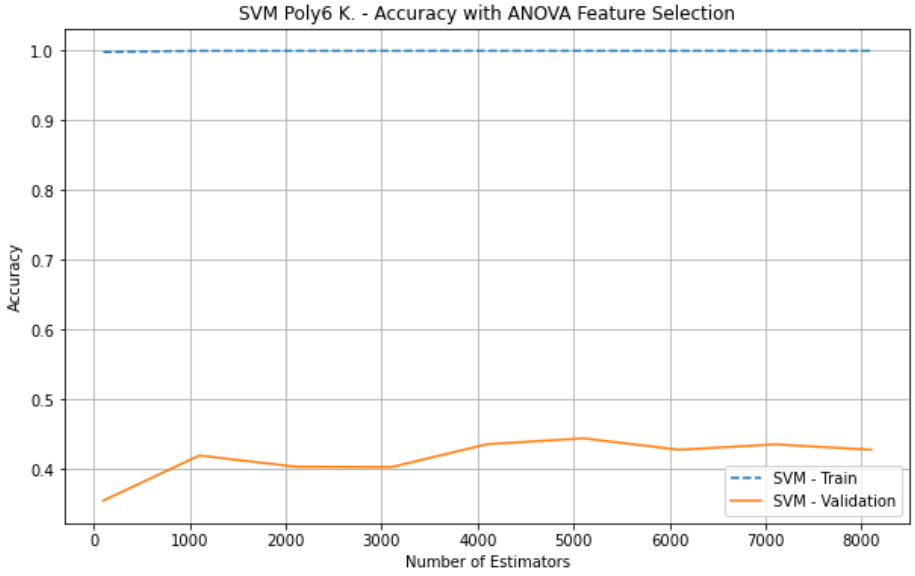


Figure 33: Accuracy with respect to the number of HOG-selected features for SVM with a polynomial kernel of degree 6. Images used were 256x256. The classifier was trained with data augmentation, adjusting class instances to match the majority class.

A.3 Third Tuning Phase for Big Cats Classification; SVM (High-degree Polynomial Kernels) Vs Random Forest & Data Augmentation Methods Comparison. Only 128x128 images.

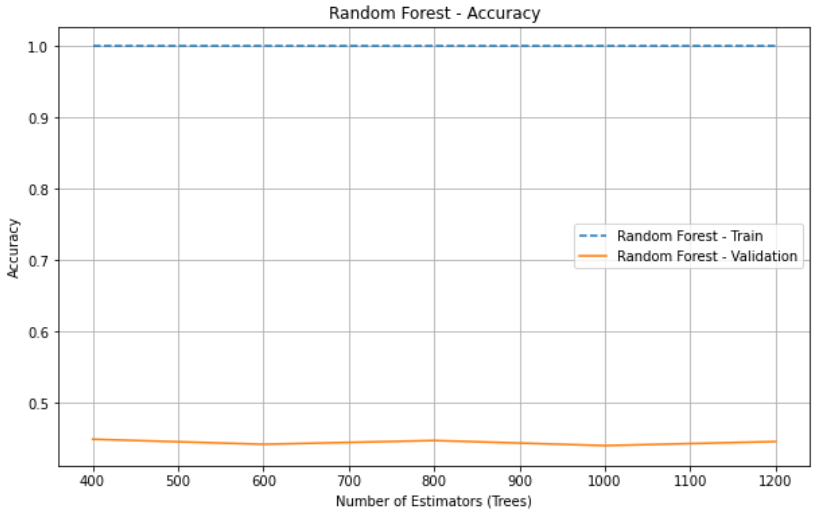


Figure 34: Accuracy with respect to the number of estimators for Random Forest on HOG features. We tuned for more estimators, also with larger steps to improve computational speed. Images used were 128x128. The classifier was trained with data augmentation, adjusting class instances to match the majority class.

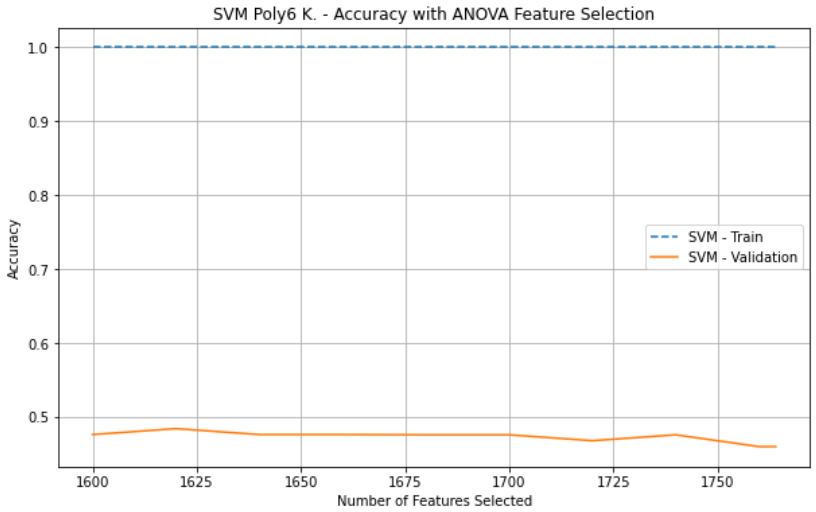


Figure 35: Accuracy with respect to the number of HOG-selected features for SVM with a polynomial kernel of degree 6. Images used were 128x128. The classifier was trained with data augmentation, adjusting class instances to match the majority class.

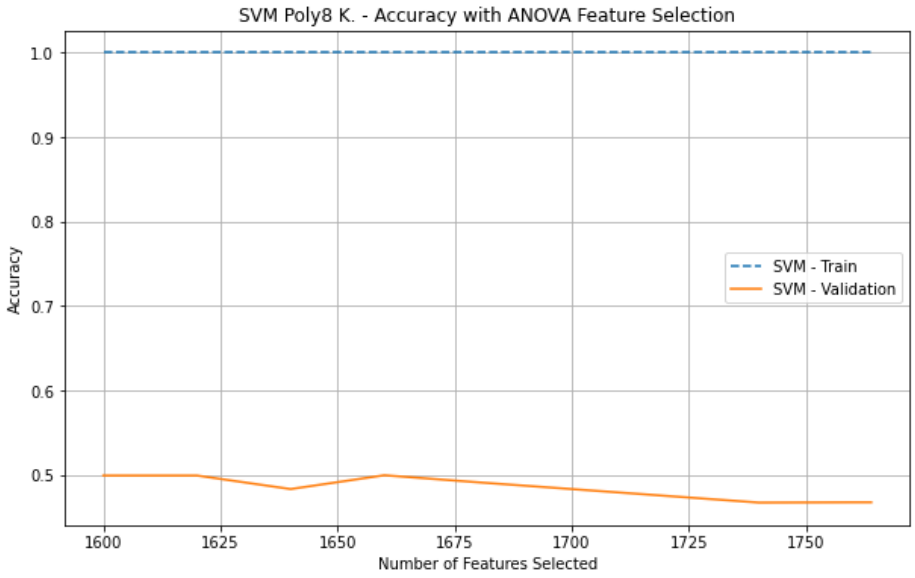


Figure 36: Accuracy with respect to the number of HOG-selected features for SVM with a polynomial kernel of degree 8. Images used were 128x128. The classifier was trained with data augmentation, adjusting class instances to match the majority class.

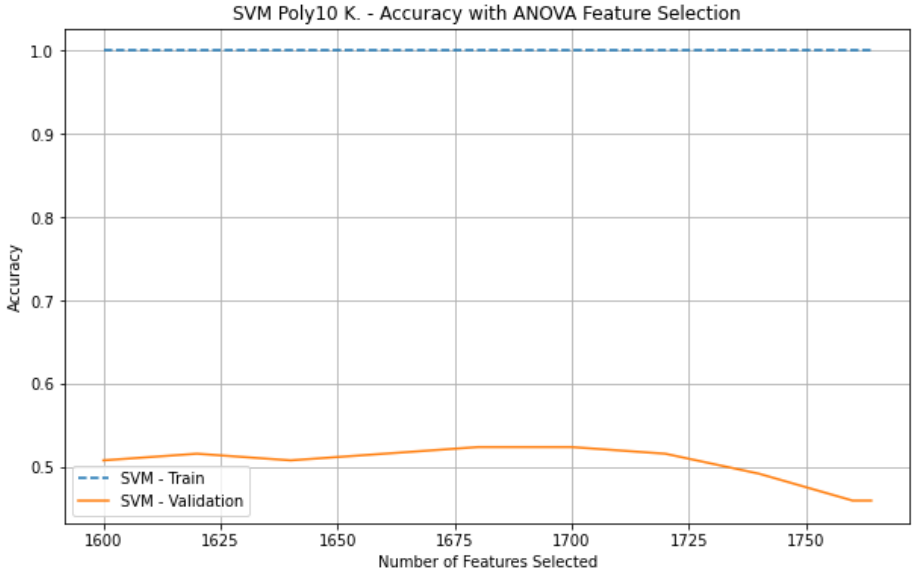


Figure 37: Accuracy with respect to the number of HOG-selected features for SVM with a polynomial kernel of degree 10. Images used were 128x128. The classifier was trained with data augmentation, adjusting class instances to match the majority class.

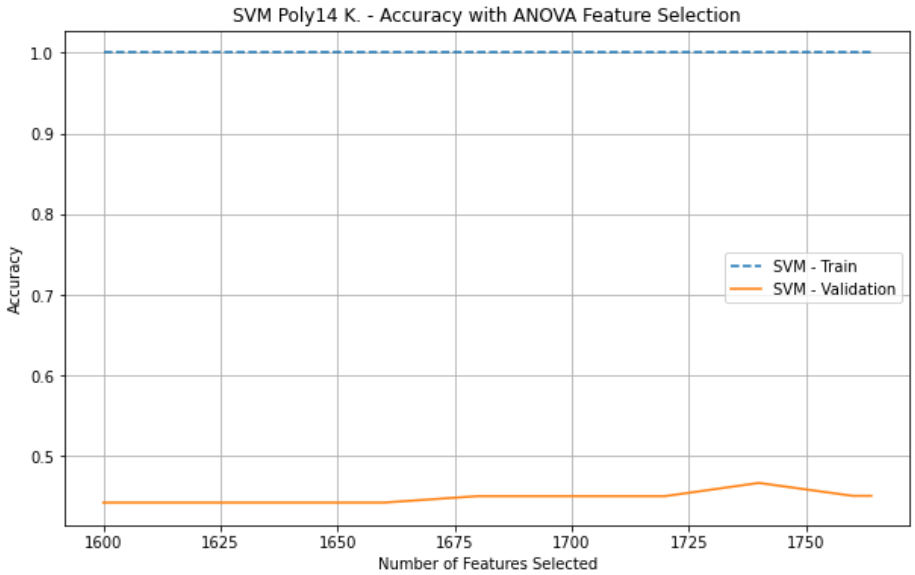


Figure 38: Accuracy with respect to the number of HOG-selected features for SVM with a polynomial kernel of degree 14. Images used were 128x128. The classifier was trained with data augmentation, adjusting class instances to match the majority class.

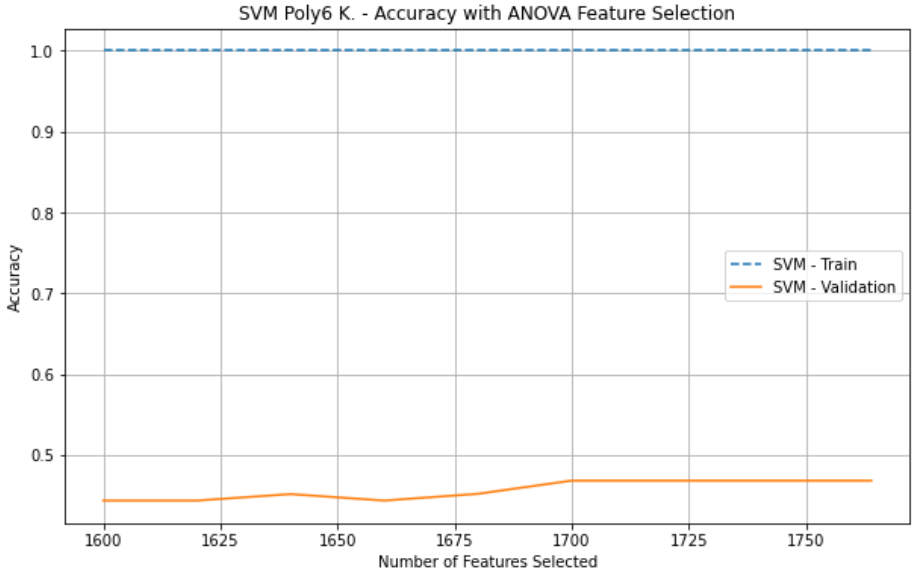


Figure 39: Accuracy with respect to the number of HOG-selected features for SVM with a polynomial kernel of degree 6. Images used were 128x128. The classifier was trained with data augmentation, adjusting class instances to 60.

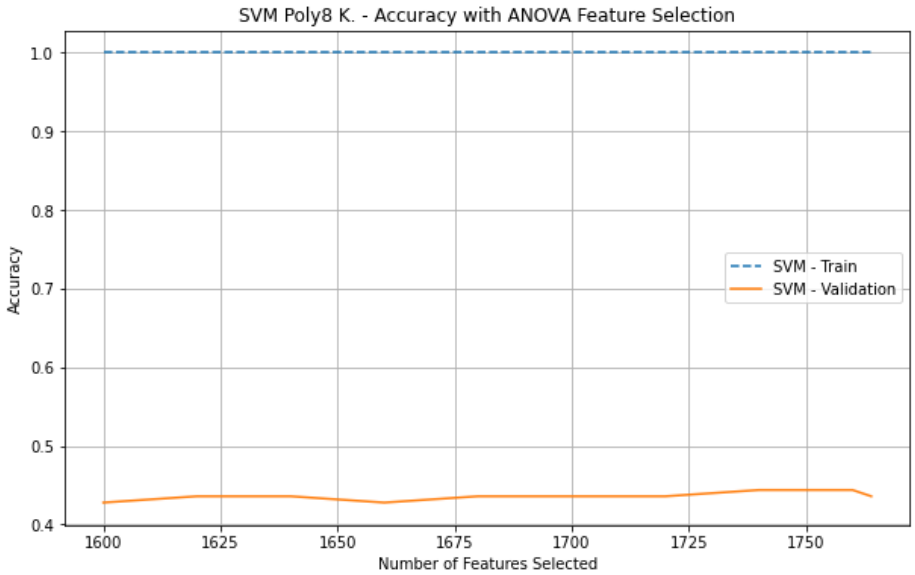


Figure 40: Accuracy with respect to the number of HOG-selected features for SVM with a polynomial kernel of degree 8. Images used were 128x128. The classifier was trained with data augmentation, adjusting class instances to 60.

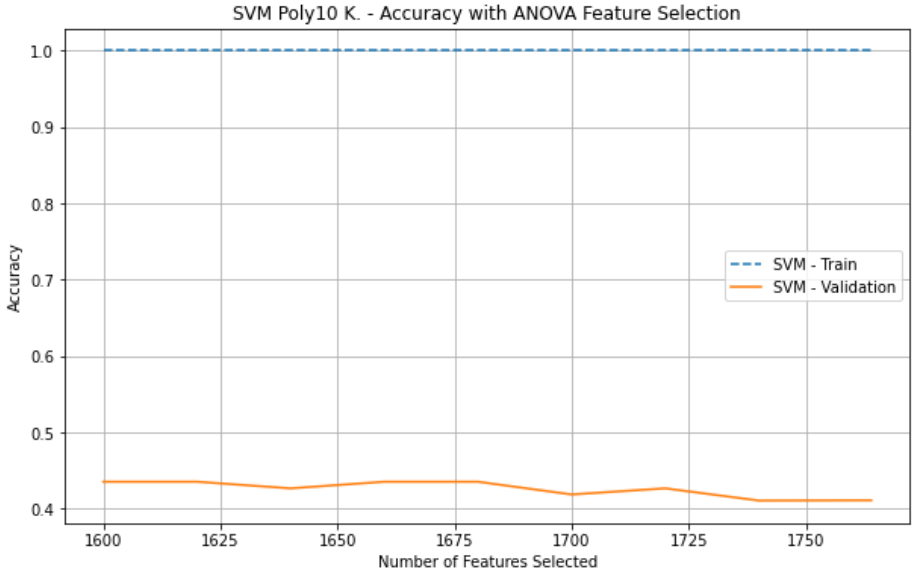


Figure 41: Accuracy with respect to the number of HOG-selected features for SVM with a polynomial kernel of degree 10. Images used were 128x128. The classifier was trained with data augmentation, adjusting class instances to 60.

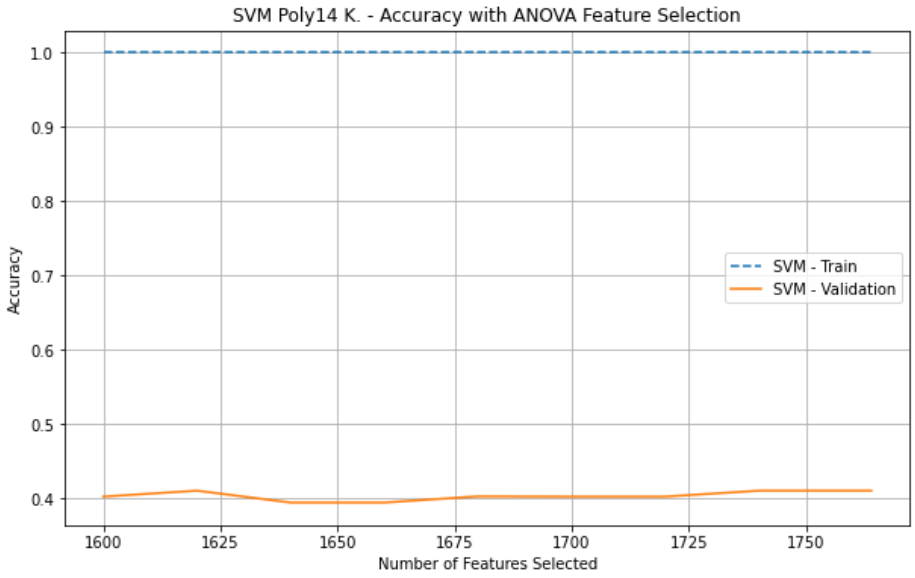


Figure 42: Accuracy with respect to the number of HOG-selected features for SVM with a polynomial kernel of degree 14. Images used were 128x128. The classifier was trained with data augmentation, adjusting class instances to 60.

A.4 Fourth Tuning Phase for Big Cats Classification; Decide on Best SVM Model with HOG Selected Features.

HOG Selected Features	Train Accuracy	Validation Accuracy
1650	1.0	0.46500
1660	1.0	0.46472
1670	1.0	0.46486
1680	1.0	0.46513
1690	1.0	0.46612
1700	1.0	0.46902
1710	1.0	0.46918
No feature selection	1.0	0.46227

Table 6: Cross-validation accuracy averages over 20 different 5-fold cross-validation runs, with respect to HOG ANOVA-selected features for SVM with different polynomial degree kernels. Images used were 128x128. The classifiers were trained with data augmentation, adjusting class instances to match the majority class.

A.5 Fifth Tuning Phase for Big Cats Classification; KNN Vs RF Vs SVM (High-degree Polynomial Kernels) for FFT Cropped Features. Only 128x128 images.

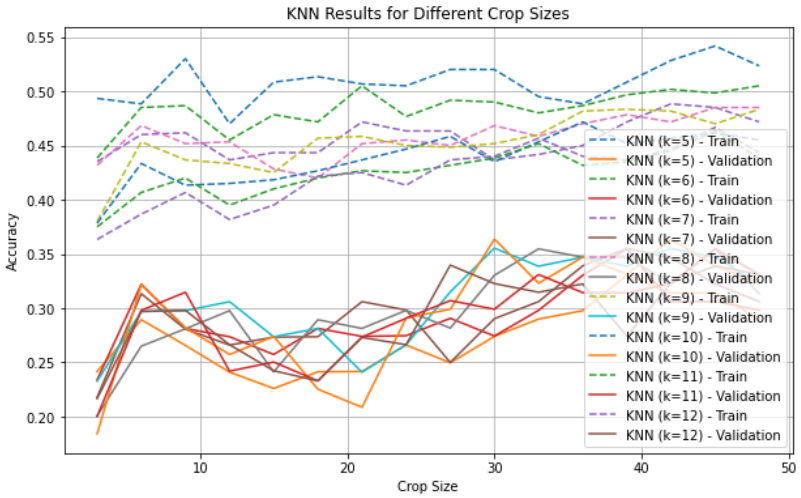


Figure 43: Accuracy with respect to the number of FFT features with smaller crop sizes (lower number of features) for KNN, showing curves for different values of K. Images used were 128x128. The classifier was trained with data augmentation, adjusting class instances to match the majority class.

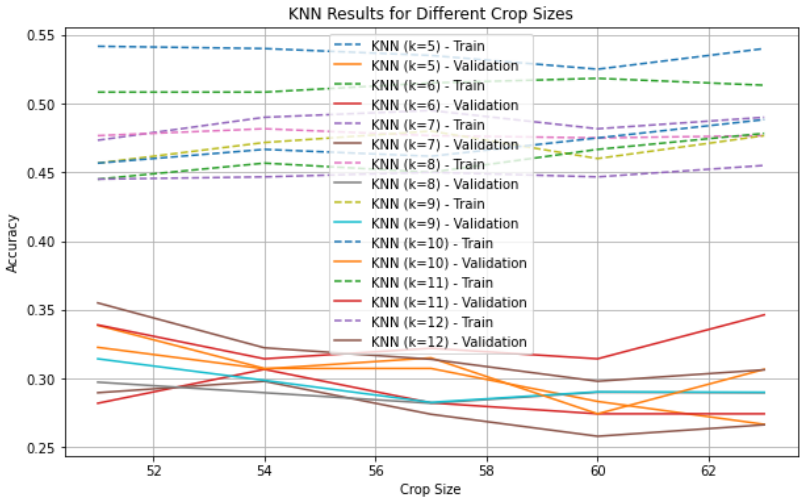


Figure 44: Accuracy with respect to the number of FFT features with larger crop sizes (higher number of features) for KNN, showing curves for different values of K. Images used were 128x128. The classifier was trained with data augmentation, adjusting class instances to match the majority class.

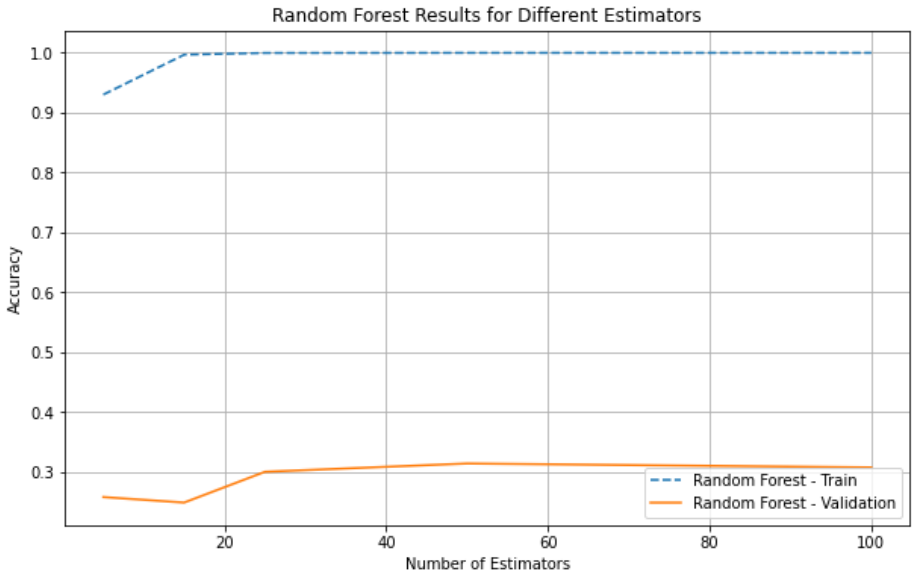


Figure 45: Accuracy with respect to the number of estimators for Random Forest for FFT features. Images used were 128x128. The classifier was trained with data augmentation, adjusting class instances to match the majority class.

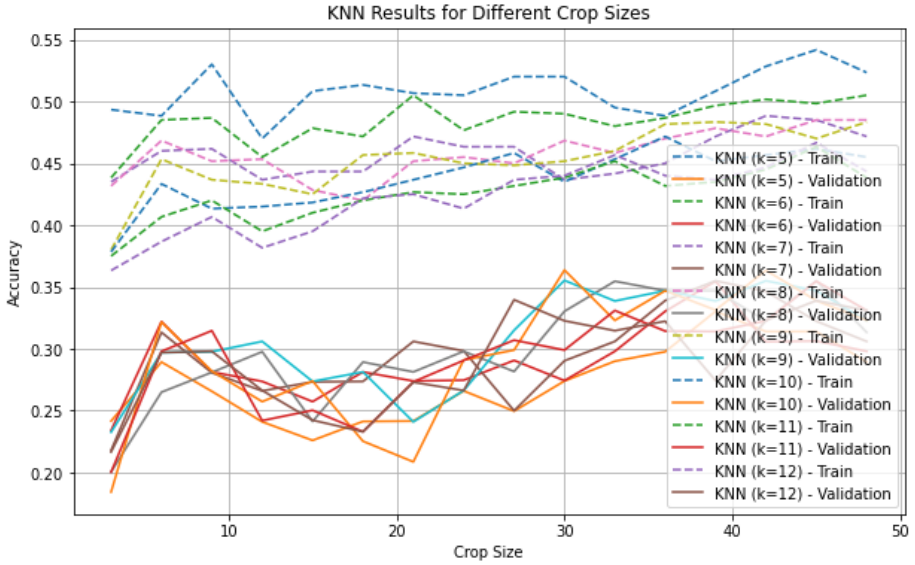


Figure 46: Accuracy with respect to the number of FFT features with smaller crop sizes (lower number of features) for KNN, showing curves for different values of K. Images used were 128x128. The classifier was trained with data augmentation, adjusting class instances to match the majority class.

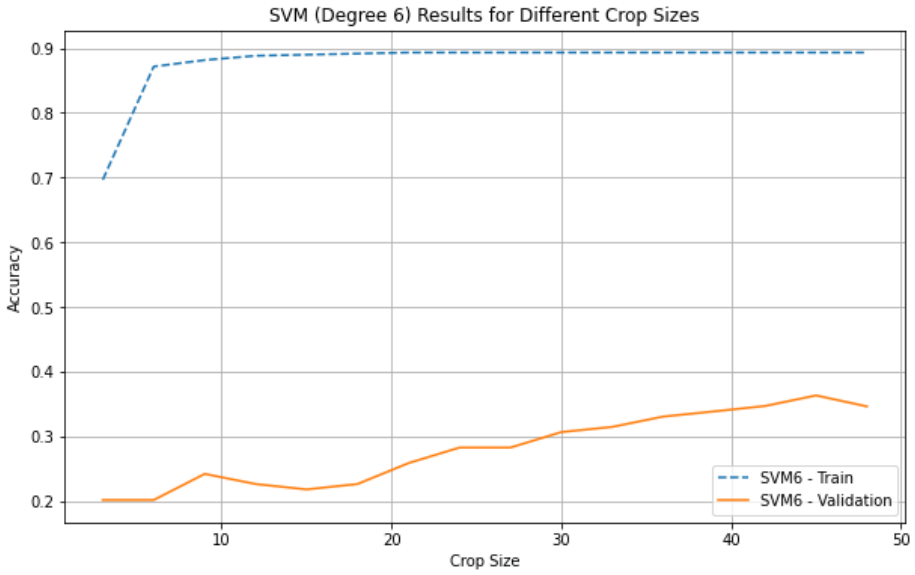


Figure 47: Accuracy with respect to the number of FFT features with smaller crop sizes (lower number of features) for SVM with polynomial kernel of degree 6. Images used were 128x128. The classifier was trained with data augmentation, adjusting class instances to match the majority class.

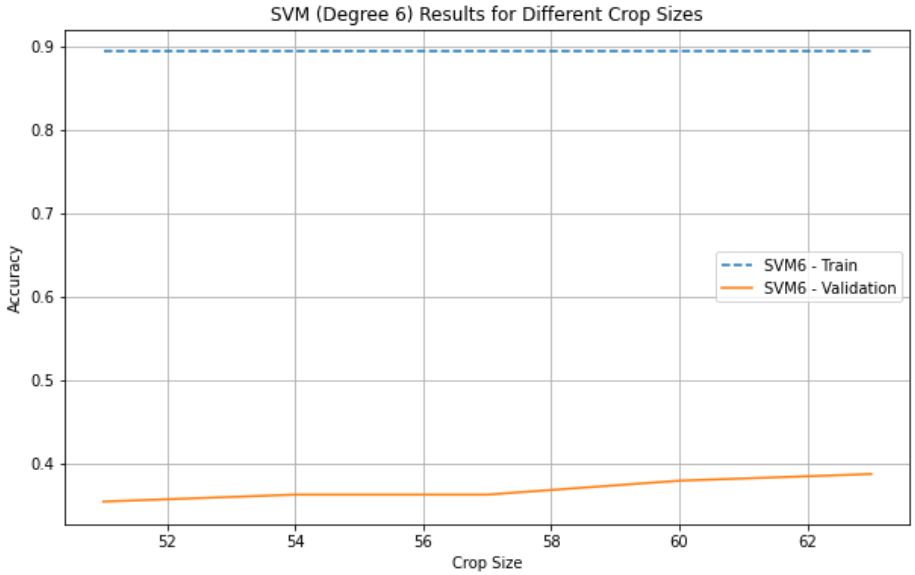


Figure 48: Accuracy with respect to the number of FFT features with larger crop sizes (higher number of features) for SVM with polynomial kernel of degree 6. Images used were 128x128. The classifier was trained with data augmentation, adjusting class instances to match the majority class.

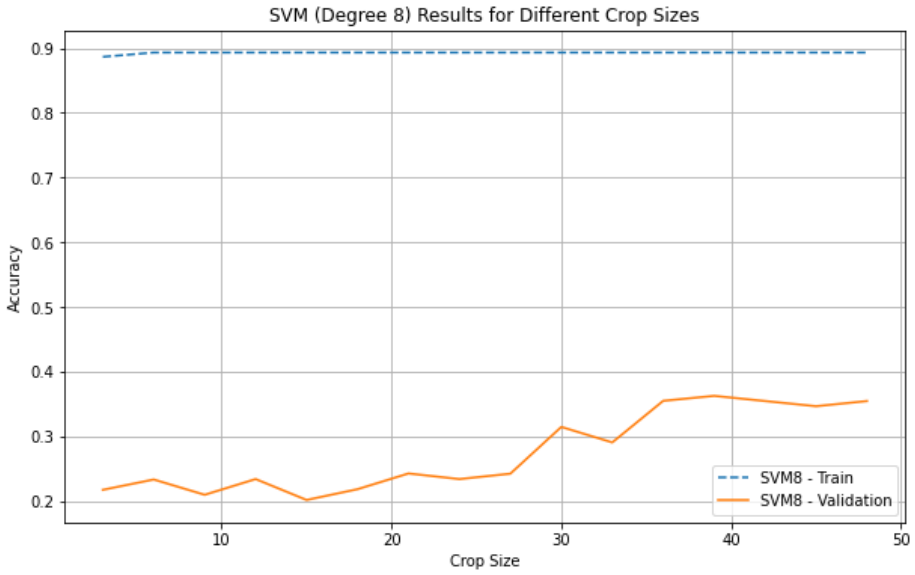


Figure 49: Accuracy with respect to the number of FFT features with smaller crop sizes (lower number of features) for SVM with polynomial kernel of degree 8. Images used were 128x128. The classifier was trained with data augmentation, adjusting class instances to match the majority class.

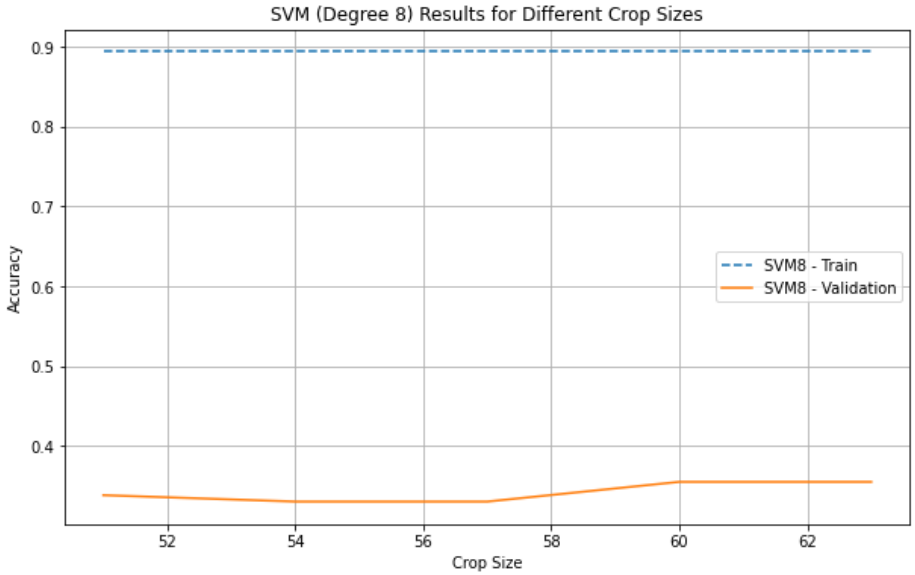


Figure 50: Accuracy with respect to the number of FFT features with larger crop sizes (higher number of features) for SVM with polynomial kernel of degree 8. Images used were 128x128. The classifier was trained with data augmentation, adjusting class instances to match the majority class.

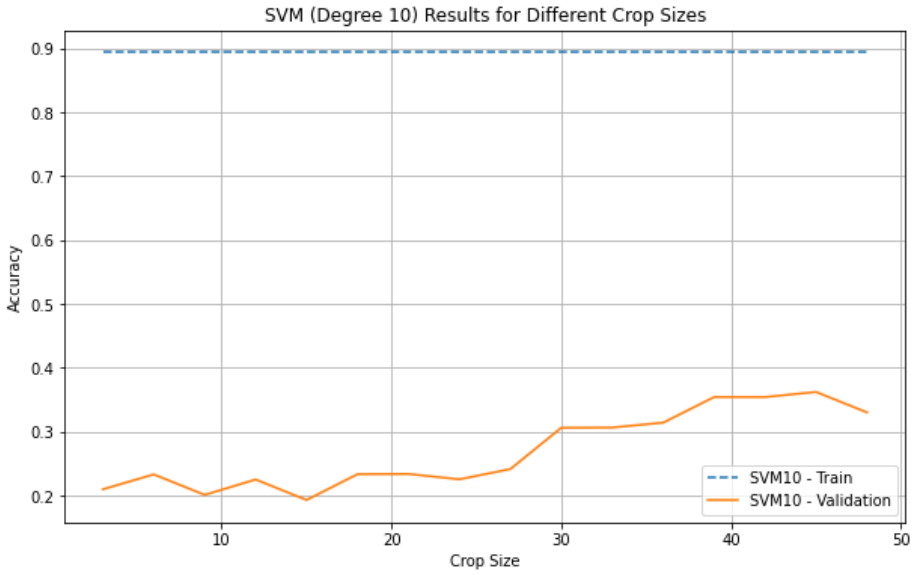


Figure 51: Accuracy with respect to the number of FFT features with smaller crop sizes (lower number of features) for SVM with polynomial kernel of degree 10. Images used were 128x128. The classifier was trained with data augmentation, adjusting class instances to match the majority class.

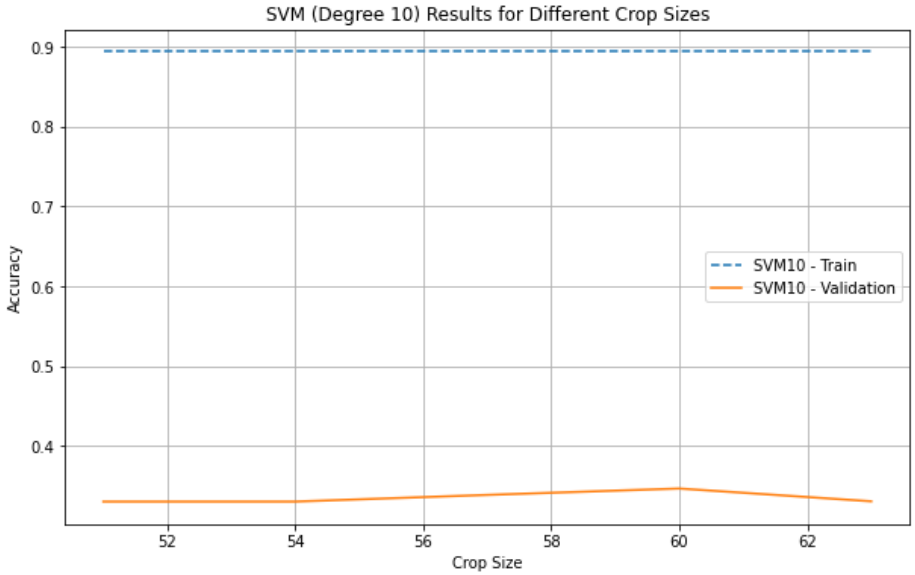


Figure 52: Accuracy with respect to the number of FFT features with larger crop sizes (higher number of features) for SVM with polynomial kernel of degree 10. Images used were 128x128. The classifier was trained with data augmentation, adjusting class instances to match the majority class.

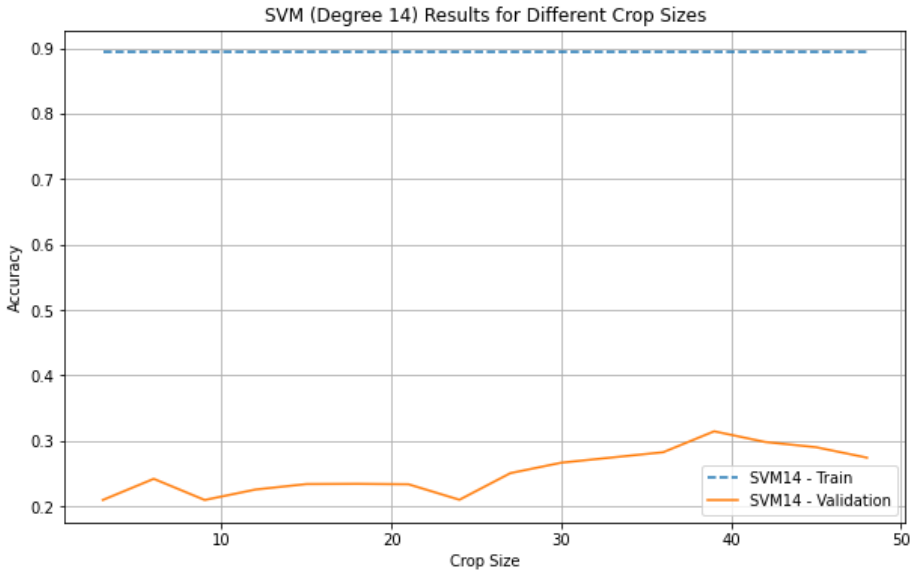


Figure 53: Accuracy with respect to the number of FFT features with smaller crop sizes (lower number of features) for SVM with polynomial kernel of degree 14. Images used were 128x128. The classifier was trained with data augmentation, adjusting class instances to match the majority class.

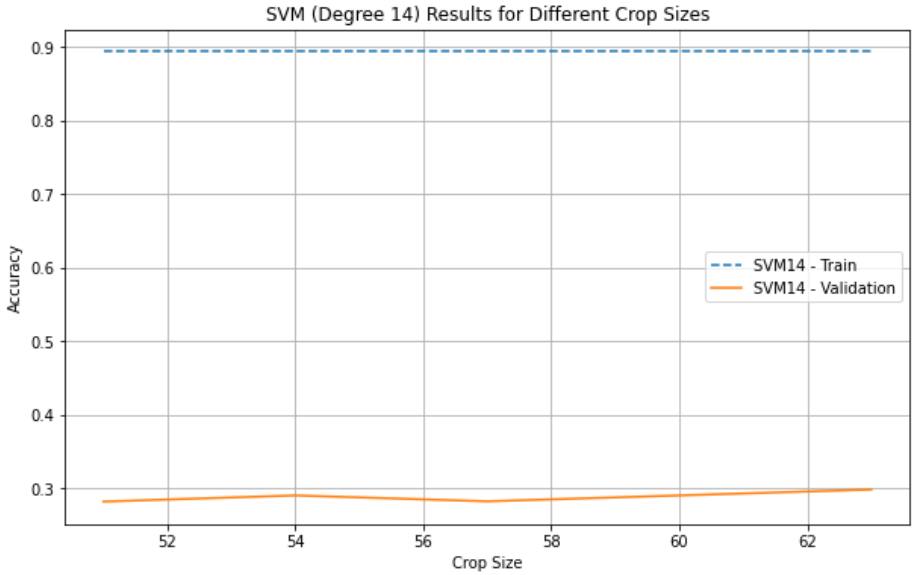


Figure 54: Accuracy with respect to the number of FFT features with larger crop sizes (higher number of features) for SVM with polynomial kernel of degree 14. Images used were 128x128. The classifier was trained with data augmentation, adjusting class instances to match the majority class.

A.6 Tuning Phase for Clustering; Supervised Approach involving Normalized Mutual Information.

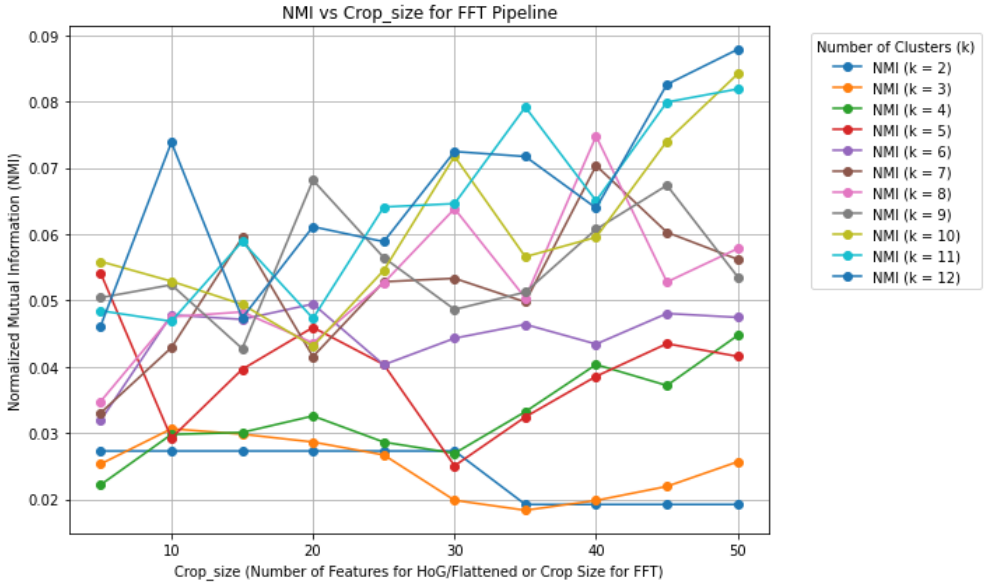


Figure 55: NMI with respect to the crop size for FFT feature selection, for Fuzzy C Means clustering. Different curves show NMI values for different values of k. Images used were 128x128.

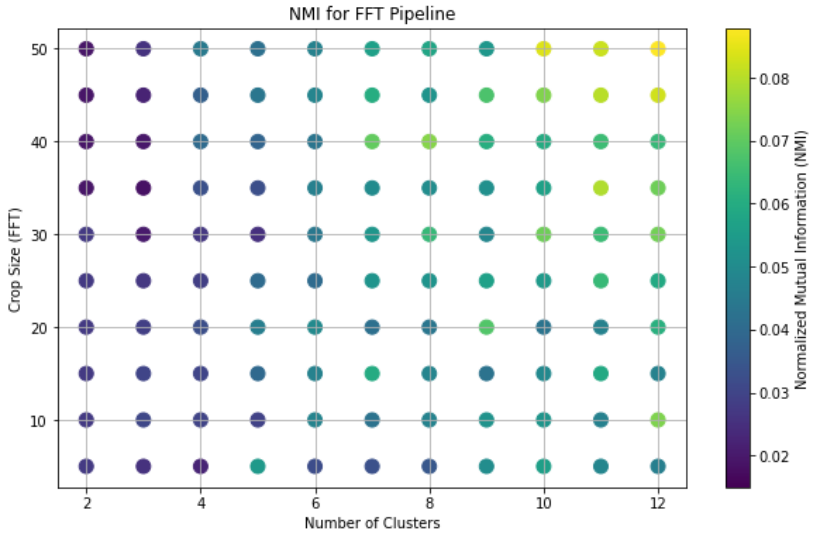


Figure 56: Crop size for FFT feature selection vs. Number of clusters for Fuzzy C Means. NMI values are represented with color based on the color bar at the right. Images used were 128x128. Corresponds to 55

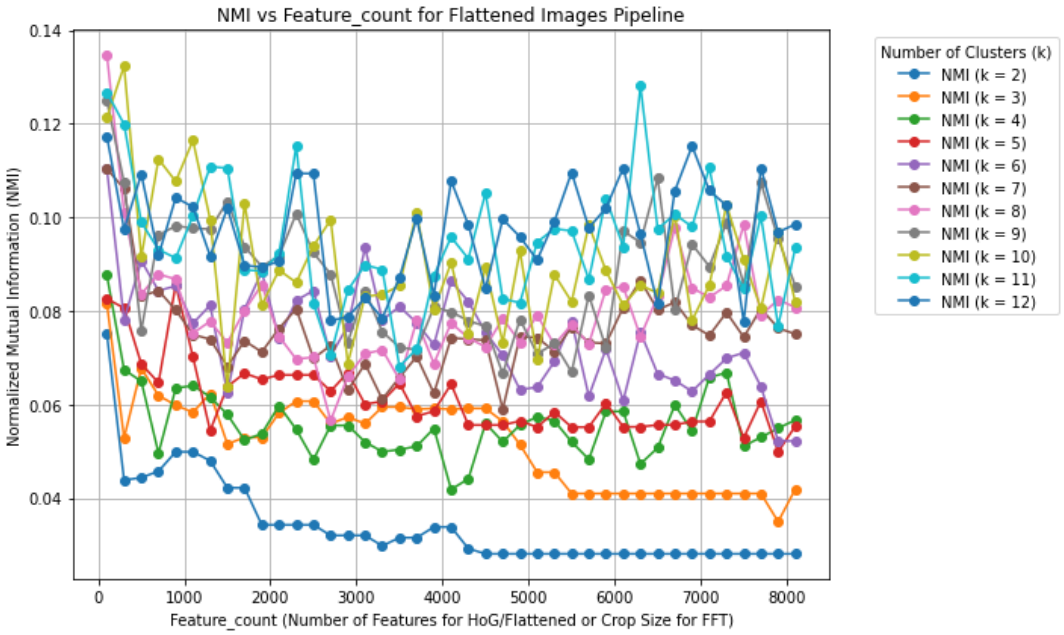


Figure 57: NMI with respect to image selected features, for Fuzzy C Means clustering. Different curves show NMI values for different values of k. Images used were 128x128.

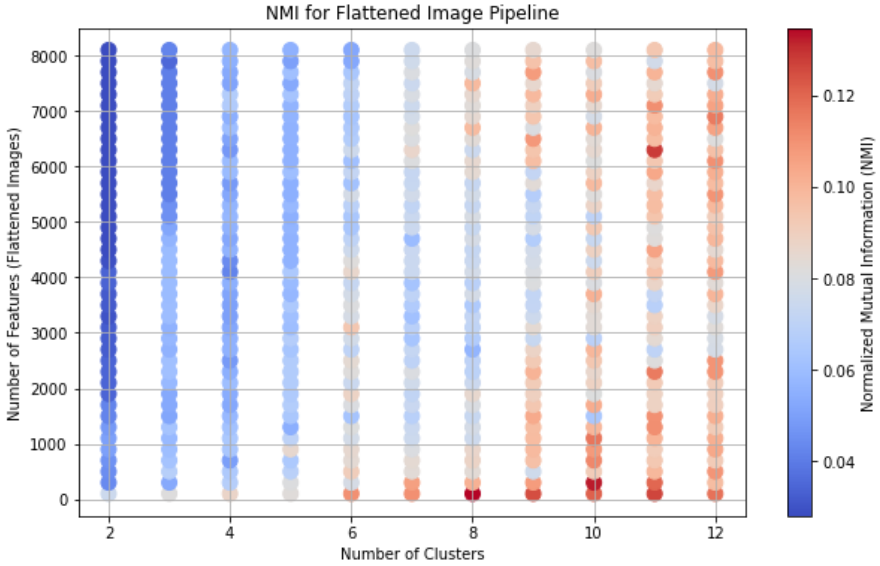


Figure 58: Image selected features vs. Number of clusters for Fuzzy C Means. NMI values are represented with color based on the color bar at the right. Images used were 128x128. Corresponds to 57

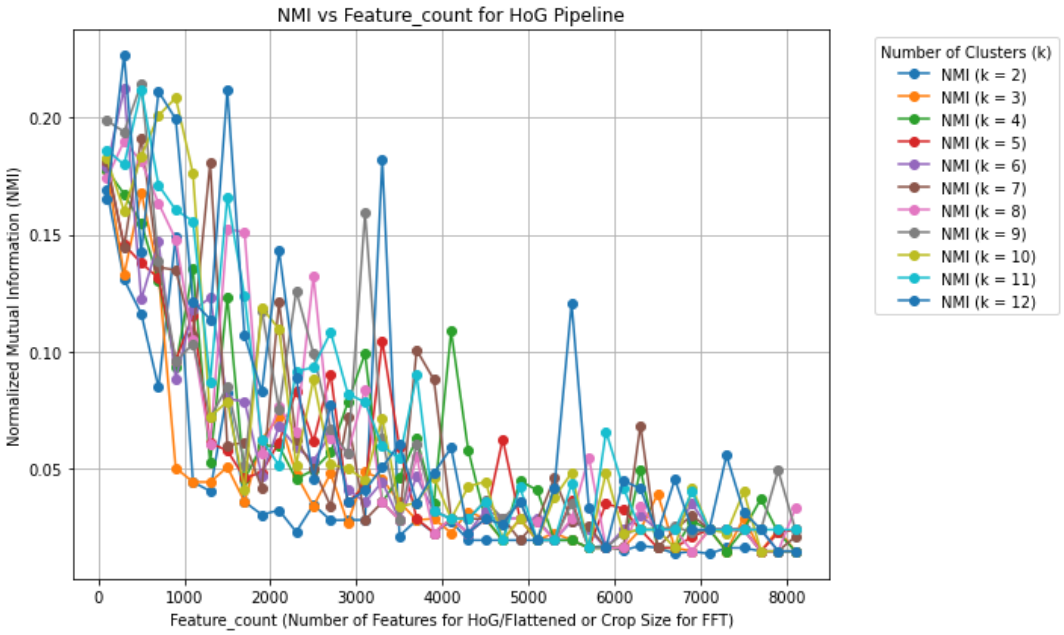


Figure 59: NMI with respect to HOG selected features, for Fuzzy C Means clustering. Different curves show NMI values for different values of k. Images used were 128x128.

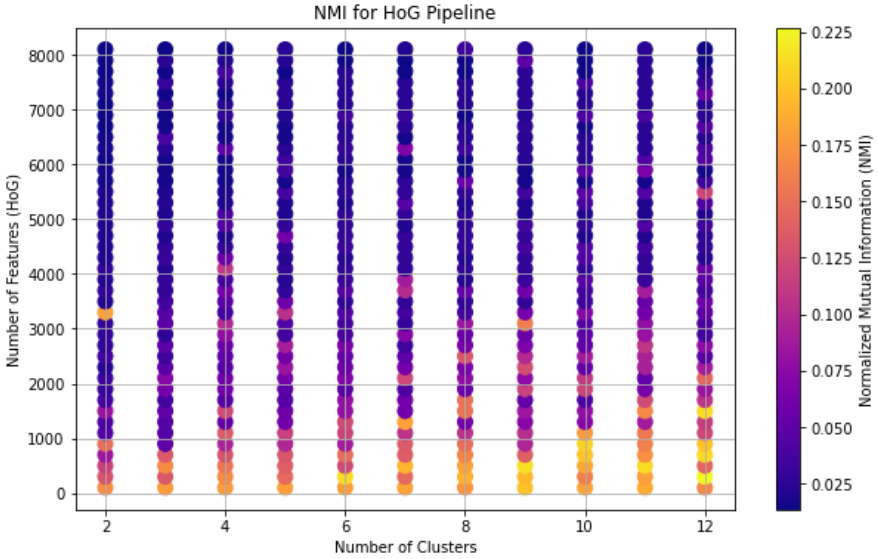


Figure 60: HOG selected features vs. Number of clusters for Fuzzy C Means. NMI values are represented with color based on the color bar at the right. Images used were 128x128. Corresponds to 59

A.7 Tuning Phase for Clustering; Unsupervised Approach involving Fuzzy Partition Coefficient.

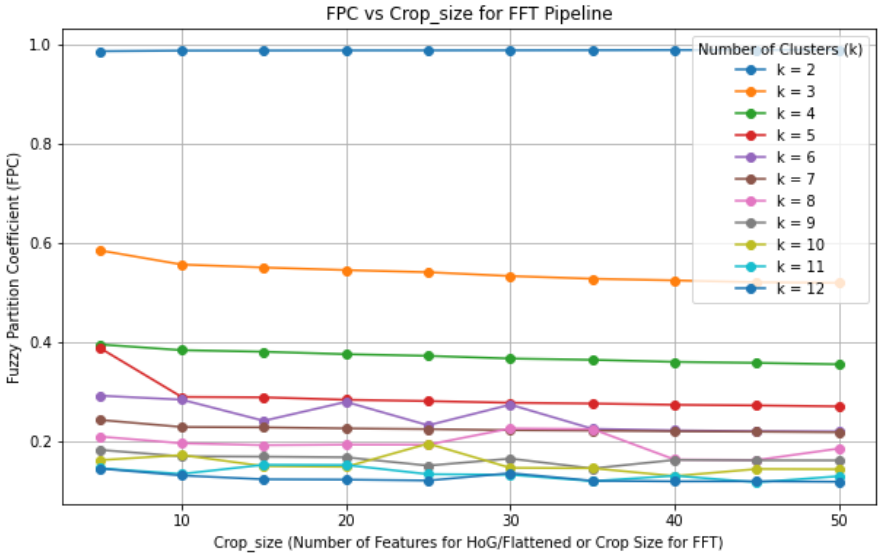


Figure 61: FPC with respect to the crop size for FFT feature selection, for Fuzzy C Means clustering. Different curves show FPC values for different values of k. Images used were 128x128.

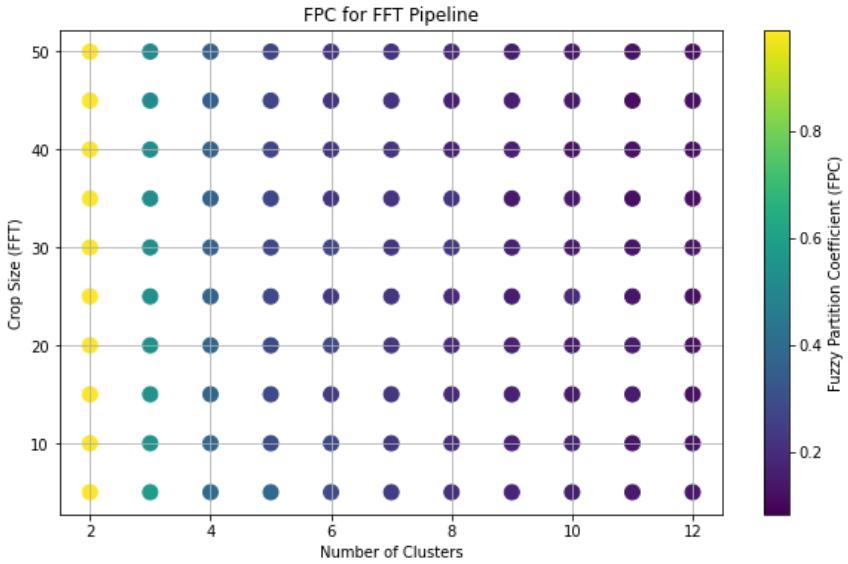


Figure 62: Crop size for FFT feature selection vs. Number of clusters for Fuzzy C Means. FPC values are represented with color based on the color bar at the right. Images used were 128x128. Corresponds to 61

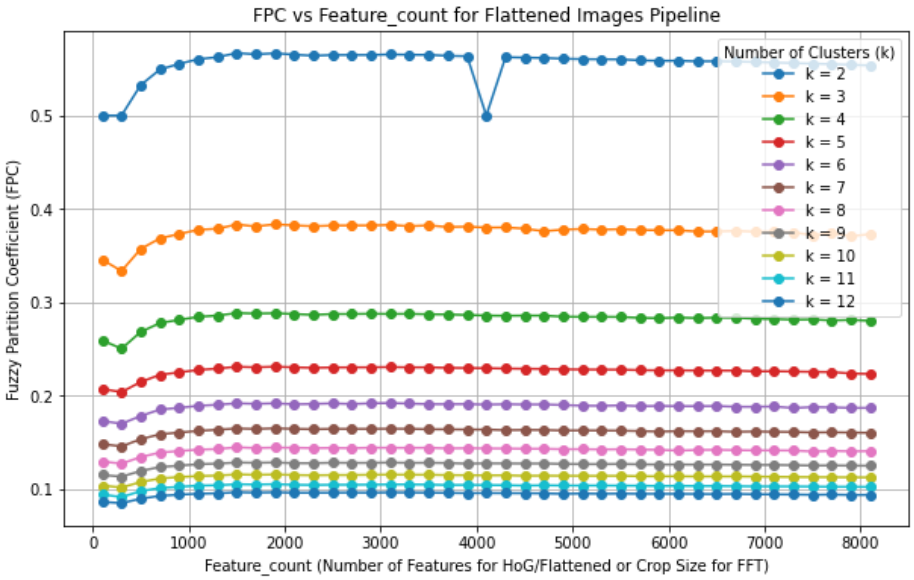


Figure 63: FPC with respect to image selected features, for Fuzzy C Means clustering. Different curves show FPC values for different values of k. Images used were 128x128.

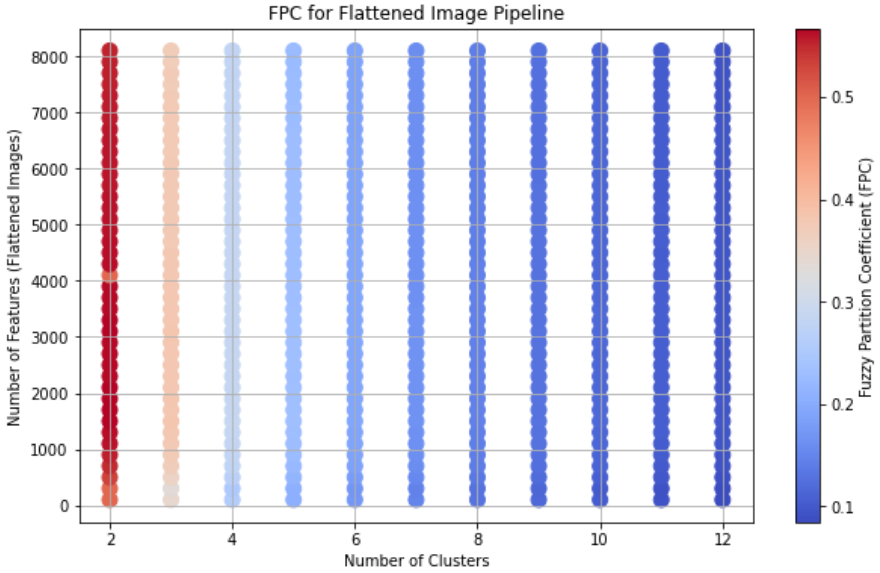


Figure 64: Image selected features vs. Number of clusters for Fuzzy C Means. FPC values are represented with color based on the color bar at the right. Images used were 128x128. Corresponds to 63

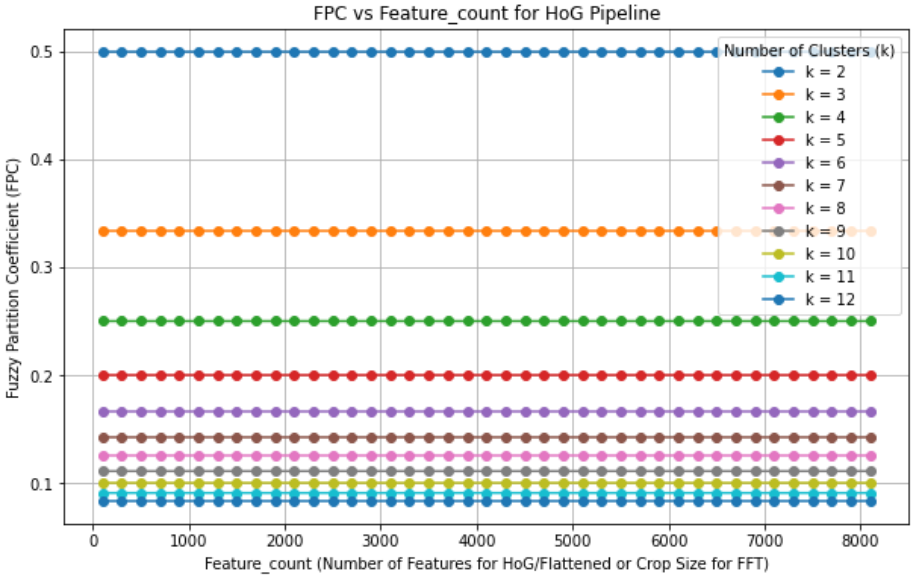


Figure 65: FPC with respect to HOG selected features, for Fuzzy C Means clustering. Different curves show FPC values for different values of k. Images used were 128x128.

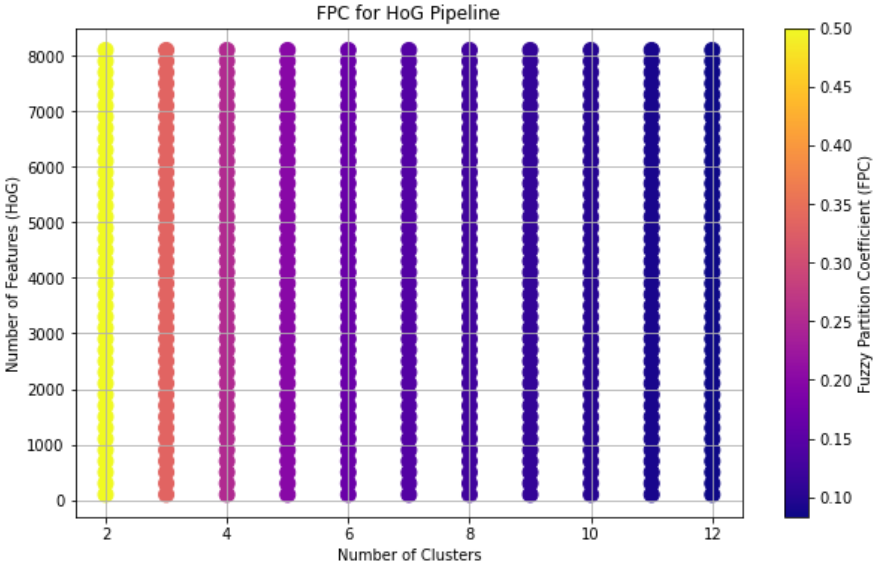


Figure 66: HOG selected features vs. Number of clusters for Fuzzy C Means. FPC values are represented with color based on the color bar at the right. Images used were 128x128. Corresponds to 65

k	NMI
2	0.0097
3	0.0234
4	0.0303
5	0.0457
6	0.0499
7	0.0528
8	0.0571
9	0.0530
10	0.0637
11	0.0556
12	0.0715

Table 7: Fuzzy c-means for flattened images without feature selection.