# Randomized Decision Forests for Image Categorisation

Andre Bharath
CID: 01054089
agb215@imperial.ac.uk

Rajan Patel
CID: 01062228
rnp15@imperial.ac.uk

## Abstract

*We present an exploration into image categorisation, by comparing the effectiveness of both Randomized Decision Forests and $k$-Means clustering in constructing a vocabulary of visual codewords (codebook). The subsequent Bag-of-(Visual)-Words (BoVW) representations of images are used as feature vectors, where the implementation of Random Forests for classification purposes are also considered. We endeavour to find an optimal approach that maximizes the classification accuracy, whilst taking into consideration computational complexity.*[1]

## 1. Problem Overview

With the aim to efficiently search, recommend, react to or reason with new image instances, it is essential to have methods that deal with high-level information pertaining to the objects contained in images belonging to large-scale digital image collections. This problem area is referred to as generic visual object categorization [5], where the processes involved in solving such problems should be generic in order to deal with a vast number of object categories. Furthermore, such processes should be able to cope with large intra-class variations, affine transformations (translation, scaling, reflection, rotation), as well as variations in view, imaging, occlusion and lighting, typical of semantic classes of everyday objects. We explore a BoVW approach, where the Scale-Invariant Feature Transform (SIFT) [8] is applied, and extracted descriptors are used to construct a codebook, via various configurations of $k$-Means clustering or Random Forest embeddings. Subsequent BoVW image representations are used as features for Random Forests to carry out classification.

### 1.1. Dataset, Definitions & Preprocessing

All experiments in this report have been conducted on a subset of the popular Caltech 101 dataset [4], which contains $N = 300$ images representing $L = 10$ unique object categories. Each unique category/class $C_i$ where i $\in$ [1, $L$], contains 30 images. The dataset is split into a training set $X_{train}$ and a testing test $X_{test}$, where 15 images from each class $C_i$ are randomly assigned to $X_{train}$ and $X_{test}$, respectively. Thus, the cardinalities of $X_{train}$ and $X_{test}$ are denoted as $N_{train} = 150$ and $N_{test} = 150$ respectively.

## 2. $k$-Means Codebook

We first consider the construction of the visual codebook using $K$-Means clustering, as summarised in Algorithm 1, where $\mathcal{I}_i$ denotes a particular image and $\mathcal{D}$ is initialized as an empty set, $D \in \emptyset$. SIFT is applied to gray-scale versions of the images in the dataset, in order to extract descriptors[2], as can be seen from Figure 8. SIFT descriptors are computed on an image neighborhood and are Gaussian derivatives computed at 8 orientation planes over a 4x4 grid of spatial locations, resulting in 128-dimensional multi-scaled dense descriptor vectors [8].

[1]Working code for all experiments is documented at: https://github.com/RajanPatel97/Computer-Vision-CW1

[2]$N_d = 100$k descriptors uniformly sampled without replacement. Implemented using the OpenCV library in Python

### 2.1. Vector Quantisation Process

The vector quantisation process (codeword assignment) is carried out using MiniBatch $K$-Means clustering [11] described in Figure 10 with nearest neighbors instead of using standard $K$-Means, since the standard version is time-consuming and scales poorly with large datasets, due to the possible large number of classes [9]. Figure 9 verifies that the computational performance of MiniBatch $k$-Means outperforms that of standard $k$-Means on randomly generated sample data. The standard $K$-means algorithm converges only to local optima of the squared distortion measure [12] and is greatly dependent on the initialisation of its cluster centroids. Thus, $k$-Means++ [2] is used for the initialization of cluster centroids, as it reduces the computational load[3] for a given value of $K$, as it has a time complexity of $O(log(K))$.

The advantage of using the BoVW approach is that the initial information loss introduced by clustering during the construction of the visual codebook is eliminated. Once the cluster centroids (codewords) are calculated, vector quantization is carried out to construct the BoVW histogram representations. Every descriptor belonging to a particular image is assigned to the nearest centroid obtained via $k$-Means clustering. The frequency of occurrences of each centroid are counted, which eventually produces the respective BoVW histogram representation. Thus, the BoVW histograms are $K$-dimensional, and provide a highly compact and robust representation of images, which greatly facilitates modelling images and categorising them [13]. Even though the sizes of the original images vary[4], the constituent codewords of the BoVW representations are consistent across all images, enabling a single RF classifier to be applied.

### 2.2. Vocabulary Size

For a small vocabulary size $K$, codewords would not be representative of all patches and generalization will be poor as there will not be enough discrimination between the various classes (underfitting). Alternatively, if $K$ is too large, generalization may improve at the cost of computational performance and quantization artifacts (overfitting). There are existing methods based on the Bayesian Information Criterion [10], enabling the automatic estimation of the choice of $K$. In the experiments conducted for this report, $K$ is determined empirically through running $K$-means with a different number of desired clusters ($K$) at different initialisations. The final clustering that gives the lowest empirical error for the categorization task is selected. In Figure 1, BoVW

[3]PCA on the descriptor vectors could also be considered for high descriptor dimensions in order to further reduce computational load.

[4]The average size of each image is roughly 300 x 200 pixels

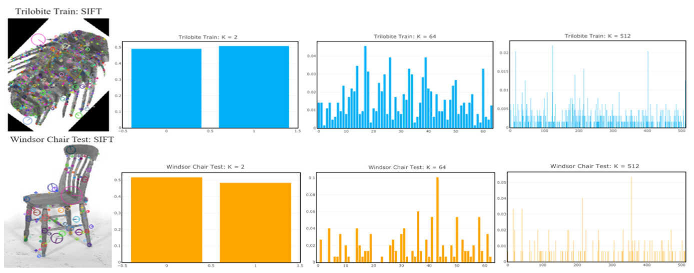


Figure 1. BoVW Histograms for $K = 2$, $K = 64$ and $K = 512$ of $X_{train}$ (top) and $X_{test}$ images (bottom).

representations of a train and test example are provided, demonstrating the difference between the feature vectors for different vocabulary sizes.

## 3. Random Forests Classifier

There are a variety of hyperparameters that strongly influence the RF classification accuracy such as: the maximum tree depth, number of trees, degree of randomness, type of weak learner and the vocabulary size. Grid Search Leave-One-Out Cross-Validation (GSLOOCV) [7] is conducted in order to discover the optimal configuration of hyperparameters that achieve the best classification accuracy.

### 3.1. Maximum Tree Depth

It is important to consider the stopping criteria for training and testing when growing the tree, as the ability of the model to generalize can be seriously degraded due to overfitting, as Decision Tree (DT) classifiers have the capacity to store every training sample in a single node. The maximum tree depth denoted as $d_{\max}$, prevents the indefinite growth of the decision tree to an upper bound of $2^{d_{\max}-1}$ nodes. In order to prevent any particular node from having too few samples, the number of samples is thresholded to a minimum value. Furthermore, excessive node splitting is limited by a minimum threshold for information gain. From Figure 2, it can be observed that classification accuracy is restricted at the extremes; for smaller values, the RF underfits the data due to an inability to split the data effectively with such a shallow depth. For larger values, the RF begins to overfit, as the leaf nodes contain fewer points. Also, deeper trees are computationally more expensive to train, as the execution time increases exponentially for training and linearly for testing with increasing $d_{\max}$. Hence, $d_{\max} = 14$ was chosen to maximize classification accuracy, but also to achieve good generalization in a reasonable time.

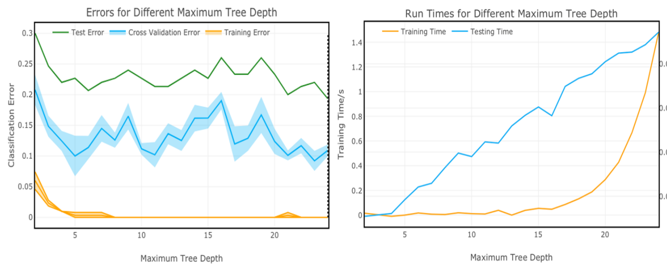


Figure 2. Classification accuracy (left) and time complexity (right) for various tree depths

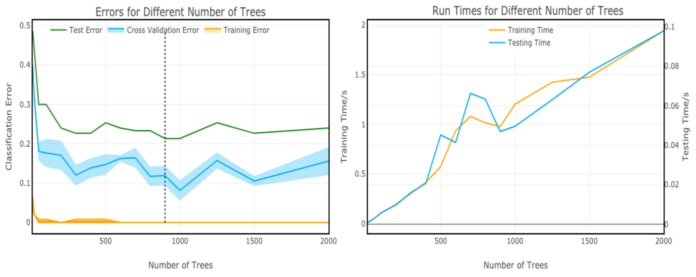


Figure 3. Classification accuracy (left) and time complexity (right) for varying $N_{trees}$

### 3.2. Number of Trees

Figure 3 illustrates that the classification accuracy increases with an increasing number of decorrelated trees, as the individual tendencies of trees to overfit the data are averaged out by the RF, leading to improved generalization and robustness. This overfitting tendency is particularly dominant for lower numbers of trees as no averaging effect is observed, explaining the initial steepness of the curve and the subsequent plateau. It can also be observed that time complexity increases in a linear fashion as the number of trees, $N_{trees}$, increase, however execution time can be reduced further as training and testing can be conducted in parallel. The final value for $N_{trees}$ selected was 950, due to satisfactory execution times of 1s and 0.05s respectively for training and testing, resulting in a slight decrease in accuracy from the highest accuracy value observed. Bagging [3] with replacement is performed to select the training data for each tree.

### 3.3. Degree of Randomness

The correlation between the constituent DTs of the RF can be reduced by utilizing random splits, where the degree of randomness introduced is controlled by the number of random splits. Figure 4 illustrates that the classification accuracy improves as the number of splits increases until it plateaus at 7 splits, whilst the training complexity continues to increase in a linear fashion, as expected. Predictably, a small number of splits leads to underfitting due to greater correlation between trees, whilst a large number of random splits increases complexity without a noticeable improvement to classification accuracy. Therefore, 7 splits are chosen, in order to maximize classification accuracy with an adequate time.

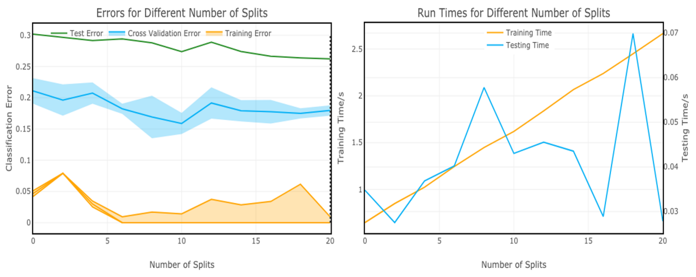


Figure 4. Classification accuracy (left) and computation times (right) for varying number of random splits.

### 3.4. Weak Learners

The following weak learners are considered for node splitting experiments; axis-aligned, two-pixel test, linear, quadratic and cubic kernels. Figure 5 illustrates that computation time in-

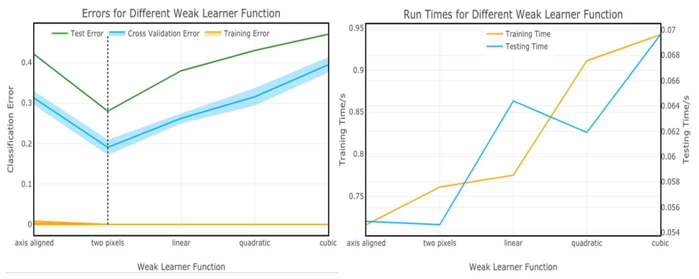


Figure 5. Classification accuracy (left) and time complexity (right) for various weak learners

creases as the weak learner becomes more sophisticated with respect to the higher degrees of freedom associated with the intrinsic flexibility of higher-order polynomial decision functions to learn non-linear decision boundaries [6, 14]. However, this excessive discriminative power can lead to model overparameterization and thus overfit the training data. As a result, the two-pixel test weak learner was chosen, since it achieves the best accuracy with a reasonable time complexity, as it provides an approximation of the gradient, whilst only using one more dimension than the axis-aligned weak learner.

### 3.5. Vocabulary Size

From Figure 6, it can be observed that there is initially a steep decrease in train/test error for increasing vocabulary size up to a size of 30. This corroborates the reasoning given in subsection 2.2, such that for small vocabulary sizes, the model underfits the data due to lack of discrimination between the various classes. The cross-validation and test errors steadily increase as the vocabulary size increases past $K = 250$. It is inferred that this originates from the RF overfitting the data due to the larger data dimensionality. The time taken for training increases linearly with larger vocabulary sizes, as can be expected.

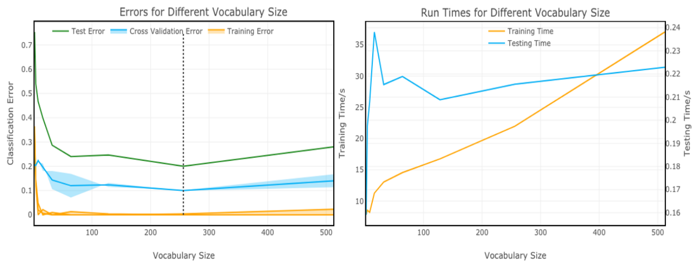


Figure 6. Classification accuracy (left) and time complexity (right) for varying vocabulary size

### 3.6. Optimal Configuration

The optimal hyperparameters chosen for the model (Table 1) achieves a 80.2% accuracy on the test set, and its corresponding classification success/failure examples and confusion matrix are provided in Figures 18 and 16, respectively. As can be observed from both figures, watches (class 7) are poorly classified as a trilobite, wrench or yin-yang, perhaps because of the round outline of the trilobite body, round hole on the wrench and yin-yang circle on the hat having similar descriptors to a circular watchface.

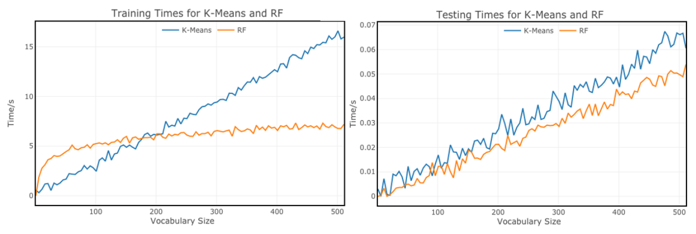


Figure 7. $k$-Means Vs RF Codebook Comparison of Vector Quantisation Process Complexity. Further results can be seen in Appendix III.

## 4. Random Forests Codebook

### 4.1. Vector Quantisation Process

In contrast to the $k$-means approach, the RF codebook is constructed in a supervised manner and exploits the valuable discriminative information in the descriptor space, which is mapped to leaf embeddings in an encoding process. During RF codebook construction, labels are taken into account and extracted descriptors are assigned to a "cluster", determined by the leaf nodes they end up in. For an RF with $T$ DT's, each descriptor reaches $T$ leaf nodes. After transforming the SIFT descriptors of a particular image, the corresponding BoVW representation is obtained by collecting all leaf embeddings of the image, across the ensemble of trees. The vocabulary size $K$ is equal to the exact number of leaves in the RF codebook. However, only an upper bound on $K$ can be controlled: $K \leq N_{trees} \times 2^{d_{\max}-1}$.

### 4.2. Optimal Configuration

Due to the relatively small size of the dataset, GSLOOCV is carried out concurrently on both the RF codebook (feature transformer) and RF classifer to test a vast range of combinations of hyperparameters belonging to both. The optimal hyperparameters chosen for the model (Table 1), achieves a 70.3% accuracy on the test set, where its corresponding classification success/failure examples and confusion matrices are provided in Figures 19 and 17, respectively. As can be observed from the figures, yin-yang images (class 6) are poorly classified as a Windsor chair, an umbrella and a wrench. The yin-yang is more often mis-classified as an umbrella (class 5), perhaps due to the upper circular shape, or in this case, both having similar descriptors for adjacent dual colors. Further results can be seen in Appendix II.

## 5. Comparisons & Conclusions

From Table 1, it can be seen that the $k$-Means codebook with RF classifier achieves a better recognition accuracy, 80.2% compared to the RF codebook with 70.3%. Despite achieving lower accuracy, the RF codebook verifies that it is capable of encoding and compressing rich discriminative information, since its corresponding RF classifier is simplier, faster and $\sim$ 3 times smaller than its $K$-Means counterpart. Specifically, the RF codebook uses an optimal 300 DTs with maximum depth 14, trained in 6.5s, while the $k$-means codebook uses 950 DTs and maximum depth 10, trained in 24.0s. Furthermore, Figure 7 illustrates and verifies that the RF codebook scales much better with vocabulary size, $K$ than the $k$-Means codebook. It can be observed that the RF codebook has a logarithmic training complexity of, $O(log(K))$, and the $K$-Means codebook has a linear training complexity of $O(K)$.

## References

[1] S. Agrawal. Machine learning – mini batch k-means, 2019.

[2] D. Arthur and S. Vassilvitskii. K-means++: The advantages of careful seeding. In *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '07, pages 1027–1035, Philadelphia, PA, USA, 2007. Society for Industrial and Applied Mathematics.

[3] S. Dudoit and J. Fridlyand. Bagging to improve the accuracy of a clustering procedure. *Bioinformatics*, 19(9):1090–1099, 2003.

[4] L. Fei-Fei, R. Fergus, and P. Perona. One-shot learning of object categories. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(4):594–611, April 2006.

[5] H. Fu. *Contributions to generic visual object categorization*. Theses, Ecole Centrale de Lyon, Dec. 2010.

[6] W. Jiang. On weak base hypotheses and their implications for boosting regression and classification. *Ann. Statist.*, 30(1):51–73, 02 2002.

[7] R. Kohavi. A study of cross-validation and bootstrap for accuracy estimation and model selection. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence - Volume 2*, IJCAI'95, pages 1137–1143, San Francisco, CA, USA, 1995. Morgan Kaufmann Publishers Inc.

[8] D. G. Lowe. Object recognition from local scale-invariant features. In *Proceedings of the Seventh IEEE International Conference on Computer Vision*, volume 2, pages 1150–1157 vol.2, Sep. 1999.

[9] D. Nistér and H. Stewénius. Scalable recognition with a vocabulary tree. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 2, pages 2161 – 2168, 02 2006.

[10] D. Pelleg and A. Moore. X-means: Extending k-means with efficient estimation of the number of clusters. In *In Proceedings of the 17th International Conf. on Machine Learning*, pages 727–734. Morgan Kaufmann, 2000.

[11] D. Sculley. Web-scale k-means clustering. In *Proceedings of the 19th international conference on World wide web - WWW '10*, pages 1177–1178, 2010. Exported from https://app.dimensions.ai on 2019/02/11.

[12] A. K. Singh, I. A. Yadav, and A. Rana. K-means with three different distance metrics. 2013.

[13] X. Solé, A. Ramisa, and C. Torras. Evaluation of random forests on large-scale classification problems using a bag-of-visual-words representation. 269:273–276, 01 2014.

[14] A. J. Wyner, M. Olson, J. Bleich, and D. Mease. Explaining the success of adaboost and random forests as interpolating classifiers. *J. Mach. Learn. Res.*, 18(1):1558–1590, Jan. 2017.

## Appendix I - $k$-Means Codebook

**Algorithm 1** $K$-Means Codebook Construction

```
For $\mathcal{I}_i \in X_{train}$                          $\triangleright i \in [1, N_{train}]$
    Extract SIFT features from $\mathcal{I}_i$ and append to $\mathcal{D}$
end
Randomly select $N_d$ descriptors from $\mathcal{D}$        $\triangleright N_d = 100,000$
Carry out $K$-Means on $N_d$ chosen descriptors
For $\mathcal{I}_i \in X_{train} \cup X_{test}$                   $\triangleright i \in [1, N]$
    Using trained $K$-means transform $\mathcal{I}_i$ from pixels to corresponding BoVW representation.
end
```

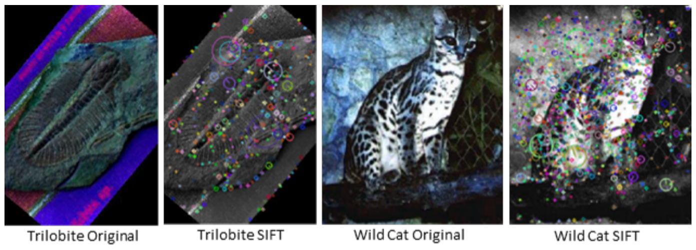


Figure 8. Original Images with SIFT applied to Greyscale Images. Trilobite images are from $X_{train}$ and Wild Cat images are from $X_{test}$

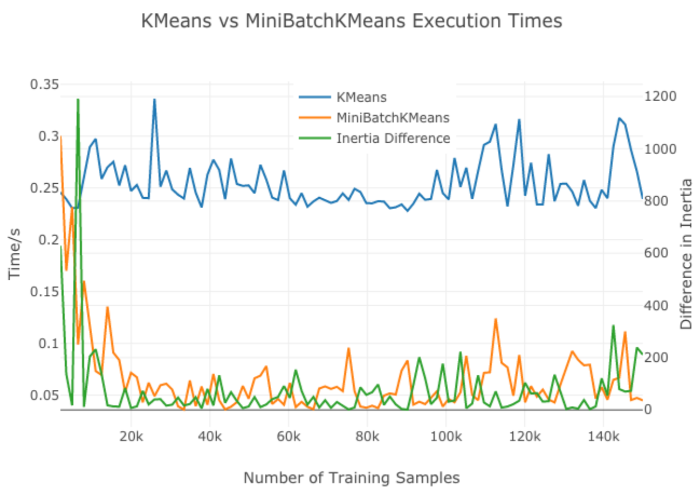
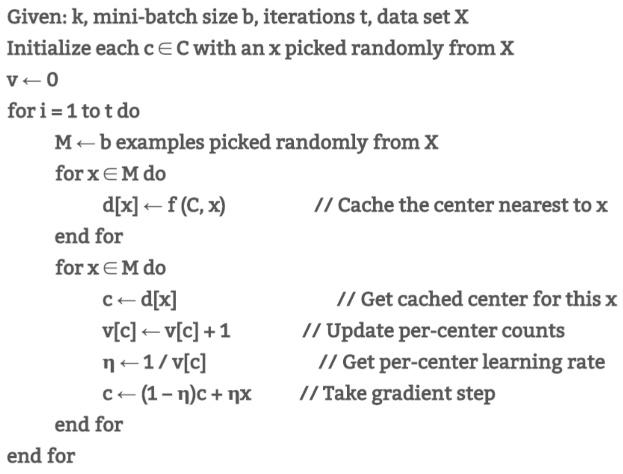


Figure 9. $K$-Means Vs MiniBatch $k$-Means Execution Times



```
Given: k, mini-batch size b, iterations t, data set X
Initialize each c ∈ C with an x picked randomly from X
v ← 0
for i = 1 to t do
    M ← b examples picked randomly from X
    for x ∈ M do
        d[x] ← f (C, x)            // Cache the center nearest to x
    end for
    for x ∈ M do
        c ← d[x]                    // Get cached center for this x
        v[c] ← v[c] + 1            // Update per-center counts
        η ← 1 / v[c]               // Get per-center learning rate
        c ← (1 – η)c + ηx          // Take gradient step
    end for
end for
```

Figure 10. The algorithm for MiniBatch $k$-Means [1].

Randomly drawn sub-sets of descriptors are used to form mini-batches, where each descriptor is assigned to its nearest cluster centroid. The centroids are constantly updated by calculating the average of the points assigned to the cluster to which the centroid belongs to. The algorithm iterates through each mini-batch up to a maximum iteration threshold or until convergence is achieved.

## Appendix II - RF Codebook

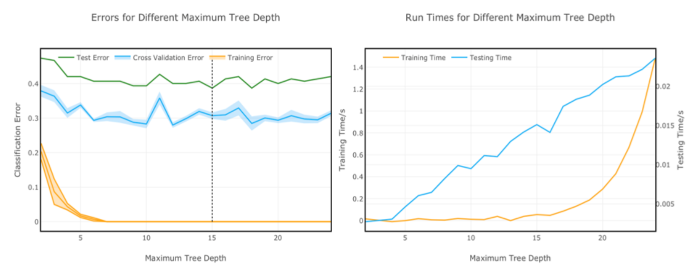


Figure 11. Classification accuracy (left) and time complexity (right) for various tree depths

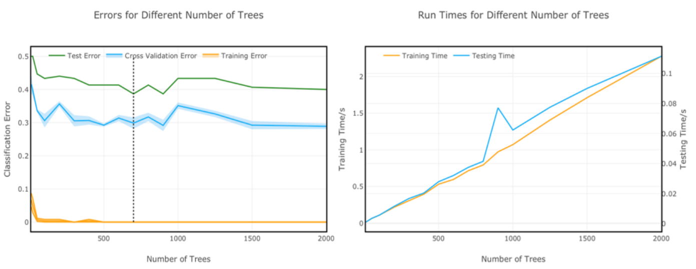


Figure 12. Classification accuracy (left) and time complexity (right) for varying $N_{trees}$

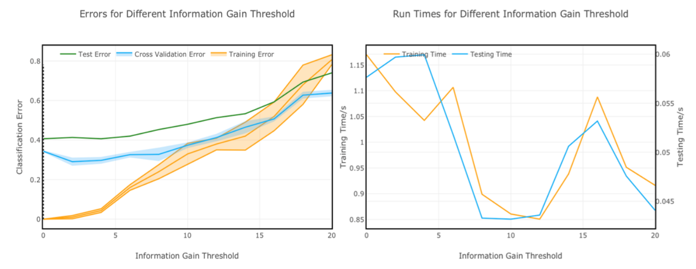


Figure 13. Classification accuracy (left) and computation times (right) for varying number of random splits

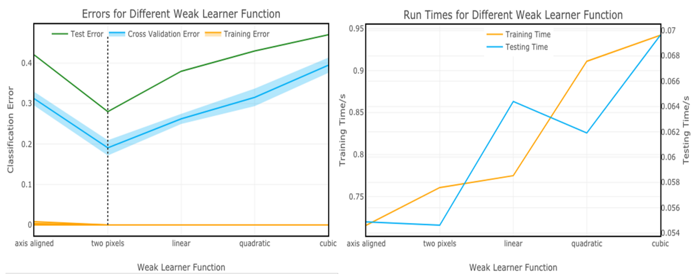


Figure 14. Classification accuracy (left) and time complexity (right) for various weak learners

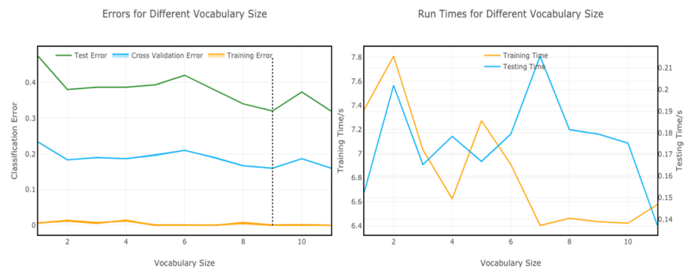
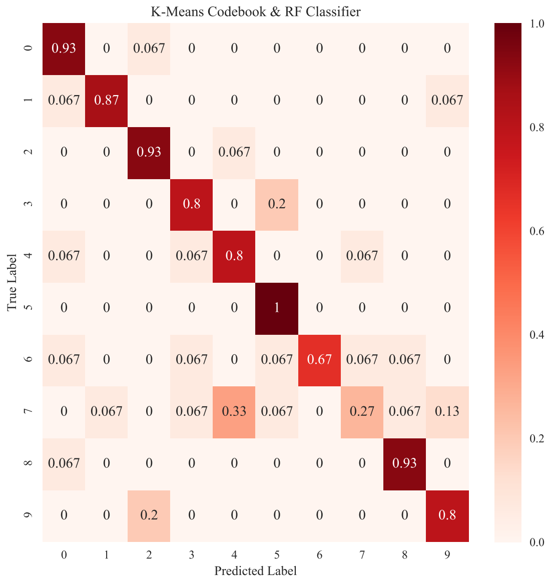


Figure 15. Classification accuracy (left) and time complexity (right) for varying vocabulary size

## Appendix III - Comparisons

| **Codebook** | ***k*-Means** | **RF** |
|---|---|---|
| Number of Centroids | 256 | - |
| Weak Learner | - | two-pixel |
| Number of Trees | - | 9 |
| Maximum Tree Depth | - | 5 |
| Number of Splits | - | 5 |
| Codebook Vocabulary Size | 256 | $\leq$144 |
| Weak Learner | two-pixel | two-pixel |
| Number of Trees | 950 | 300 |
| Maximum Tree Depth | 10 | 14 |
| Minimum Samples at Node | 7 | 6 |
| Number of Splits | 7 | 6 |
| Test Accuracy | 80.2% | 70.3% |
| Vector Quantisation Time | 17.5s | 8.9s |
| Training Time | 24.0s | 6.5s |
| Testing Time | 0.028s | 0.033s |

Table 1. Optimal hyperparameter configurations for $k$-Means and RF codebooks



| True \ Predicted | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0.93 | 0 | 0.067 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 0.067 | 0.87 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0.067 |
| 2 | 0 | 0 | 0.93 | 0 | 0.067 | 0 | 0 | 0 | 0 | 0 |
| 3 | 0 | 0 | 0 | 0.8 | 0 | 0.2 | 0 | 0 | 0 | 0 |
| 4 | 0.067 | 0 | 0 | 0.067 | 0.8 | 0 | 0 | 0.067 | 0 | 0 |
| 5 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| 6 | 0.067 | 0 | 0 | 0.067 | 0 | 0.067 | 0.67 | 0.067 | 0.067 | 0 |
| 7 | 0 | 0.067 | 0 | 0.067 | 0.33 | 0.067 | 0 | 0.27 | 0.067 | 0.13 |
| 8 | 0.067 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0.93 | 0 |
| 9 | 0 | 0 | 0.2 | 0 | 0 | 0 | 0 | 0 | 0 | 0.8 |

Figure 16. Confusion Matrix: $k$-Means Codebook & RF Classifer

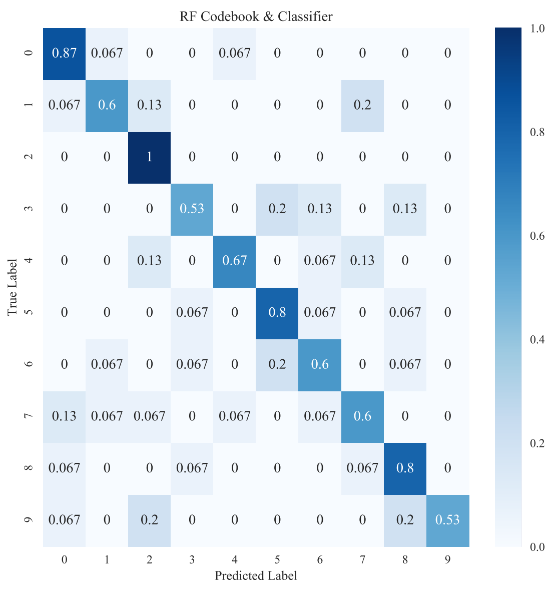


Figure 17. Confusion Matrix: RF Codebook & RF Classifer

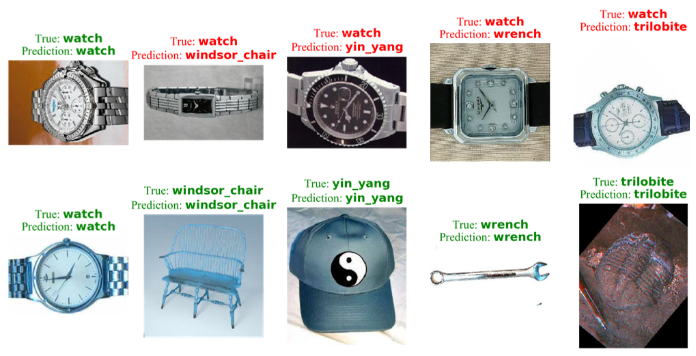


Figure 18. RF Classifier with $k$-Means codebook: examples of worst accuracy on "watch" class (top), examples of correct classifications of respective classes (bottom) .

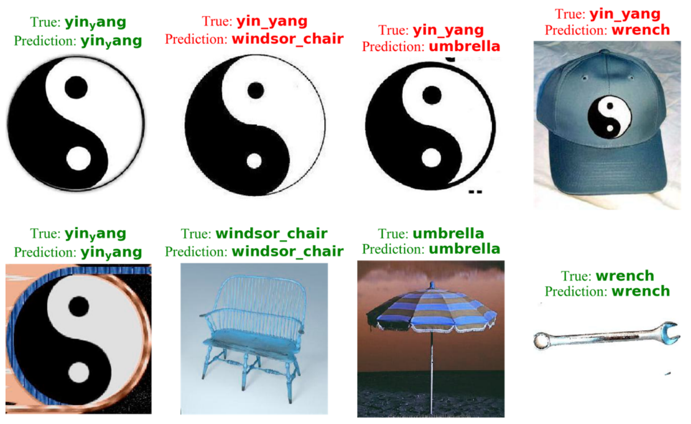


Figure 19. RF Classifier with RF codebook: examples of worst accuracy on "watch" class (top), examples of correct classifications of respective classes (bottom) .

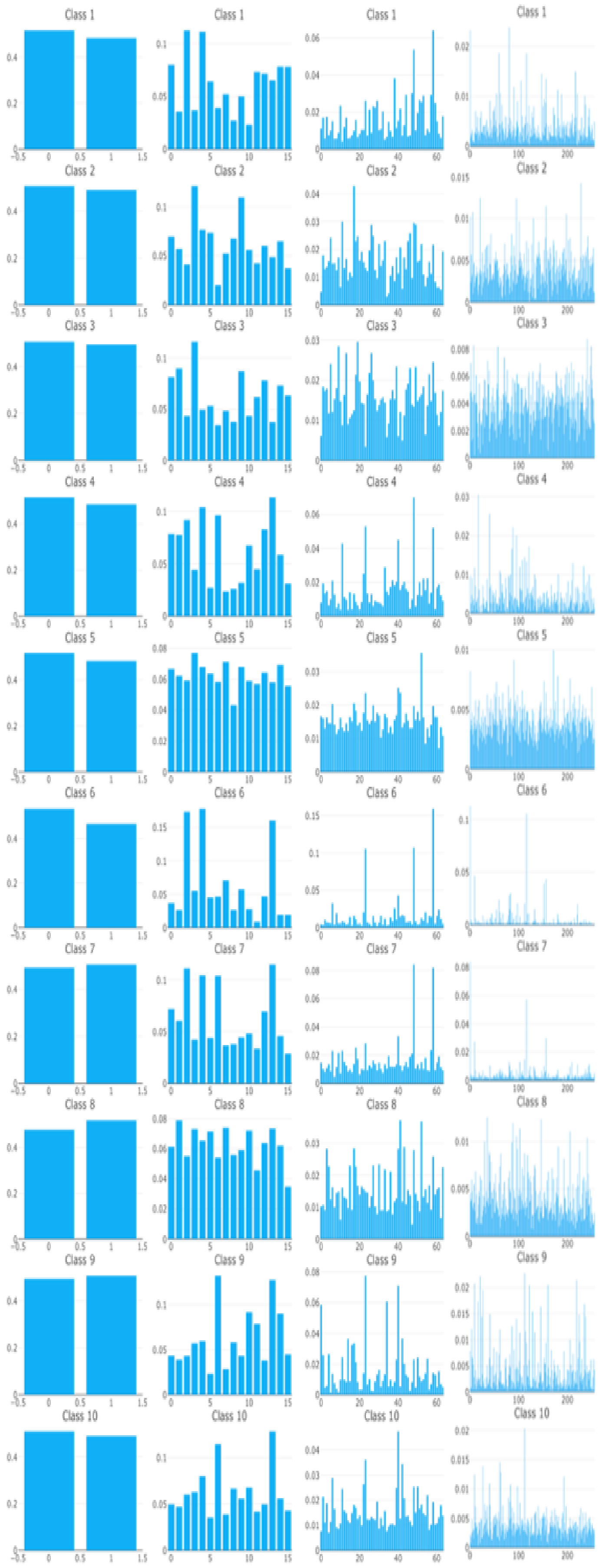


Figure 20. $k$-Means Codebook BoVW Histogram Representations for $K = 2$, $K = 16$, $K = 64$, $K = 256$ from left to right

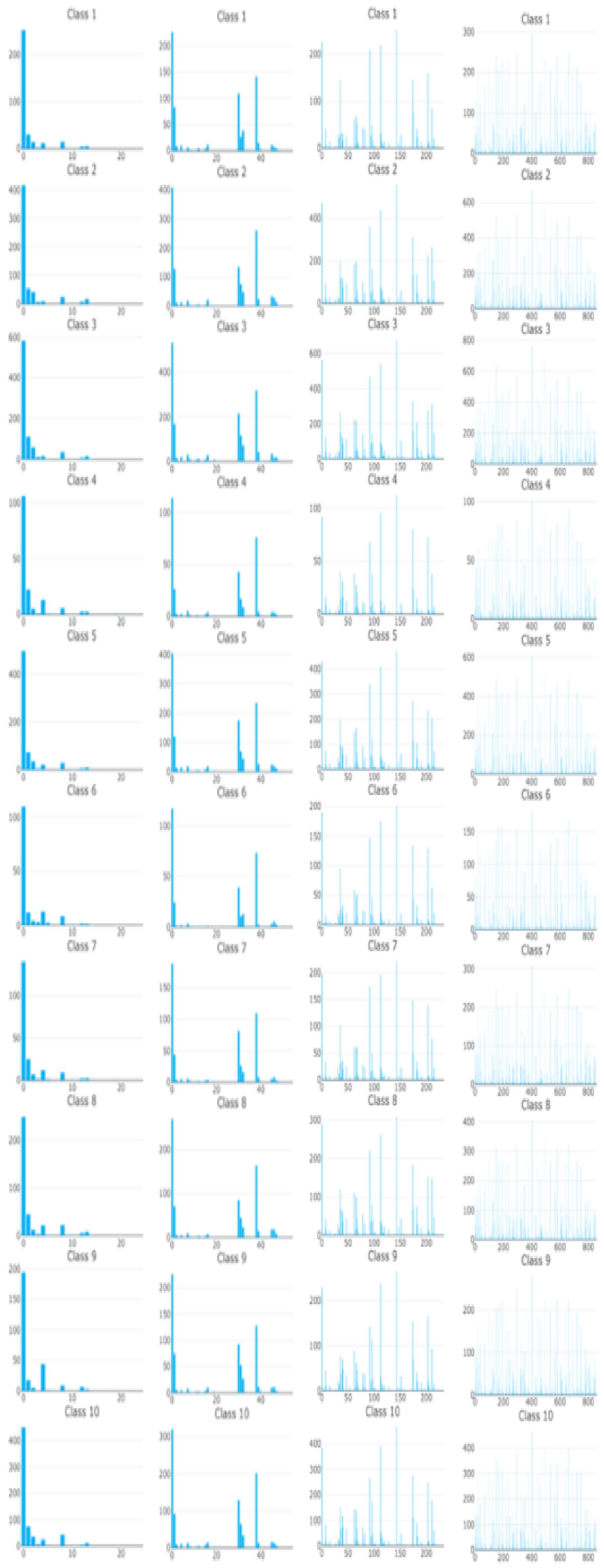


Figure 21. RF Codebook BoVW Histogram Representations for numtrees = 1, numtrees = 2, numtrees = 8, numtrees = 32 from left to right