Architecture Classification for Indian Monuments

Tanvi Sahay
College of Information and Computer Science
University of Massachusetts
Amherst, MA 01002
tsahay@cs.umass.edu

Ankita Mehta
College of Information and Computer Science
University of Massachusetts
Amherst, MA 01002
amehta@cs.umass.edu

Shruti Jadon
College of Information and Computer Science
University of Massachusetts
Amherst, MA 01002
sjadon@cs.umass.edu

Abstract

In this project, the task of architecture classification for monuments and buildings from the Indian subcontinent was explored. Five major classes of architecture were taken and various supervised learning methods, both probabilistic and non-probabilistic, were experimented with in order to classify the monuments into one of the five categories. The categories were: 'Ancient', 'British', 'IndoIslamic', 'Maratha' and 'Sikh'. Local ORB feature descriptors were used to represent each image and clustering was applied to quantize the obtained features to a smaller size. Other than the typical method of using features to do an image-wise classification, another method where descriptor wise classification is done was also explored. In this method, image label was provided as the mode of the labels of the descriptors of that image. It was found that among the different classifiers, k nearest neighbors for the case of descriptor-wise classification performed the best.

1 Introduction

Vision based classification of objects has gained immense popularity in the past few years and is being used for a multitude of tasks ranging from face recognition to vehicle detection. While in the earlier days, this problem would have been solved by measuring the similarities between features of two images and classifying images with most matching features as belonging to the same class, automatic classification has now become possible, thanks to the sophisticated machine learning algorithms in use today. In this project, certain supervised machine learning classification techniques were applied to the exciting task of classification of different monument images based on their architecture.

India, a country which has seen the rise and fall of several dynasties throughout the ages, has a rich architectural and cultural inheritance to behold. The history and heritage of every kingdom is greatly preserved in the monuments and buildings that were built during their reign. Decades later, while those dynasties no longer continue to exist, these preserved relics still flaunt the grandeur of those kingdoms through the uniqueness of their architecture. What is worth noting is that these architectural styles, though unique in themselves, are often inspired by others, which makes determining the architecture of a particular monument through visual observation a rather challenging task. In this project, an attempt to determine how well machine learning algorithms perform in such a situation was made. Local features have been found to be useful for learning the keypoints of an image in the past, and so the ORB local features were used in this project.

For the different methods experimented with during the course of this project, supervised machine learning techniques were not able to provide a high accuracy of classification in the chosen task. However, the high overlap between different architectures made this result intuitive. Also, most methods were still able to map which architectures are similar to which others and it was concluded that most supervised machine learning methods can successfully learn the local features of images and classify similar images together under a single class.

2 Related Work

This section summarizes some of the most relevant prior works referred to for this project.

2.1 Classification of Archeological Monuments for Different Art Forms with an Application to CBIR [5]

This paper finds similarity between two images using the Canberra distance as a similarity measure. While classifying images, the feature extraction method plays a very crucial role in deciding the accuracy of classification. These features can be a combination of texture, color, intensity, shape etc. Recently, Edge Detection methods have become common as they give best results in terms of recognizing the relevant structure of monuments. In this paper, shape and texture features have also been computed. It also employs certain morphological methods such as erosion and dilution to enhance certain features in order to better detect the similarity between images.

2.2 Image based Monument Recognition using Graph based Visual Saliency [6]

Image classification of well-known monuments in Heraklion, Crete, Greece has been done by utilizing Graph Based Visual Saliency (GBVS) and employing Scale Invariant Feature Transform (SIFT) or Speeded Up Robust Features (SURF). The paper uses the local feature extraction techniques SIFT and SURF and computes similarity between two images to classify the query image. It proposes addition of the Graph Based Visual Saliency method in the pre-processing step in order to reduce the time performance of both SIFT and SURF.

2.3 Fast Image Classification for Monument Recognition [3]

This paper proposes two new methods of classifying images using kNN classifiers in combination with local features. Four feature extraction methods SIFT, SURF, ORB and BRISK were used and outputs obtained for kNN classifier were compared with several state-of-the-art alternatives. In the first proposed method, for each local feature point, k nearest local features are taken as candidate matches. Pairwise image distance matching is then done but only the candidate matching local features are considered. In the second approach, an image is classified to a particular class depending on how many descriptors are said to belong to that class i.e. descriptor wise classification instead of image wise classification is performed in this case. It was concluded that these proposed methods performed significantly better than state-of-the-art methods.

2.4 Classification of Images using Support Vector Machines [4]

Support Vector machines have been widely applied in the field of machine vision, since they are binary classifiers and can handle the multiple classification tasks. As discussed in this paper, two approaches are used i.e. One-Against-One (1A1) and One-Against-All (1AA). 1A1 approach constructs a single classifier for each class and 1AA is the most common SVM multiclass approach. The performance of these classifiers has been compared and evaluated to establish the best classifier for architecture recognition. It is concluded in the paper that not much difference is present in the two classifiers and that the final choice of classifier depends on the user's personal preference.

3 Dataset

In this project, the dataset used comprised of images of Indian monuments, collected from several sources such as personal blogs, open source image websites such as pintrest and flickr and other educational sites such as wikipedia. A total of 5 architectural classes were considered while collecting

these images, which were labeled as: 'Ancient', 'British', 'IndoIslamic', 'Maratha' and 'Sikh'. A separate python script was written to scrape the sources mentioned above to extract all the relevant images. Once obtained, these images were manually scanned through to make sure only noiseless images were taken into consideration. Figure 1 gives an example of the images of each class that were taken into consideration. Beginning at top left and going clockwise, the images represent the classes 'IndoIslamic', 'Maratha', 'British', 'Sikh' and 'Ancient' respectively. Figure 2 gives an example of a noisy image.

Figure 1: Example images for each dataset

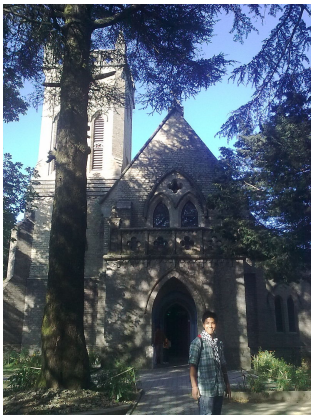

Figure 2: Noisy Image Example

ORB (Oriented FAST and Rotated Brief) feature extraction technique was used to get the local features of each image. First, key points of the image were found. These key points are interest points or corners in the images, which stay the same regardless of orientation of the image, which makes this method ideal for this project. Once key points are detected, another corner detection method is applied to rank the corners detected and only the N most important ones are selected. Once the key points are available, BRIEF is used to convert the them into binary descriptors of size 32. For this project, 500 key points were take for each image and the total feature vector per image was of the size 500 x 32. Due to the large size of this matrix and the variation in sizes and pixels of images, which give different number of descriptors for different images, vector quantization was employed to re-size each feature vector to a 15 x 32 matrix. Feature vectors for each class of images was stored in separate csv files to facilitate their usage with different classification methods.

While the above features are useful for finding similarities between images, the method of descriptor-wise classification considered each descriptor to be a separate input, with 32 features each. Then, one image gave 500 descriptors i.e. there are 500 training inputs per image and 32 features per training input.

4 Proposed Solution

In this project, two approaches were explored in order to classify images based on architecture. In the first approach, a quantized set of feature vectors for each image was used to train supervised learning models such as kNN, logistic regression, support vector machines and random forests and query or test images were passed through these trained models to analyze their prediction performance. In this case, image classification was done based on an overall similarity between the images. In the second approach, local feature based method proposed in [2] was employed with kNN to perform per descriptor classification for each image. In this case, final class of the query image is taken as the class to which a majority descriptors belong. Figure 3 shows the experimentation pipeline for approach 1. For approach 2, no vector quantization is performed. Each component of the pipeline has been explained in detail in the following subsections. Data collection has already been explained in the above section and so will be omitted here.

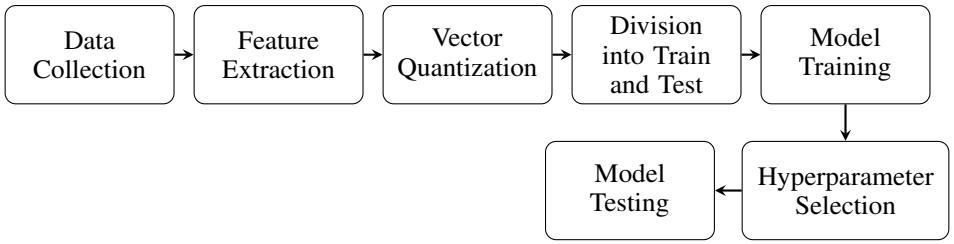

Figure 3: Experimentation Pipeline

4.1 Feature Extraction

For any successful classification task, appropriate features need to be extracted. In the case of image processing, these features should be such that they can map all the factors that provide the image or class its uniqueness. Through a study of past experiments, it was learned that for tasks such as monuments classification, local features work well. Of the various local feature extraction techniques available, ORB(Oriented FAST and Rotated BRIEF) was used as to extract features for the image dataset as it's performance is almost comparable to those of SIFT and SURF, other more commonly used techniques but it is faster than both of those methods. ORB, as the name suggests, uses the FAST algorithm to compute key points in the image and BRIEF algorithm to compute the descriptors of those key points. Harris Corner Detection is employed before converting the key points to descriptors in order to account for the edge sensitivity of FAST and only the first N key points are picked out. Figure 4 shows an example image with all its key points after application of ORB.

Typical features for an image can be edges, corners, blobs or ridges. However, features that have variations with direction are better than others as they are more likely to be unique. Intuitively, corners in an image are more unique and descriptive than edges or plain areas. The FAST key point detector considers a circle of 16 pixels around each pixel p and if 'n' (typically 12) pixels out of these 16 have their intensity value lying within the range $I_p + T$, where I_p is the intensity of the query pixel and T is a manually chosen threshold, that pixel p is taken as a key point. However, this method simply gives key points without taking into account the fact that corners are more preferable than other key points. Therefore, Harris corner detection is applied to order the key points based on how well they describe a corner and only the top N key points are picked. These key points are converted to real valued vectors called descriptors using the BRIEF descriptor method. This method gets random pairs of points from the image based on a gaussian distribution. For each pair (X,Y), if intensity of X is greater than that of Y, the descriptor vector gets a 1 and 0 otherwise. However, BRIEF performs very poorly with rotation of images. Thus, to make ORB orientation invariant, BRIEF features are 'steered' according to orientation of key points. In the end, a length-32 vector is obtained as descriptor for

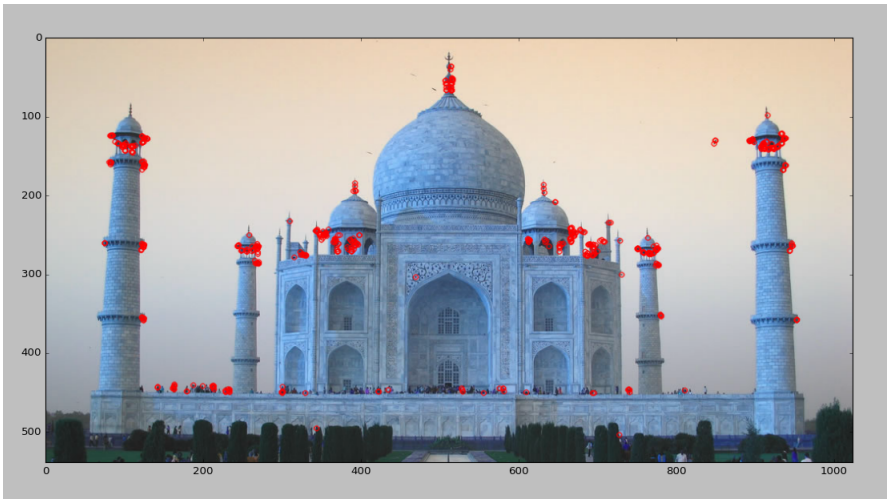

Figure 4: Top 500 ORB descriptors for an IndoIslamic Image

each key point. Thus, for each image, 500 keypoints, each with a descriptor length of 32, comprise of the final feature matrix.

4.2 Vector Quantization

Vector Quantization [1] is a lossy data compression technique often performed using K-means clustering in an attempt to represent data using a smaller number of bits than its original representation. VQ can be defined as a mapping function that maps k-dimensional vector space to a finite set CB = C1, C2, C3,....., CN. The set CB is called codebook consisting of N number of codevectors and each codevector Ci = ci1, ci2, ci3,....., cik is of dimension k. In this project, K-means clustering was used to produce this codebook.

The K-means clustering algorithm takes the number of cluster to be formed, k, as input and returns the centroids of clusters that were formed. For this, centroid are first randomly initialized and all the points are assigned that cluster whose centroid is closest to them. Once these initial clusters are formed, centroids are re-computed. This methods is then iteratively repeated until minimum distortion is achieved. Distortion can be defined as the sum of the squared distances between each observation and its closest centroid. Mathematically, this can be given as:

$$J = \sum_{j=1}^{k} \sum_{i=1}^{n} \parallel x_i^{(j)} - c_j \parallel^2 \tag{1}$$

where c_j represents the centroid of cluster j and $x_i^{(j)}$ represents the data point i in cluster j. The centroids for each cluster are then presented as output to be used as the codebook.

4.3 Supervised Classification Models and their hyperparameters

In this project, the following supervised classification techniques have been employed:

4.3.1 k Nearest Neighbors

The k Nearest Neighbors classifier makes use of a distance function to find the k closest neighbors of a query point. The class of this query point is then taken as the most frequently occurring class among those k neighbors. Different distance functions can be used for the purpose of deciding which

points are to be considered neighbors, but the most common is the euclidean distance, which is the one that is being used for the purpose of this project. Mathematically, it can be written as:

$$d(x_j, x_k) = \sqrt{\sum_i (x_{j,i} - x_{k,i})^2} \tag{2}$$

kNN for the purpose of classification can also be written mathematically in the following manner:

$$f_{KNN}(\mathbf{x}) = \underset{y \in Y}{\operatorname{argmax}} \sum_{i \in K} \mathbb{I}(y_i = y) \tag{3}$$

kNN takes the values of k and type of distance function as hyperparameter. However, for the purpose of this project, distance function has been fixed as euclidean and only the value of k is being varied.

4.3.2 Logistic Regression

Logistic regression is a discriminative classifier in the sense that it models the conditional probability of a data point belonging to a particular class directly, rather than taking a round about approach as in generative classifiers such as Naive Bayes. Logistic regression models the probability $p(y|x)$ which is defined as follows to ensure it's range to be [0,1]:

$$p(y|x) = \frac{exp(w^T x + b)}{1 + exp(w^T x + b)} \tag{4}$$

The value of y that maximizes this equation is chosen as the class of the corresponding data point x. However, the parameters w and b need to be found first in order to find that maxima. These parameters can be found by optimizing the logistic loss with l2 regularization, which can be given as:

$$C \sum_{i=1}^{n} logP(Y = y_i|X = x_i) - ||w||_2^2 \tag{5}$$

Optimization is done using the gradient ascent technique for finding parameters that maximize the above function.
The hyperparameters to be selected for logistic regression are w and b which depend on C, which indicates the strength of regularization, when optimizing the logistic loss.

4.3.3 Support Vector Machines

Support Vector Classifiers are binary classifiers that find the optimum hyperplane lying between two classes such that the margin around the hyperplane is maximized (maximum margin hyperplane). This margin is determined by the points lying closest to the boundary, which are known as support vectors. The decision boundary for SVC is given by:

$$g_{svc}(x) = sign(w^T x + b) \tag{6}$$

Here the vector w is called a weight vector and b is called bias. Model selection is done by minimizing the hinge loss, which can be given as follows, for n training examples:

$$C \sum_{i=1}^{n} max(0, 1 - y_i g(x_i)) + ||w||_2^2 \tag{7}$$

Here C is a regularization parameter which is the hyperparameter to be set in this case. If C is small, there will be narrow margins which get rarely violated. It results in high fit classifier which has low bias and high variance. If C is large, there will be wider margins, many support vectors. It will result in more bias and low variance. In case of non-linear classification, the kernel trick can be used to map data points to a plane where they are linearly separable. Radial Basis Function is one of the most widely used kernels in machine learning, primarily due to its infinite complexity. RBF kernel can be defined as:

$$exp(-gamma * |u - v|^2) \tag{8}$$

where
u = Feature vector of the hyperplane
v = Feature vector of the support vectors
$gamma$ = Constant parameter to be defined by the user

4.3.4 Random Forest

Random Forests are an ensemble technique that take the method of bagged decision trees one step further towards de-correlating the learned trees by considering only a random sub-set of the available features when deciding which variable to split on at each node in the tree. In general, bagged decision trees are a technique where the training set is split into K sub-sets (with replacements) and K separate decision trees are trained on each of these sub-sets. When testing, the same test data is passed through each tree and outputs for each are recorded. In case of classification, the mode of these outputs is taken as the class of that query data. In case of random forests, when fitting a decision tree, only a subset of the possible splits is considered while fitting data. This ensures further de-correlation as now, different splits are considered while fitting different trees. Because of this randomness, bias of the set may be increased but variance gets decreased due to averaging. Decrease in variance is usually more than the compensating increase in bias, hence yielding a better model.

4.4 Cross-Validation

Cross validation is a technique used to estimate the testing error on unseen data in order to determine how well a particular classifier would generalize, without making use of the actual test data. This is done in order to make sure that the hyperparameters of the classifier can be optimized without the risk of learning anything about the test data, which is possible if the classifier is run with the same test data again and again. The method used for selecting hyperparameters for this assignment has been the 3-fold cross validation. In general, the K-fold cross validation divides the training data set into K almost equal subsets. Of these, K-1 are used for training the classifier while the remaining set is used to test it. Hyperparameters are then selected by minimizing the error rate, which can be given as:

$$\frac{1}{n}\sum_{i=1}^{n}\mathbb{I}(y_i \neq \widehat{y_i}) = \frac{1}{n}\sum_{i=1}^{n} Err_i \tag{9}$$

Here n is the number of data points in the test data. In case of a three fold cross validation, the data is divided into three equal sized subsets and testing error is calculated on each subset, for varying values of the hyperparameter. The value that minimizes the error rate is then selected as the final hyperparameter value and the classifier is then trained again using the entire training set. This model is considered the optimum model for testing with previously unseen data. Data is generally partitioned sequentially in order to make sure that no data point is missed while performing cross validation.

4.5 Unsupervised Learning models

K-means clustering algorithm was also experimented with for the purpose of this project:

4.5.1 K-Means Clustering

The K means clustering algorithm, as define above for the case of vector quantization, was used to find centroids of each of the five clusters of training data, where each cluster should ideally represent each data class. Performance of the clusters was analyzed using the Silhouette Score. This score is equal to the mean silhouette coefficient of each data sample. The silhouette coefficient of a point is a measure of how well the point has been assigned to its respective cluster. This coefficient, s, is defined in terms of two variables, a, which is the mean intra-cluster distance and b, which is the mean nearest-cluster distance. For a single data point i, a is computed by taking the mean of distance of i with every point in that cluster and b is computed by taking the distance of that point from the cluster nearest to it, to which it does not belong. Then, given a and b, s can be defined as:

$$s = \frac{b-a}{max(a,b)} \tag{10}$$

It can be seen from the equation above that the value of silhouette score should be between -1 and 1. A value close to 1 means that the data is well clustered while one close to -1 means that the data should belong to a neighboring cluster. A value close to 0 indicates that the data point lies on the border of two clusters.

5 Experimentation and Results

In this project, experiments were conducted in three different approaches i.e., Supervised learning with Image based classification, Supervised learning with descriptor based classification and unsupervised classification. Whole experimentation was done on the image dataset consisting of 500 monument images over the five categories: British, Ancient, Indo-Islamic, Maratha and Sikh. Cross-validation was done for a range of hyperparameter values for various classifiers and accuracy scores were plotted. Each approach has been explained in detail below.

5.1 Unsupervised Learning

In this approach, an attempt to cluster the data using K-means clustering was done. The silhouette score for this was found to be 0.00435270170603. As explained above for the silhouette score, a value close to 0 indicates poor clustering and is indicative of the fact that different clusters are overlapping. This can also be corroborated from the plot of the first two dimensions of the data, as shown in figure 5. Due to such prominent overlap, unsupervised classifiers did not perform well and so, supervised classifiers were explored.

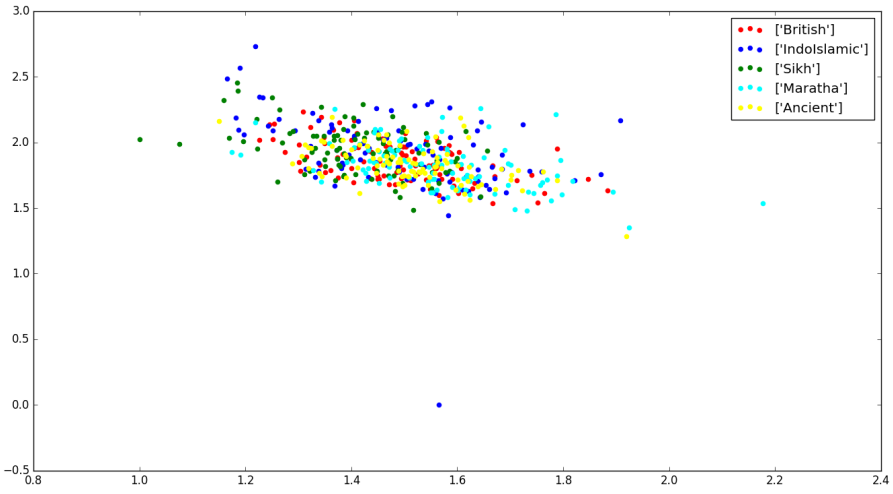

Figure 5: Scatter Plot of the first two dimensions of training data

5.2 Approach 1: Supervised learning with Image based classification

In this approach, image-wise classification was done, where one image was considered one input point in the training set. This training data set consisted of 400 images and each image was converted to a 15x32 feature vector. The test data set consisted of 100 images with a division of 20 images per class. The 5-fold cross validation was applied to find the best hyperparameter value for the following classifiers.

5.2.1 kNN

In this approach for kNN, the hyperparameter was the number of nearest neighbors considered. This range was taken from 1 to 50, with steps of 1. For this range, CV was performed and the resultant

graph has been shown in figure 6. As can be observed, the best cross-validation accuracy was found at k=10. For this k, test accuracy was found to be 0.32 or 32%.

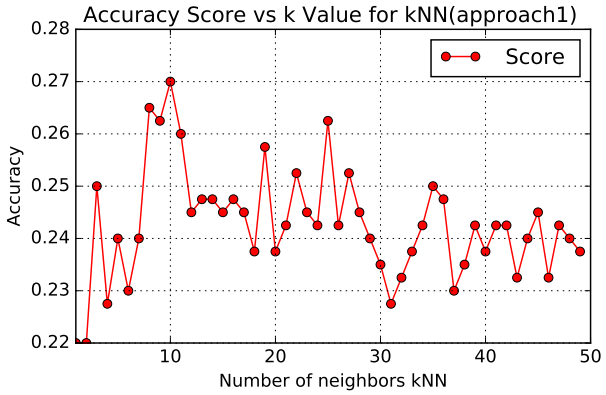

Figure 6: kNN score for different hyperparameter values for image-wise classification

5.2.2 Logistic Regression

Since logistic regression is a probabilistic classifier, the probability of an image belonging to each class was computed separately. For 20 test images of each case, these probabilities were computed for each image and an average was taken. This provided us with the overall probability with which our classifier assigns an image to each class. These probabilities are shown in table 1. The case which was assigned highest probability has been highlighted in bold. Overall accuracy of the logistic regression classifier was found to be 0.296 or 29.6%.

Table 1: Probabilistic output for each class using Logistic Regression

	Ancient	British	IndoIslamic	Maratha	Sikh
Ancient	**0.36918551**	0.17033972	0.15058176	0.23289506	0.07699795
British	0.16164913	0.19997659	0.18631509	0.20826837	**0.24379082**
IndoIslamic	0.20120497	0.14231259	0.26214085	0.13165389	**0.2626877**
Maratha	**0.38405067**	0.11368019	0.12213835	0.30496206	0.07516873
Sikh	0.15851569	0.09137546	0.28275484	0.12325935	**0.34409467**

5.2.3 Support Vector Classifier

In this approach for C-Support Vector Classifier, the hyperparameter was the value of the regularization parameter C. This range was taken from 0.1 to 10, with steps of 0.1. For these values of C, cross validation was performed and the resultant graph has been shown in figure 7. As can be observed, the best cross-validation accuracy was found at C=1.2 to be equal to 0.29. For this C, test accuracy was found to be 0.3 or 30%.

5.2.4 Random Forest

For this experiment, number of trees in the forest were varied from 50 to 300 on a scale of 5. For this range maximum cross validation accuracy was found at number of estimators = 115. The test data accuracy for this value was 0.25 or 25%. The graph for this cross-validation instance has been shown in the figure 8. The drawback of this approach was that because random forests choose training data sub-sets and splitting sub-sets randomly, accuracy score was found to be different for every initialization.

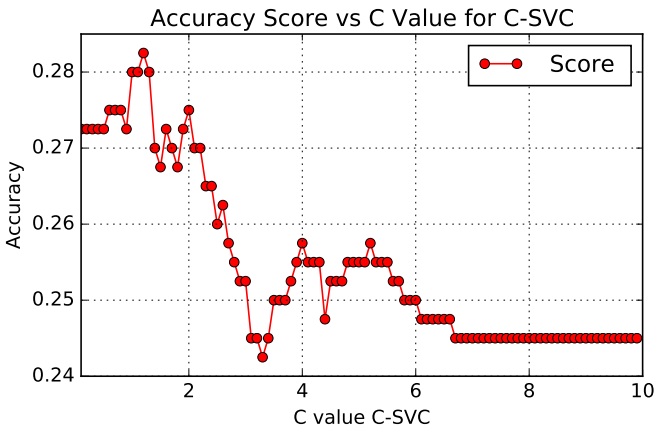

Figure 7: Accuracy score for SVM for different hyperparameter values

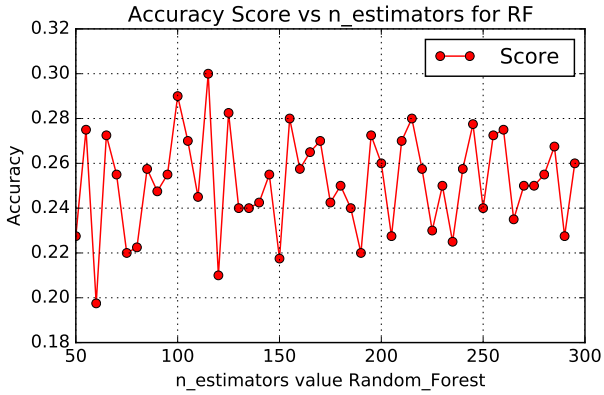

Figure 8: Accuracy score for Random Forests for different hyperparameter values

5.3 Approach 2: Supervised Learning with Descriptor based classification

In this approach, descriptor-wise classification was done, where one key point descriptor was considered as one input point in the training set, with output label the same as label of the image. For example, for a single image belonging to the class 'Ancient', all descriptors were taken as separate training points and each was given the label 'Ancient'. All 500 images were taken as training data and a total of 249086 training points were obtained for all 5 classes. Testing was done using 10 images belonging to each class, giving a total of 50 test images. The following supervised classifiers were trained for this new dataset.

5.3.1 k Nearest Neighbors

The kNN classifier was fit using the 249086 training points and tested for values of k ranging from 1 to 30, with steps of 1. Highest accuracy was achieved at k=23 and was equal to 0.46 or 46%. This was also the highest accuracy achieved in this experiment. The accuracy versus k plot has been shown in figure 9.

For k=23, table 2 shows the division of classified images amongst each class. 10 test images for each class label are considered.

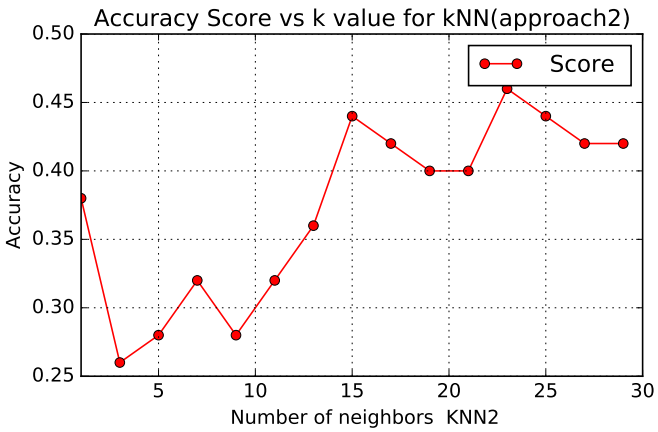

Figure 9: kNN score for different hyperparameter values for local feature-wise classification

Table 2: Division Of Classified Images

	Ancient	British	IndoIslamic	Maratha	Sikh
Ancient	6	0	1	1	2
British	1	4	0	2	3
IndoIslamic	2	1	6	0	1
Maratha	1	2	2	4	1
Sikh	3	0	0	3	4

5.3.2 Support Vector Machine

For the dataset given above, A C-SVC was also trained. However, due to the large amount of data to be processed, each iteration took several hours to complete. For C=0.5, the accuracy was found to be equal to 0.33 or 33%. However, due to time constraint and the high processing required by SVC, this method was not explored further.

5.4 Overall Result

Table 3 shows the test data accuracies for all supervised models along with the hyperparameter values for which those accuracies were obtained. It can be noted that the descriptor-wise kNN classifier performed the best for k = 23, with accuracy score of 0.46.

Table 3: Overall Comparison between different classification models

Model	Accuracy	Hyperparameter Value
kNN-approach 1	0.320	k=10
kNN-approach 2	0.460	k=23
LR	0.296	
CSVC	0.300	C=1.2
Random Forest	0.25	n_trees=115

6 Discussion and Conclusions

From the above results, several conclusions can be drawn, which can be broadly termed as domain conclusions and experimental conclusions.

1. **Domain Conclusions:** From table 2 it can be seen that images belonging to 'Ancient' and 'IndoIslamic' architectures were classified with a higher accuracy as compared to the other architecture types. The reason for this is that both these architectural types have a rich

and unique database, with a large number of monuments spread all over the country. As opposed to that, other architectures are more regional and have a comparatively smaller database. Because of this, multiple images of the same monument had to be taken from different angles, which resulted in redundant features and thus in poor classification accuracy.

Another conclusion that can be derived from the results shown above is that the problem of architecture classification is not an easily solved one. This is due to the fact that different architectures are inspired by each other and have several features that they borrow from one another, which makes their features very similar and not easily separable into unique and distinguishable clusters. This can be seen in table 2 as well as 1. Even for the best classifier, the highest per class recognition accuracy did not exceed 60%, which shows that each architecture has a mixture of features taken from other architecture styles.

2. **Experimental Conclusions:** From the experiments conducted in this project, it can be concluded that for a relatively small and overlapping dataset, most supervised learning classifiers perform poorly, due to the fact that they are not able to map the high overlap present in the data successfully. This problem cannot be solved unless the data is de-correlated using either some pre-processing techniques or extracting a set of more discriminative features.

   It can also be concluded that the descriptor-wise image classification performs much better than the image-wise classification, especially for a small data set. This can be chalked up to the fact that vector quantization, though helpful in decreasing the processing time and computation, causes loss of information which has a degrading impact on classification accuracy. The descriptor wise classification method keeps all information intact, which make the dataset large and increases computation time, but also provides better classification. The same has been shown in table 3.

In conclusion, it can be stated that the task of architecture classification was performed with decent accuracy for ORB features using descriptor-wise classification with k Nearest Neighbors. While classification of data with such high overlap is a challenging task, further experimentation in this field can still be done by exploring global feature extraction techniques such as GIST and include color descriptors such as color histograms. Moreover, the latest classification techniques such as multilayer perceptrons, recurrent and convolutional neural networks can also be explored for this type of dataset.

References

[1] G. Amato, F. Falchi, and P. Bolettieri. Recognizing landmarks using automated classification techniques: Evaluation of various visual features. In *Advances in Multimedia (MMEDIA), 2010 Second International Conferences on*, pages 78–83, June 2010.

[2] Giuseppe Amato, Fabrizio Falchi, and Claudio Gennaro. Geometric consistency checks for knn based image classification relying on local features. In *Proceedings of the Fourth International Conference on SImilarity Search and APplications*, SISAP '11, pages 81–88, New York, NY, USA, 2011. ACM.

[3] Giuseppe Amato, Fabrizio Falchi, and Claudio Gennaro. Fast image classification for monument recognition. *J. Comput. Cult. Herit.*, 8(4):18:1–18:25, August 2015.

[4] M. S. Bhatt and T. P. Patalia. Genetic programming evolved spatial descriptor for indian monuments classification. In *2015 IEEE International Conference on Computer Graphics, Vision and Information Security (CGVIS)*, pages 131–136, Nov 2015.

[5] P. Desai, J. Pujari, N. H. Ayachit, and V. K. Prasad. Classification of archaeological monuments for different art forms with an application to cbir. In *Advances in Computing, Communications and Informatics (ICACCI), 2013 International Conference on*, pages 1108–1112, Aug 2013.

[6] Georgios Triantafyllidis and Gregory Kalliatakis. Image based monument recognition using graph based visual saliency. *ELCVIA Electronic Letters on Computer Vision and Image Analysis*, 12(2), 2013.