Tuijin Jishu/Journal of Propulsion Technology
ISSN: 1001-4055
Vol. 45 No. 2 (2024)

Performance Analysis of Feature Combination in Object Classification

[1]Amitav Mahapatra, [2]Prashanta Kumar Patra
[1] Biju Patnaik University of Technology, Rourkela, Odisha, India.
[2] SOA University, Bhubaneswar, Odisha, India.

Abstract: — The objective of this paper is to evaluate the strategy of feature combination i.e. to show the effectiveness of feature combination over single features (in terms of classification accuracy) and thus to study the behavior of recognition metrics by changing certain parameters. We can conclude from the results that feature combination is quite efficient in practical approach. The addition of new features to the existing ones gradually enhances the classification accuracy. Thus such systems can be a great help from the realistic point of view and can be utilized in day to day life. The recognition metrics depend differently on different number of aspects, thus if we can manage such parameters wisely, we can achieve better results by extending the application of feature combination.

Keywords: Feature Combination, Classification Accuracy

1. Introduction

The ever-expanding realm of computer vision offers a vast landscape for extensive research and innovation. Despite decades of focus, crafting a successful object categorization system remains challenging, even with basic image datasets. This difficulty arises from the significant variations within classes and the similarities across different classes, stemming from both intra-class variability and inter-class correlation.

Various tools, such as SIFT [30], HOG [31], and SURF [32], have been introduced to tackle these challenges. However, they exhibit a notable limitation: while proficient in certain object classes, they lack the versatility to encompass all. To address this, the fusion of multiple complementary features becomes imperative. By amalgamating the strengths of different features, a final feature emerges, surpassing the classification accuracy achievable by any single feature alone.

Feature selection as kernel selection:

As a kernel is associated with a feature, implies that kernel selection/combination transforms to feature selection/combination. Cross Validation (CV) is an approach to find a kernel from the set $\{k_1, \dots k_F\}$.

The following are the classes of Feature combination in brief [2]:

1. BASELINES

Deterministically combining kernels for SVM training is covered by two baseline techniques. These are:

Average Method

The easiest approach to merge kernels is average. Definition of kernel function is

$k^*(x, x') = \frac{1}{F}\sum_{m=1}^{F} k_m(x, x')$.

While several combination algorithms have been proposed, this baseline method is worth to be compared with.

Product method

As the name reflects, this baseline method fuses several kernels by multiplication. Here

$$k^*(x,x') = \left(\prod_{m=1}^{F} k_m(x,x')\right)^{1/F}$$

is the kernel function definition that is used as the single kernel in a SVM.

MKL (Multiple Kernel Learning)

This kernel selection method uses known kernels rather than producing fresh ones and discovers an optimal linear and non-linear combination during algorithm training. MKL optimizes linear kernel combinations simultaneously.

$$k^*(x,x') = \sum_{m=1}^{F} \beta_m \, k_m(x,x')$$

and the parameters $a \in R^N$ and $b \in R$ of an SVM. There are strong reasons behind the rigorous use of multiple kernel learning which include a) it provides a larger set of kernels for the selection of kernels and parameters and b) combines data from distinguished sources (images, sound etc)

BOOSTING

Boosting method gathers features in a mixed way that are coded by dissimilar systems. Inspired by the MKL decision function, boosting method is based on adding new kernels iteratively until a stopping criterion is reached. To improve weak learning algorithms, boosting was invented. An updated boosting method, Ada-Boost, combines weak classification functions to strengthen weak classifiers.

The intuitive idea that 'more features lead to higher performance' boosted feature combination research for principled classification systems. Some old feature combination approaches and recent algorithm design methods have been reviewed and adjusted to find the optimal algorithm

SVM classifiers transform feature combinations to kernel combinations. MKL (multiple kernel learning) is a prominent kernel combination method that optimizes kernel weights to provide the greatest combination performance.

$$k^*(x,y) = \sum_{i=1}^{n} w_i \, k_i(x,y)$$

alongside SVM parameters a and b [6]-[10]. Canonical MKL assumes a consistent weighting scheme across the input space, while sample-specific MKL based kernel weights on kernel functions and samples. The former technique boosts performance but increases computation and overfitting. The two contradictory approaches were addressed using group-sensitive MKL.

MKL has been eye-catching recently, but not always. It has no advantage over baseline average combination sometimes. Although some MKL-like kernel combination algorithms have been authorized, these optimization methods are controversial. Optimization algorithms waste memory and are expensive to compute. MKL may have been exaggerated due to lack of comparison with simple but powerful baseline approaches.

Instead of optimizing all parameters at once, [2] suggested using LPBoost to train them independently in two phases. Boosting approaches consistently outperform MKL and baselines.

Traditional techniques of feature combination

Conventional methods for combining features involve (1) merging various feature vectors into a single composite vector and training a classifier on it (Dassigi et al., 2001); (2) forcefully combining all the features into a single long feature vector and reducing its dimensionality using techniques such as PCA (Principal Component Analysis), LDA, etc. The initial two techniques are referred to as "Feature Fusion," while the final one is known as "Decision Fusion."These two strategies dominate "Data Fusion". The aforementioned feature combination strategies can be split into three tiers (low, middle, and high) for effective classification from human perception. With low-level feature combination, features are worked on without refining. The middle level selects a subset of original features. The higher level of feature combination combines classifiers from different systems' programmed feature sets. However, each level has flaws. For low-level feature combinations, worthless, redundant, or incompatible features may cause poor classifier performance on lengthy feature vectors. Sometimes the composite vector has too many dimensions, making computation expensive. Even with advanced feature selection approaches, the middle level may not contain an ideal feature set. The high level of feature combination may lack feature communication in training because it focuses on component classifiers.

2. Literature review

Although feature combination work is extended to different branches like text classification, disease diagnosis, human faces detection etc. here we stick to the works on the Feature combination in 'Object classification'. We elaborately present the literature works carried out in the past, their methods adopted, comparisons, advantages and disadvantages of certain methods involved and finally a summary. So the literature works are as follows:

A new boosting-based feature combination method is proposed [1]. Boosting combines features here. Unlike ordinary boosting, variation boosting trains weak classifiers on multiple feature sets. Finally, weighted voting combines classifiers. Output classifier for that round. Research shows this feature combination technique can combine feature selection, communication, and classifier learning. Schapire (1990) introduced boosting to improve weak learning algorithms. AdaBoost extends boosting and voting classifiers through two classes. A similar feature vector is given to weak classifier components in conventional AdaBoost. Even though all features are different types, each training instance includes a fixed-length feature vector with similar attributes in a specified order. As seen in [1], AdaBoost works better. This boosting strategy trains a middle final classifier from several weak classifiers on samples of each feature vector using system-coded characteristics at each cycle. Each round of this algorithm identifies combinations using weighted voting. Weak learning algorithms can be chosen for AdaBoost and other boosting algorithms like decision trees, neural networks, and others. This boosting variation can be changed in multi-class cases like general boosting. The suggested method in [1] classifies specifically, unlike AdaBoost, which classifies broadly. The variant of boosting showed much better classification performance than traditional approaches on three data sets. Boosting, feature extraction optimization, and neural network parameter adjustment can improve its performance.

Further works on boosting in [2] confirmed its consistency and robustness by extending it to several multiclass setting. In [2] vivid kernel methods of feature combination have been defined in a clear manner which includes baselines, MKL and boosting. The paper proposed formulations based on LPBoost. In particular two methods have been proposed that are inspired by the MKL decision function. They are LPBoost and its multiclass variants i.e. LP-β and LP-B that have been experimented on specific datasets. In MKL solution, combining coefficients are regarded as feature influence on a class, but in multiclass situation, this is false. Thus, multiclass decision-making values all features equally. LP-β chooses three of seven features, while other approaches aim to choose all features. Oxford flower experiment results show MKL and LP-β learning methods are resilient to unnecessary characteristics, but CG-Boosting strategy worsens with time.

Instead of monotonous time-consuming learning procedures in MKL, prior information was introduced into kernel mixing [3]. So, the weight of each attribute in combination depends on how well it classifies the class. Thus, numerous methods that combine local feature weights are presented in [3] along with the new strategy of establishing feature weights based on classification performance. Feature combination is solved with bag-of-words histogram features and kernel-based classifiers. The factor-based algorithm is

1) Each bag of words feature has a kernel.
2) Each kernel trains SVM classifiers.
3) Validation dataset predicts classifier performance.
4) Each class combines first-step kernels into one kernel and uses third-step performance ratings as coefficients.
5) The final classifier predicts validation dataset picture labels using the single kernel.

The fact that prediction ability weights each feature's effect on the final prediction makes this technique strong. To mix features and kernels, performance prior knowledge is calculated after normalising intermediate values. This historical information might be included two ways:

3. Proposed Model

To propose a model that can successfully implement feature combination in object classification by taking certain image datasets as input and with the help of classification algorithm trains and tests the given data, thus displaying the recognition parameters which are analyzed further.

So the goal of the model is twofold: To combine the features with the help of some local feature descriptors and detectors To obtain the recognition parameters and thus analyze them in a graphical manner by changing certain parameters. The proposed model has been designed with respect to the hardware and software configurations.

As the simulation environment (system) has a limited capacity to withstand dynamic changes, so the proposed model has been designed with a minimal scope. Underneath the proposed model, there are a lot of intermediate and individual coding works that are integrated to obtain the desired result.In order to avoid malfunctioning of the system, limited numbers of features are combined.The proposed model has been presented in the form of a flowchart which is shown below.The terms related to the flowchart are thereby described in a nutshell.The details of the flowchart and the parameters are also elucidated

Fig1. Represents flowchart as the result of integration of different intermediate works done on python.

The process starts with the input of the image data sets which are in a specific format (name.image_number) in order to be processed.The images undergo different pre-processing techniques like feature extraction, then feature concatenation etc.The features are concatenated using certain feature descriptors and/or feature detectors (more than one descriptor/detector).After the feature combination takes place, a classifier algorithm is applied (classifier).The classification algorithms included here are Linear SVM and the K-Nearest-Neighbor.The purpose of choosing the algorithm is not reason oriented but because of the motivation that we received from the specific paper.

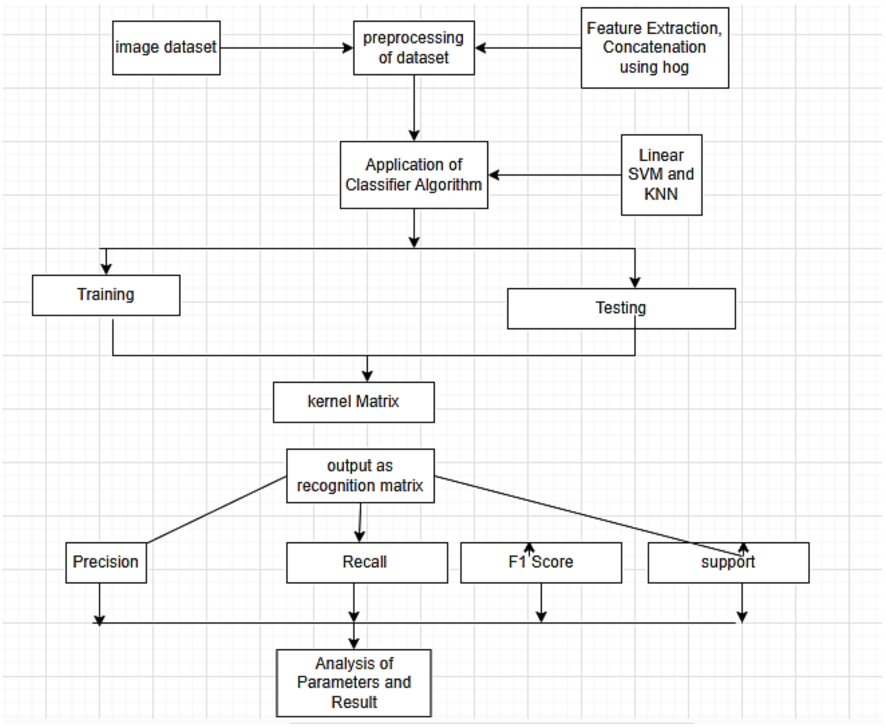


image dataset
preprocessing of dataset
Feature Extraction, Concatenation using hog
Application of Classifier Algorithm
Linear SVM and KNN
Training
Testing
kernel Matrix
output as recognition matrix
Precision
Recall
F1 Score
support
Analysis of Parameters and Result

Fig.1 Flowchart of proposed model

Keeping this aside, the KNN and SVM are the common algorithms applied in feature combination for classification purpose.The classification algorithm splits the data to get training and testing sets.After the images are trained, a kernel matrix is obtained with certain dimension and as the testing images are passed through the kernel matrix, it classifies them resulting the accuracy.The recognition metrics obtained are precision, recall, support and F1 score in SVM and only accuracy rate in kNN.The parameters are changed like the size of the data, types of classifiers etc to analyze its impact on the above metrics especially with precision (accuracy).The result is the trend found after the analysis has been done.Our proposed model resembles that of the early combination model.

3.1 Details of the descriptor algorithms (feature descriptor) used

As we have used two types of descriptors to show the feature combination, here we give a brief about them.

A. Color histogram in HSV space

In image processing, color histogram is the representation of the distribution of colors in an image.For a digital image, color histogram represents the number of pixels that spans the image's color space, the set of all possible colors.HSV stands for hue, saturation & intensity value.The color histogram can be built for any kind of color spaces like the three dimensional color spaces RGB or HSV.

B. HOG

It is used for image recognition and object detection**.**It is basically a feature extraction algorithm that converts an image of fixed size to a feature vector of fixed size.It is based on the idea that local object appearance can be effectively described by the distribution(histogram) of edge directions(oriented gradients)As defined in the introduction chapter, HOG involves the technique of counting occurrences of gradient orientation in localized portions of an image.First the color histogram is applied in order to extract the features.Then the HOG is included to extract the HOG features.Both these features are concatenated to give the effect of feature combination.

3.2 Classifier algorithms used

Two classifier algorithms have been included in our work to show what effect they have on accuracy. They are:

A. kNN

The k-nearest neighbor algorithm is used for classification and regression.In both the cases the input consists of k closest training examples in feature space.In our work we have taken k value as 2.Some parameters included in kNN are n_neighbors, weights, algorithm, leaf_size, metric etc.

B. Linear SVM

Support vector machine is a discriminative classifier or supervised learning methods used for classification, regression and outlier detection.In other words, given labeled training data (supervised learning), the algorithm outputs an optimal hyper plane which categorizes new examples.Binary classification: A classification that classifies the data into two classes. We have included the binary classification scheme in our work.

3.3 Kernels and Kernel matrix

A kernel is a similarity function. In image processing a kernel, a convolution matrix, or mask is a small matrix. It is useful for blurring, sharpening, embossing, edge detection, and more. This is accomplished by means of convolution between a kernel and an image.Every feature corresponds to one or more kernels.

3.4 Terms associated with Recognition parameters:

In the field of image processing and computer vision, the word 'Recognition' plays a vital role. In most of the papers, a term recognition rate is used to show the results of classification. It indicates how correctly the algorithm can detect the test images after it has been trained. The sole purpose behind so much of research work is just to improve this recognition rate thus designing a more robust model.

There are various parameters which can be changed to study the final results. For example: Change in input, Number of input and Change in classification algorithm: SVM, kNN, change in training-testing split ratio

The results of the classification are analyzed by the following metrices:

Precision: (P) The ability of the classifier not to label as positive, a sample that is negative. In simple words, precision is a measure of result relevancy. Precision is defined as the number of true positives (T_P) over the number of true positives plus the number of false positives (F_P)

$$P = \frac{T_P}{T_P+F_P}$$

1. **Recall: (R)** The ability of the classifier to find all the positive samples. In simple words recall is a measure of how many truly relevant results are returned. Recall is defined as the number of true positives (T_P) over the number of false negatives (F_n)

$$R = \frac{T_p}{T_p+F_n}$$

High scores for both i.e. precision and accuracy show that the classifier is returning accurate results (high precision), as well as returning a majority of all positive results (high recall).

2. **F1 score: (F_1)** The harmonic mean of Precision and Recall.

$$F1 = 2\,\frac{P*R}{P+R}$$

3. **Support:** It is the number of samples of true response that a class contains.

4. Result And Discussion

The results and discussions of this paper are based on numerous works undertaken in a trial and error method using python programming.

We have included here three data sets. They are

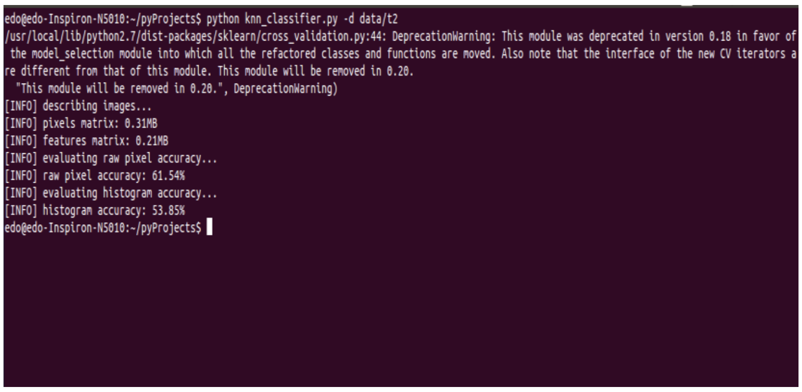
- Data set 1- two classes from Event-8 (bocce & rock climbing, 52 each)[34]
- Data set 2- two classes from Caltech-101(Buddha & lamp, 60 each)[35]
- Data set 3 – two classes from Pets (cats & dogs, 500 each)[36]

In results, first we have evaluated the classification accuracy without any kind of feature combination that is the accuracy simply obtained by using color histogram in HSV space (color descriptor). Then to show the effectiveness of Feature combination over single features, we have again evaluated the classification accuracy by concatenating the existing features with the features of HOG descriptor. We have evaluated the same using kNN classifier. After evaluation of feature combination, we have changed the parameters one by one to check the change in the recognition metrics. The parameters changed are type of classifier, training testing splits and the input size (in terms of number of images) The results obtained are converted to graphs and hence analyzed. In order to obtain desirable results, it was made sure that the system remains stable. The results have been noted under minimal inclusion of libraries, functions and codes; else it produced abnormal results.

The results are as follows:

1. Single feature

a) Data set 1 (bocce & rock climbing, 52 number of images each) Classifier used- kNN

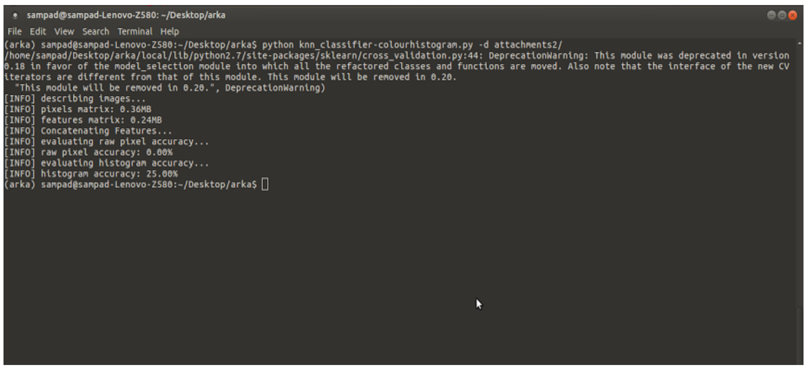


```
edo@edo-Inspiron-N5010:~/pyProjects$ python knn_classifier.py -d data/t2
/usr/local/lib/python2.7/dist-packages/sklearn/cross_validation.py:44: DeprecationWarning: This module was deprecated in version 0.18 in favor of
 the model_selection module into which all the refactored classes and functions are moved. Also note that the interface of the new CV iterators a
re different from that of this module. This module will be removed in 0.20.
  "This module will be removed in 0.20.", DeprecationWarning)
[INFO] describing images...
[INFO] pixels matrix: 0.31MB
[INFO] features matrix: 0.21MB
[INFO] evaluating raw pixel accuracy...
[INFO] raw pixel accuracy: 61.54%
[INFO] evaluating histogram accuracy...
[INFO] histogram accuracy: 53.85%
edo@edo-Inspiron-N5010:~/pyProjects$ 
```

Fig.2: Single feature for data set 1

Accuracy = 53.85 %

b) Data set 2 (Buddha & lamp, 60 each) classifier used- kNN

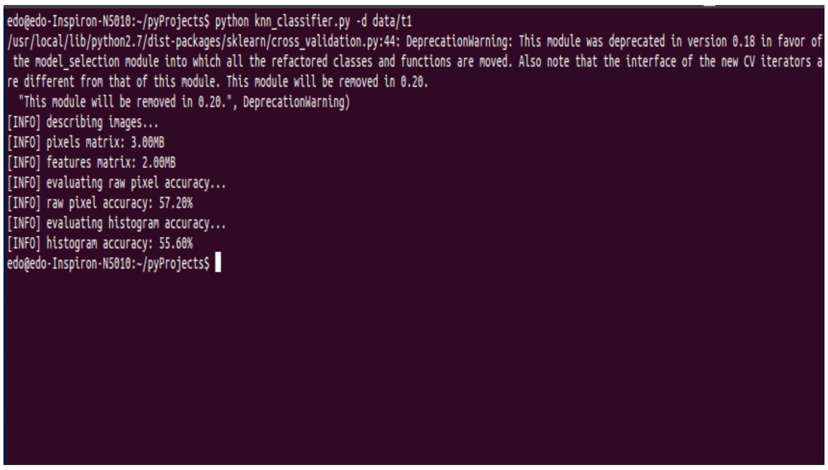


```
sampad@sampad-Lenovo-Z580: ~/Desktop/arka
File  Edit  View  Search  Terminal  Help
(arka) sampad@sampad-Lenovo-Z580:~/Desktop/arka$ python knn_classifier-colourhistogram.py -d attachments2/
/home/sampad/Desktop/arka/local/lib/python2.7/site-packages/sklearn/cross_validation.py:44: DeprecationWarning: This module was deprecated in version
0.18 in favor of the model_selection module into which all the refactored classes and functions are moved. Also note that the interface of the new CV
iterators are different from that of this module. This module will be removed in 0.20.
  "This module will be removed in 0.20.", DeprecationWarning)
[INFO] describing images...
[INFO] pixels matrix: 0.36MB
[INFO] features matrix: 0.24MB
[INFO] Concatenating Features...
[INFO] evaluating raw pixel accuracy...
[INFO] raw pixel accuracy: 0.00%
[INFO] evaluating histogram accuracy...
[INFO] histogram accuracy: 25.00%
(arka) sampad@sampad-Lenovo-Z580:~/Desktop/arka$ 
```

Fig.3: Single feature for data set 2

Accuracy= 25.00 %

c) Data set 3(cats and dogs, 500 each) classifier used- kNN



```
edo@edo-Inspiron-N5010:~/pyProjects$ python knn_classifier.py -d data/t1
/usr/local/lib/python2.7/dist-packages/sklearn/cross_validation.py:44: DeprecationWarning: This module was deprecated in version 0.18 in favor of
 the model_selection module into which all the refactored classes and functions are moved. Also note that the interface of the new CV iterators a
re different from that of this module. This module will be removed in 0.20.
  "This module will be removed in 0.20.", DeprecationWarning)
[INFO] describing images...
[INFO] pixels matrix: 3.00MB
[INFO] features matrix: 2.00MB
[INFO] evaluating raw pixel accuracy...
[INFO] raw pixel accuracy: 57.20%
[INFO] evaluating histogram accuracy...
[INFO] histogram accuracy: 55.60%
edo@edo-Inspiron-N5010:~/pyProjects$ 
```

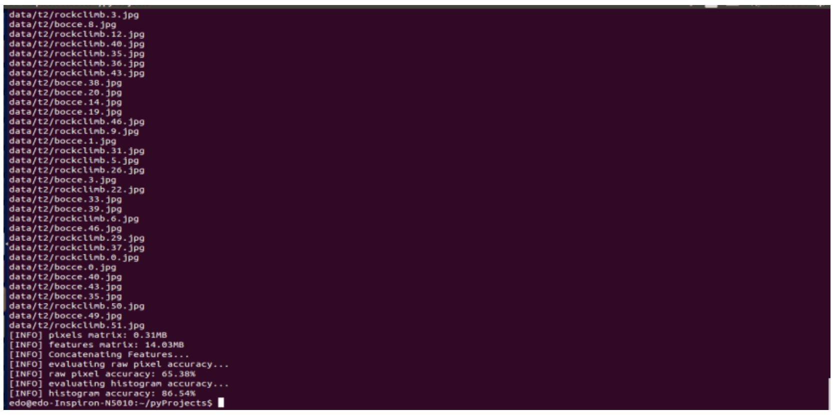
Fig.4: Single feature for data set 3

Accuracy =55.60 %

The above results are obtained by taking one color descriptor i.e. color histogram in HSV space.

2. Feature combination

a. Data set 1 Event (bocce & rock-climbing,52 each) classifier- kNN

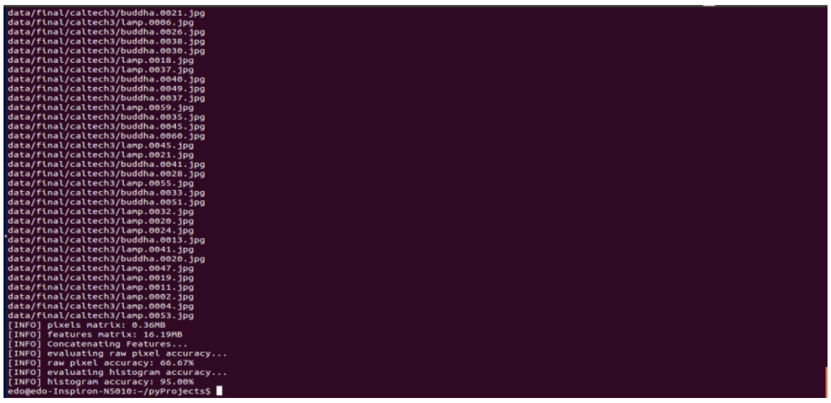


```
data/t2/rockclimb.3.jpg
data/t2/bocce.8.jpg
data/t2/rockclimb.12.jpg
data/t2/rockclimb.40.jpg
data/t2/rockclimb.35.jpg
data/t2/rockclimb.36.jpg
data/t2/rockclimb.43.jpg
data/t2/bocce.38.jpg
data/t2/bocce.20.jpg
data/t2/bocce.14.jpg
data/t2/bocce.19.jpg
data/t2/rockclimb.46.jpg
data/t2/rockclimb.9.jpg
data/t2/bocce.1.jpg
data/t2/rockclimb.31.jpg
data/t2/rockclimb.5.jpg
data/t2/rockclimb.26.jpg
data/t2/bocce.3.jpg
data/t2/rockclimb.22.jpg
data/t2/bocce.33.jpg
data/t2/bocce.39.jpg
data/t2/rockclimb.6.jpg
data/t2/bocce.46.jpg
data/t2/rockclimb.29.jpg
data/t2/rockclimb.37.jpg
data/t2/rockclimb.0.jpg
data/t2/bocce.0.jpg
data/t2/bocce.40.jpg
data/t2/bocce.43.jpg
data/t2/bocce.35.jpg
data/t2/rockclimb.50.jpg
data/t2/bocce.49.jpg
data/t2/rockclimb.51.jpg
[INFO] pixels matrix: 0.31MB
[INFO] features matrix: 14.03MB
[INFO] Concatenating Features...
[INFO] evaluating raw pixel accuracy...
[INFO] raw pixel accuracy: 65.38%
[INFO] evaluating histogram accuracy...
[INFO] histogram accuracy: 86.54%
edo@edo-Inspiron-N5010:~/pyProjects$ █
```

Fig. 5: Feature combination for data set 1

Accuracy=86.54%

b. Data set 2 Caltech (Buddha & lamp, 60 each) classifier- kNN

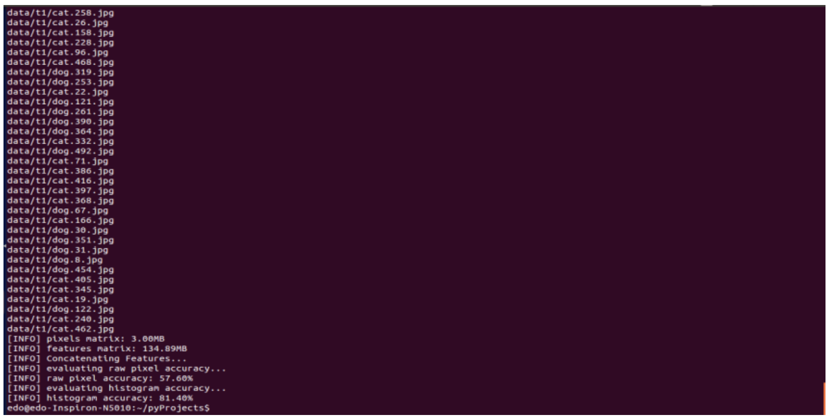


```
data/final/caltech3/buddha.0021.jpg
data/final/caltech3/lamp.0006.jpg
data/final/caltech3/buddha.0026.jpg
data/final/caltech3/buddha.0038.jpg
data/final/caltech3/buddha.0030.jpg
data/final/caltech3/lamp.0018.jpg
data/final/caltech3/lamp.0037.jpg
data/final/caltech3/buddha.0040.jpg
data/final/caltech3/buddha.0049.jpg
data/final/caltech3/buddha.0037.jpg
data/final/caltech3/lamp.0059.jpg
data/final/caltech3/buddha.0035.jpg
data/final/caltech3/buddha.0045.jpg
data/final/caltech3/buddha.0060.jpg
data/final/caltech3/lamp.0045.jpg
data/final/caltech3/lamp.0021.jpg
data/final/caltech3/buddha.0041.jpg
data/final/caltech3/buddha.0028.jpg
data/final/caltech3/lamp.0055.jpg
data/final/caltech3/buddha.0033.jpg
data/final/caltech3/buddha.0051.jpg
data/final/caltech3/lamp.0032.jpg
data/final/caltech3/lamp.0020.jpg
data/final/caltech3/lamp.0024.jpg
data/final/caltech3/buddha.0013.jpg
data/final/caltech3/lamp.0041.jpg
data/final/caltech3/buddha.0020.jpg
data/final/caltech3/lamp.0047.jpg
data/final/caltech3/lamp.0019.jpg
data/final/caltech3/lamp.0011.jpg
data/final/caltech3/lamp.0002.jpg
data/final/caltech3/lamp.0004.jpg
data/final/caltech3/lamp.0053.jpg
[INFO] pixels matrix: 0.36MB
[INFO] features matrix: 16.19MB
[INFO] Concatenating Features...
[INFO] evaluating raw pixel accuracy...
[INFO] raw pixel accuracy: 66.67%
[INFO] evaluating histogram accuracy...
[INFO] histogram accuracy: 95.00%
edo@edo-Inspiron-N5010:~/pyProjects$ █
```

Fig. 6: Feature combination for data set 2

Accuracy= 95.00 %

c. Data set 3 Pet animals(cat & dog,500 each) classifier-kNN



```
data/t1/cat.258.jpg
data/t1/cat.26.jpg
data/t1/cat.158.jpg
data/t1/cat.228.jpg
data/t1/cat.96.jpg
data/t1/cat.468.jpg
data/t1/dog.319.jpg
data/t1/dog.253.jpg
data/t1/cat.22.jpg
data/t1/dog.121.jpg
data/t1/dog.261.jpg
data/t1/dog.390.jpg
data/t1/dog.364.jpg
data/t1/cat.332.jpg
data/t1/dog.492.jpg
data/t1/cat.71.jpg
data/t1/cat.386.jpg
data/t1/cat.416.jpg
data/t1/cat.397.jpg
data/t1/cat.368.jpg
data/t1/dog.67.jpg
data/t1/cat.166.jpg
data/t1/dog.30.jpg
data/t1/dog.351.jpg
data/t1/dog.31.jpg
data/t1/dog.8.jpg
data/t1/dog.454.jpg
data/t1/cat.405.jpg
data/t1/cat.345.jpg
data/t1/cat.19.jpg
data/t1/dog.122.jpg
data/t1/cat.240.jpg
data/t1/cat.462.jpg
[INFO] pixels matrix: 3.00MB
[INFO] features matrix: 134.89MB
[INFO] Concatenating Features...
[INFO] evaluating raw pixel accuracy...
[INFO] raw pixel accuracy: 57.60%
[INFO] evaluating histogram accuracy...
[INFO] histogram accuracy: 81.40%
edo@edo-Inspiron-N5010:~/pyProjects$
```

Fig. 7: Feature combination for data set 3

Accuracy=81.40 %

The results for feature combination are obtained by adding another descriptor HOG with the existing one.

Single feature vs. feature combination graph

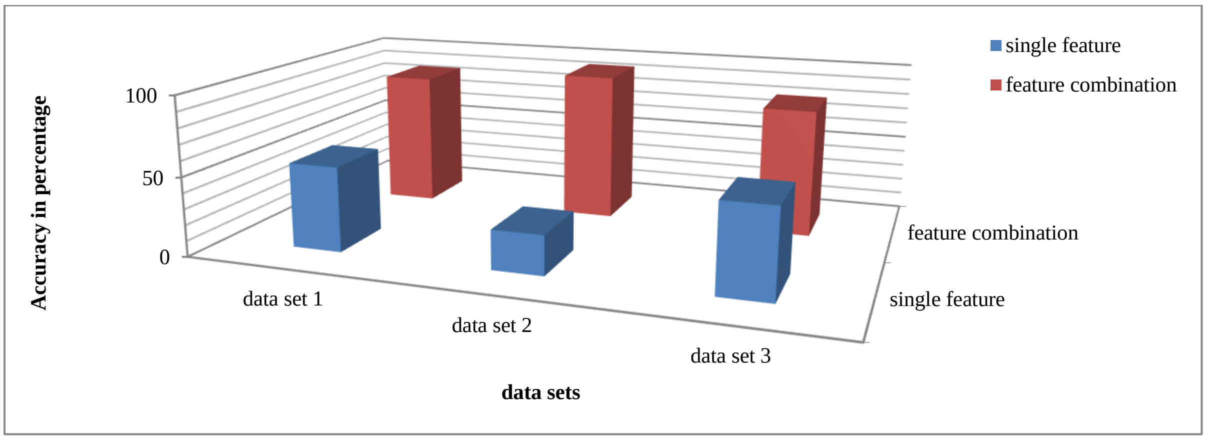
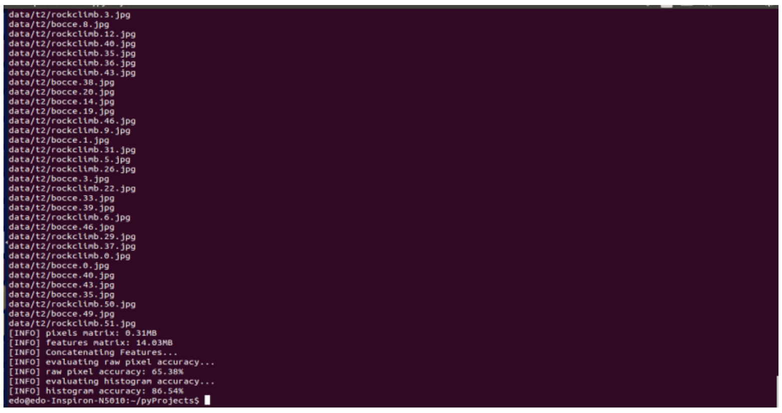


Fig. 8 feature combination graph

From the analysis of the above graph, we can clearly say that combination of HOG features to the existing features obtained from color histogram, yields remarkable results. Thus proving the effectiveness (in terms of classification accuracy) of the feature combination strategy. Due to the limited capacity of the simulation system, we have only illustrated the basic level of combining two descriptors, else more number of features (like SIFT, SURF etc) can be added further to observe the effect of feature combination on classification accuracy. One more interesting thing to note is that the as HOG features were added, the pixel matrix size remains constant but the size of the feature matrix increases in huge amount, thus showing how new features affect the computation.

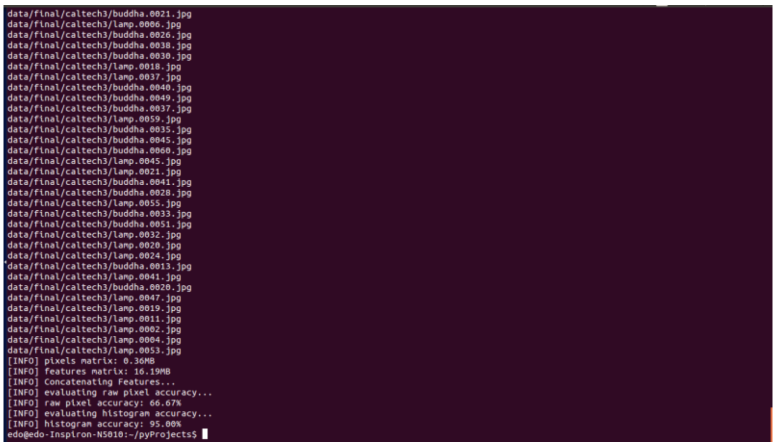
After having evaluated the Feature Combination, we now change certain aspects of processing to see the change in recognition metrics. Now we study the effect of change of parameters on recognition rate. For this we have taken three aspects. First is changing the classifiers (SVM and kNN alternatively) Second is changing the training testing split ratio (taking testing 25%, 30% and 50%) Third is changing the number of images (image size) as 400, 1000, 2000. Although we can consider more aspects, but inclusion of the above three are intuitively relevant to our work. The comparisons are as follows:



```
data/t2/rockclimb.3.jpg
data/t2/bocce.8.jpg
data/t2/rockclimb.12.jpg
data/t2/rockclimb.40.jpg
data/t2/rockclimb.35.jpg
data/t2/rockclimb.36.jpg
data/t2/rockclimb.43.jpg
data/t2/bocce.38.jpg
data/t2/bocce.20.jpg
data/t2/bocce.14.jpg
data/t2/bocce.19.jpg
data/t2/rockclimb.46.jpg
data/t2/rockclimb.9.jpg
data/t2/bocce.1.jpg
data/t2/rockclimb.31.jpg
data/t2/rockclimb.5.jpg
data/t2/rockclimb.26.jpg
data/t2/bocce.3.jpg
data/t2/rockclimb.22.jpg
data/t2/bocce.33.jpg
data/t2/bocce.39.jpg
data/t2/rockclimb.6.jpg
data/t2/bocce.46.jpg
data/t2/rockclimb.29.jpg
data/t2/rockclimb.37.jpg
data/t2/rockclimb.0.jpg
data/t2/bocce.0.jpg
data/t2/bocce.40.jpg
data/t2/bocce.43.jpg
data/t2/bocce.35.jpg
data/t2/rockclimb.50.jpg
data/t2/bocce.49.jpg
data/t2/rockclimb.51.jpg
[INFO] pixels matrix: 0.31MB
[INFO] features matrix: 14.03MB
[INFO] Concatenating Features...
[INFO] evaluating raw pixel accuracy...
[INFO] raw pixel accuracy: 65.38%
[INFO] evaluating histogram accuracy...
[INFO] histogram accuracy: 86.54%
edo@edo-Inspiron-N5010:~/pyProjects$ 
```

Fig. 9: kNN with data set 1

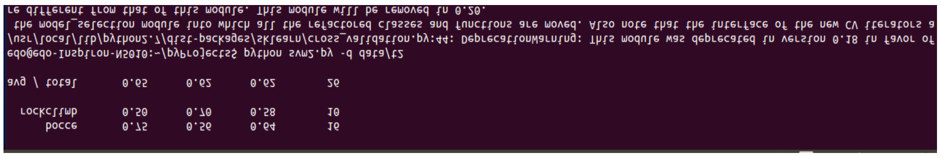
Accuracy=86.54 %



```
data/final/caltech3/buddha.0021.jpg
data/final/caltech3/lamp.0006.jpg
data/final/caltech3/buddha.0026.jpg
data/final/caltech3/buddha.0038.jpg
data/final/caltech3/buddha.0030.jpg
data/final/caltech3/lamp.0018.jpg
data/final/caltech3/lamp.0037.jpg
data/final/caltech3/buddha.0040.jpg
data/final/caltech3/buddha.0049.jpg
data/final/caltech3/buddha.0037.jpg
data/final/caltech3/lamp.0059.jpg
data/final/caltech3/buddha.0035.jpg
data/final/caltech3/buddha.0045.jpg
data/final/caltech3/buddha.0060.jpg
data/final/caltech3/lamp.0045.jpg
data/final/caltech3/lamp.0021.jpg
data/final/caltech3/buddha.0041.jpg
data/final/caltech3/buddha.0028.jpg
data/final/caltech3/lamp.0055.jpg
data/final/caltech3/buddha.0033.jpg
data/final/caltech3/buddha.0051.jpg
data/final/caltech3/lamp.0032.jpg
data/final/caltech3/lamp.0020.jpg
data/final/caltech3/lamp.0024.jpg
data/final/caltech3/buddha.0013.jpg
data/final/caltech3/lamp.0041.jpg
data/final/caltech3/buddha.0020.jpg
data/final/caltech3/lamp.0047.jpg
data/final/caltech3/lamp.0019.jpg
data/final/caltech3/lamp.0011.jpg
data/final/caltech3/lamp.0002.jpg
data/final/caltech3/lamp.0004.jpg
data/final/caltech3/lamp.0053.jpg
[INFO] pixels matrix: 0.36MB
[INFO] features matrix: 16.19MB
[INFO] Concatenating Features...
[INFO] evaluating raw pixel accuracy...
[INFO] raw pixel accuracy: 66.67%
[INFO] evaluating histogram accuracy...
[INFO] histogram accuracy: 95.00%
edo@edo-Inspiron-N5010:~/pyProjects$ 
```

Fig. 10: kNN with data set 2

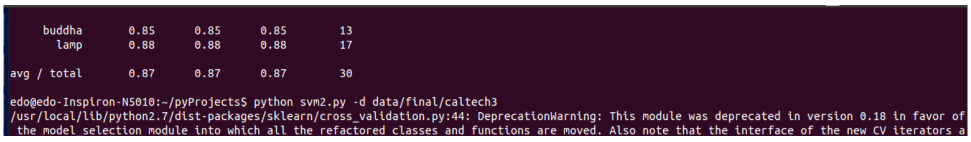
Accuracy= 95.00 %



```
      bocce       0.75      0.56      0.64        16
  rockclimb       0.50      0.70      0.58        10

avg / total       0.65      0.62      0.62        26

edo@edo-Inspiron-N5010:~/pyProjects$ python svm2.py -d data/f2
/usr/local/lib/python2.7/dist-packages/sklearn/cross_validation.py:44: DeprecationWarning: This module was deprecated in version 0.18 in favor of
 the model_selection module into which all the refactored classes and functions are moved. Also note that the interface of the new CV iterators a
re different from that of this module. This module will be removed in 0.20.
```

Fig. 11: SVM with data set 1

Accuracy=65.00 %



```
     buddha       0.85      0.85      0.85        13
       lamp       0.88      0.88      0.88        17

avg / total       0.87      0.87      0.87        30

edo@edo-Inspiron-N5010:~/pyProjects$ python svm2.py -d data/final/caltech3
/usr/local/lib/python2.7/dist-packages/sklearn/cross_validation.py:44: DeprecationWarning: This module was deprecated in version 0.18 in favor of
 the model_selection module into which all the refactored classes and functions are moved. Also note that the interface of the new CV iterators a
```

Fig. 12: SVM with data set 2

Accuracy=60%

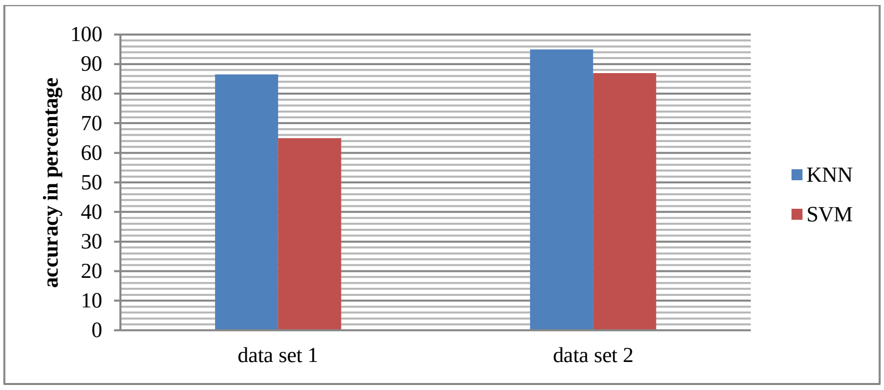


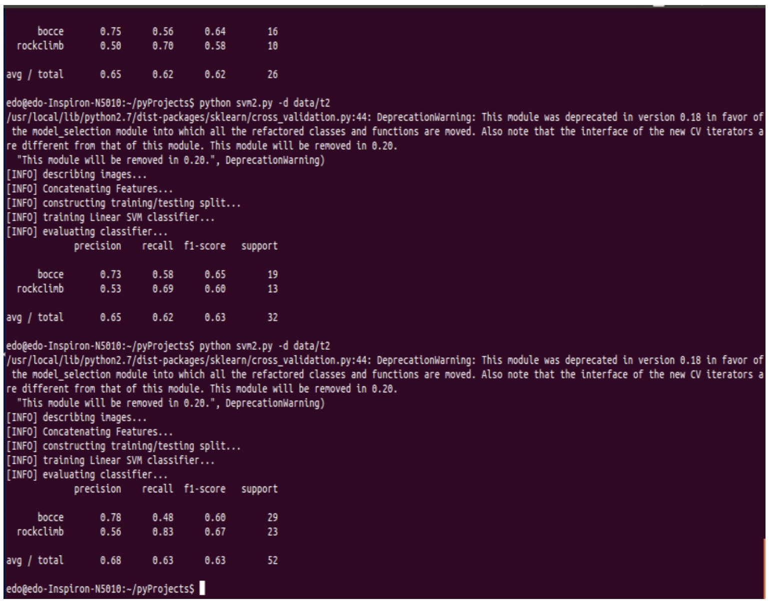
Fig. 13: KNN versus SVM

The above graph above depicts that kNN gives better classification accuracy than SVM

Change in training-testing split ratio.

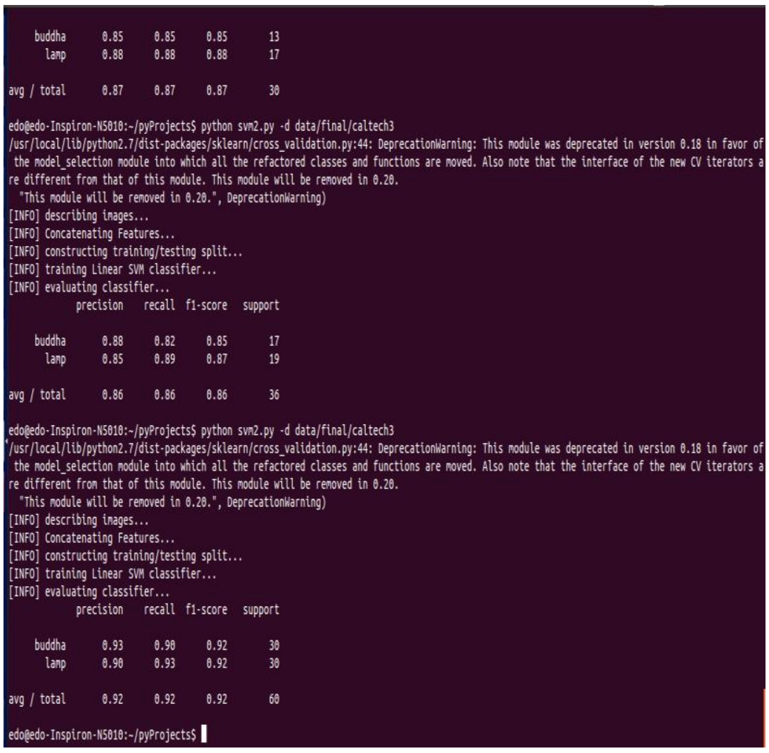
- Here we observe how the recognition metrics change with the change in training and testing ratio.
- We have experimented on two data sets that are data set 1 and data set 2 using SVM classifier. For each data set, we have taken 3 training testing split ratio that are a.3:1, 7:3 and 1:1

A. dataset 1



```
      bocce       0.75      0.56      0.64        16
  rockclimb       0.50      0.70      0.58        10

avg / total       0.65      0.62      0.62        26

edo@edo-Inspiron-N5010:~/pyProjects$ python svm2.py -d data/t2
/usr/local/lib/python2.7/dist-packages/sklearn/cross_validation.py:44: DeprecationWarning: This module was deprecated in version 0.18 in favor of
 the model_selection module into which all the refactored classes and functions are moved. Also note that the interface of the new CV iterators a
re different from that of this module. This module will be removed in 0.20.
  "This module will be removed in 0.20.", DeprecationWarning)
[INFO] describing images...
[INFO] Concatenating Features...
[INFO] constructing training/testing split...
[INFO] training Linear SVM classifier...
[INFO] evaluating classifier...
             precision    recall  f1-score   support

      bocce       0.73      0.58      0.65        19
  rockclimb       0.53      0.69      0.60        13

avg / total       0.65      0.62      0.63        32

edo@edo-Inspiron-N5010:~/pyProjects$ python svm2.py -d data/t2
/usr/local/lib/python2.7/dist-packages/sklearn/cross_validation.py:44: DeprecationWarning: This module was deprecated in version 0.18 in favor of
 the model_selection module into which all the refactored classes and functions are moved. Also note that the interface of the new CV iterators a
re different from that of this module. This module will be removed in 0.20.
  "This module will be removed in 0.20.", DeprecationWarning)
[INFO] describing images...
[INFO] Concatenating Features...
[INFO] constructing training/testing split...
[INFO] training Linear SVM classifier...
[INFO] evaluating classifier...
             precision    recall  f1-score   support

      bocce       0.78      0.48      0.60        29
  rockclimb       0.56      0.83      0.67        23

avg / total       0.68      0.63      0.63        52

edo@edo-Inspiron-N5010:~/pyProjects$ 
```

Fig. 14: training testing ratio change for data set 1



```
     buddha       0.85      0.85      0.85        13
       lamp       0.88      0.88      0.88        17

avg / total       0.87      0.87      0.87        30

edo@edo-Inspiron-N5010:~/pyProjects$ python svm2.py -d data/final/caltech3
/usr/local/lib/python2.7/dist-packages/sklearn/cross_validation.py:44: DeprecationWarning: This module was deprecated in version 0.18 in favor of
 the model_selection module into which all the refactored classes and functions are moved. Also note that the interface of the new CV iterators a
re different from that of this module. This module will be removed in 0.20.
  "This module will be removed in 0.20.", DeprecationWarning)
[INFO] describing images...
[INFO] Concatenating Features...
[INFO] constructing training/testing split...
[INFO] training Linear SVM classifier...
[INFO] evaluating classifier...
             precision    recall  f1-score   support

     buddha       0.88      0.82      0.85        17
       lamp       0.85      0.89      0.87        19

avg / total       0.86      0.86      0.86        36

edo@edo-Inspiron-N5010:~/pyProjects$ python svm2.py -d data/final/caltech3
/usr/local/lib/python2.7/dist-packages/sklearn/cross_validation.py:44: DeprecationWarning: This module was deprecated in version 0.18 in favor of
 the model_selection module into which all the refactored classes and functions are moved. Also note that the interface of the new CV iterators a
re different from that of this module. This module will be removed in 0.20.
  "This module will be removed in 0.20.", DeprecationWarning)
[INFO] describing images...
[INFO] Concatenating Features...
[INFO] constructing training/testing split...
[INFO] training Linear SVM classifier...
[INFO] evaluating classifier...
             precision    recall  f1-score   support

     buddha       0.93      0.90      0.92        30
       lamp       0.90      0.93      0.92        30

avg / total       0.92      0.92      0.92        60

edo@edo-Inspiron-N5010:~/pyProjects$ 
```

Fig. 15: training testing ratio change for data set 2

We have sketched the graph between datasets and precision by changing the training-testing split ratio.

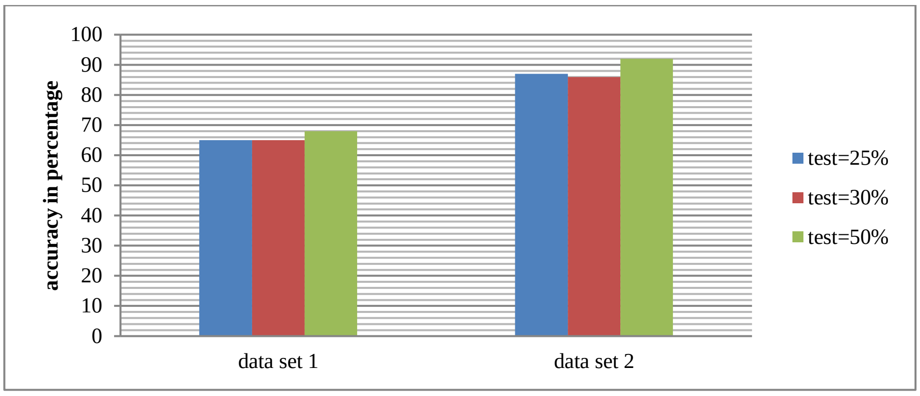


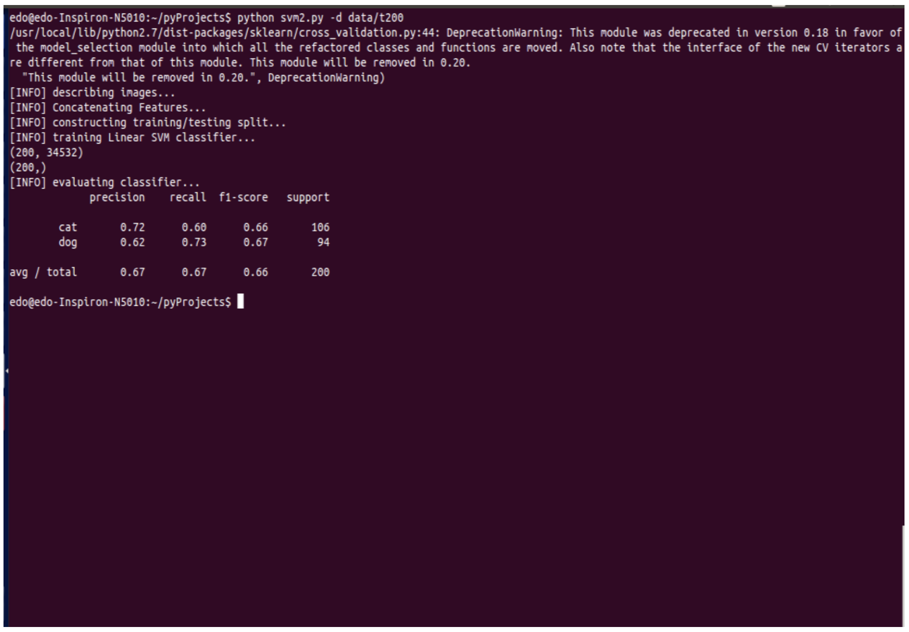
Fig.16: training testing ratio change graph

The data set 1 obtains constant accuracy at first, and then the accuracy rises. In data set 2 the accuracy rises then drops and again rises

3. Change in input size

- Here we analyze how the recognition metrics, especially how the precision (accuracy) rate changes with change in the number of input images.
- We have experimented this using the SVM classifier.
- We have checked this using data set 3 with varying the number of input images as follows

a. input size = 400 images



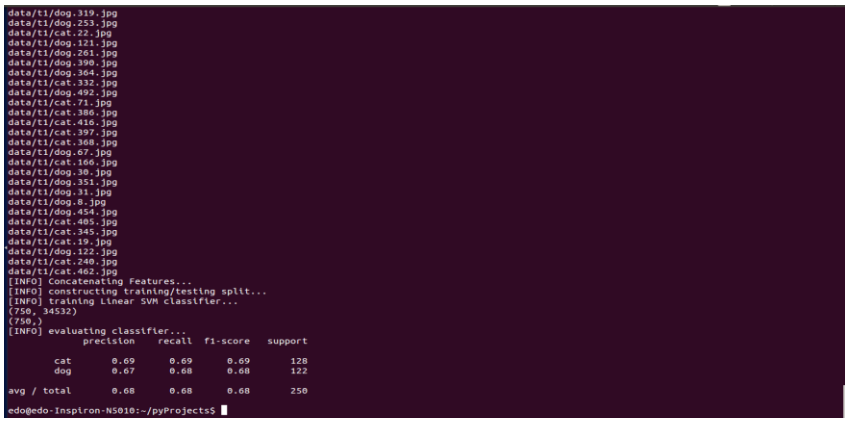
```
edo@edo-Inspiron-N5010:~/pyProjects$ python svm2.py -d data/t200
/usr/local/lib/python2.7/dist-packages/sklearn/cross_validation.py:44: DeprecationWarning: This module was deprecated in version 0.18 in favor of
 the model_selection module into which all the refactored classes and functions are moved. Also note that the interface of the new CV iterators a
re different from that of this module. This module will be removed in 0.20.
  "This module will be removed in 0.20.", DeprecationWarning)
[INFO] describing images...
[INFO] Concatenating Features...
[INFO] constructing training/testing split...
[INFO] training Linear SVM classifier...
(200, 34532)
(200,)
[INFO] evaluating classifier...
             precision    recall  f1-score   support

        cat       0.72      0.60      0.66       106
        dog       0.62      0.73      0.67        94

avg / total       0.67      0.67      0.66       200

edo@edo-Inspiron-N5010:~/pyProjects$ █
```

Fig.17: SVM for 400 images of data set 3

Accuracy =67.00 %

Processing time =44 seconds

b. input size = 1000 images



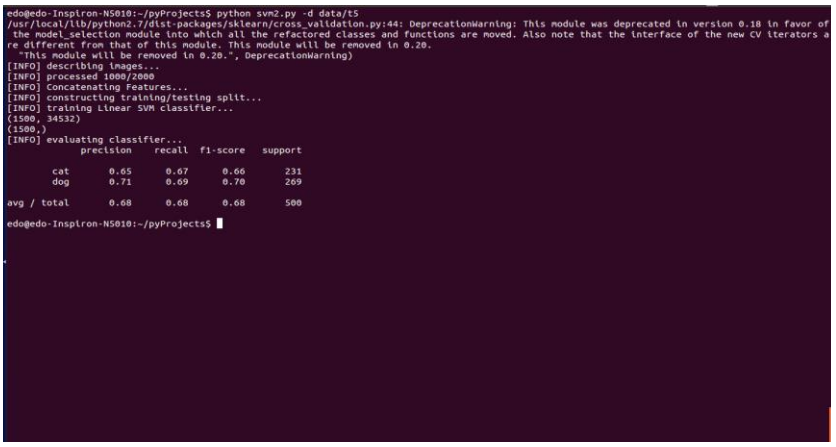
```
data/t1/dog.319.jpg
data/t1/dog.253.jpg
data/t1/cat.22.jpg
data/t1/dog.121.jpg
data/t1/dog.261.jpg
data/t1/dog.390.jpg
data/t1/dog.364.jpg
data/t1/cat.332.jpg
data/t1/dog.492.jpg
data/t1/cat.71.jpg
data/t1/cat.386.jpg
data/t1/cat.416.jpg
data/t1/cat.397.jpg
data/t1/cat.368.jpg
data/t1/dog.67.jpg
data/t1/cat.166.jpg
data/t1/dog.30.jpg
data/t1/dog.351.jpg
data/t1/dog.31.jpg
data/t1/dog.8.jpg
data/t1/dog.454.jpg
data/t1/cat.405.jpg
data/t1/cat.345.jpg
data/t1/cat.19.jpg
data/t1/dog.122.jpg
data/t1/cat.240.jpg
data/t1/cat.462.jpg
[INFO] Concatenating Features...
[INFO] constructing training/testing split...
[INFO] training Linear SVM classifier...
(750, 34532)
(750,)
[INFO] evaluating classifier...
             precision    recall  f1-score   support

        cat       0.69      0.69      0.69       128
        dog       0.67      0.68      0.68       122

avg / total       0.68      0.68      0.68       250

edo@edo-Inspiron-N5010:~/pyProjects$ █
```

Fig. 18: SVM for 1000 images of data set 3

Accuracy= 68.00 %

Processing time =146 seconds

c. input size = 2000 images



```
edo@edo-Inspiron-N5010:~/pyProjects$ python svm2.py -d data/t5
/usr/local/lib/python2.7/dist-packages/sklearn/cross_validation.py:44: DeprecationWarning: This module was deprecated in version 0.18 in favor of
 the model_selection module into which all the refactored classes and functions are moved. Also note that the interface of the new CV iterators a
re different from that of this module. This module will be removed in 0.20.
  "This module will be removed in 0.20.", DeprecationWarning)
[INFO] describing images...
[INFO] processed 1000/2000
[INFO] Concatenating Features...
[INFO] constructing training/testing split...
[INFO] training Linear SVM classifier...
(1500, 34532)
(1500,)
[INFO] evaluating classifier...
             precision    recall  f1-score   support

        cat       0.65      0.67      0.66       231
        dog       0.71      0.69      0.70       269

avg / total       0.68      0.68      0.68       500

edo@edo-Inspiron-N5010:~/pyProjects$ █
```

Fig. 19: SVM for 2000 images of data set 3

Accuracy=68.00 % Processing time=153 seconds

Here we draw graph taking into account data of various size and observing its impact upon two metrics. They are a. accuracy/precision and b. processing time

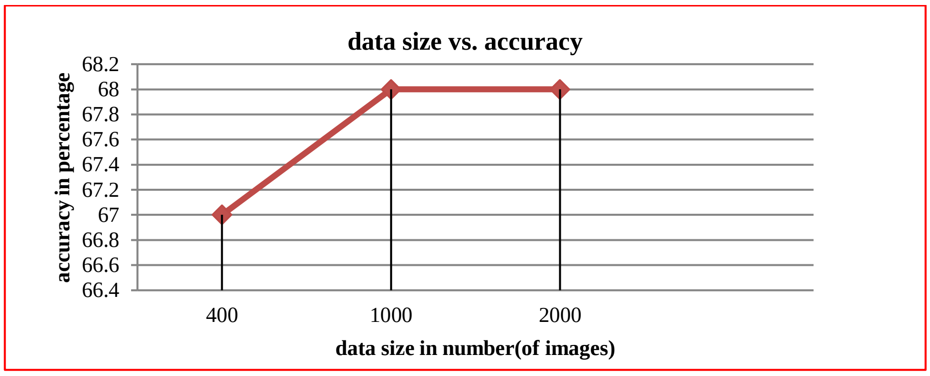


Fig. 20: data size versus accuracy

The accuracy rate increase at first but as the input number of images increases, it attempts a constant value.

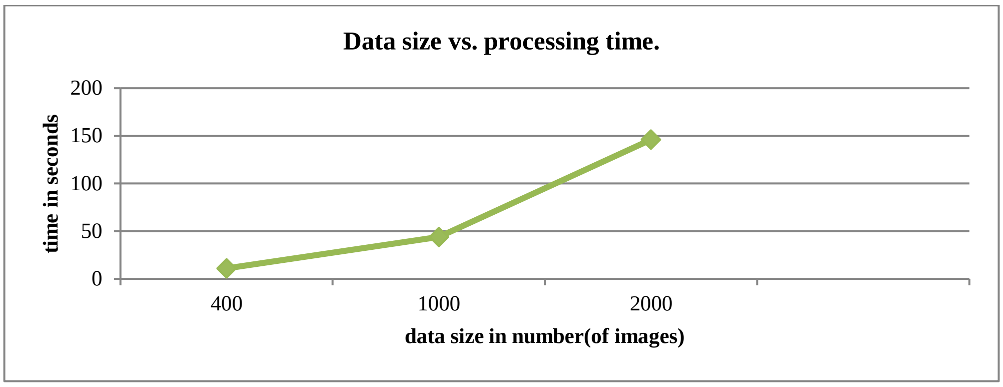


Fig. 21: data size versus processing time

As the number of images increases, the processing time increases. This may be because of the expansion of feature matrix that consumes processing time.

5. Conclusion & Future Work

So we can conclude from the above results that feature combination is quite efficient in practical approach. The addition of new features to the existing ones gradually enhance the classification accuracy. Thus such systems can be a great help from the realistic point of view and can be utilized in day to day life. The recognition metrics depend differently on different number of aspects, thus if we can manage such parameters wisely, we can achieve better results by extending the application of feature combination. A lot of advanced works have been done on feature combination like introducing the Dsets clusters, other clustering algorithms, still a lot more can be done. The dimension of the feature matrix is a major challenge that increases with addition of features thus increasing the processing complexity. Advanced works can be done to minimize or reduce the dimensions of the feature matrix so as to ease the processing. The different algorithms can be modified well, optimized for better performance. More robust feature descriptor algorithms can be developed for enhancing the accuracy of the classification.

Refrences

[1] Xu Cheng Yin, Chang –Ping Liu and Zhi Han, “Feature combination using boosting” X.-C. Yin et al. / Pattern Recognition Letters 26 (2005) 2195–2205
[2] P. Gehler and S. Nowozin, “On feature combination for multiclass object classification,” in *Proc. IEEE 12th Conf. Comput. Vis.*, Kyoto, Japan, Sep./Oct. 2009, pp. 221–228.
[3] Linbo Zhang, Baihua Xiao, Chuncheng Wang, Gang Cheng and Yunxue Shao, “An efficient strategy for features combination” 2010 3rd International Congress on Image and Signal Processing (CISP2010)
[4] J. Hou, B.-P. Zhang, N.-M. Qi, and Y. Yang, “Evaluating feature combination in object classification,” in *Proc. Int. Symp. Vis. Comput.*, Las Vega, NV, USA, Sep. 2011, pp. 597–606.
[5] Jian Hou, Wei-Xue Liu and Hamid Reza Karimi, “Exploring the best classification from average feature combination” Volume 2014, Article ID 602763, 7 pages
[6] G. R. G. Lanckriet, N. Cristianini, P. Bartlett, L. El Ghaoui, and M. I. Jordan, “Learning the kernel matrix with semidefinite programming,” *J. Mach. Learn. Res.*, vol. 5, pp. 27–72, Jan. 2004.
[7] A. Kumar and C. Sminchisescu, “Support kernel machines for objectrecognition,” in *Proc. IEEE 11th Int. Conf. Comput. Vis.*, Rio de Janeiro,Brazil, Oct. 2007, pp. 1–8.
[8] Y.-Y. Lin, T.-L. Liu, and C.-S. Fuh, “Local ensemble kernel learning forobject category recognition,” in *Proc. IEEE Conf. Comput. Vis. PatternRecognit.*, Minneapolis, MN, USA, Jun. 2007, pp. 1–8
[9] M. Varma and D. Ray, “Learning the discriminative power-invariance trade-off,” in *Proc. IEEE 11th Int. Conf. Comput. Vis.*, Rio de Janeiro,Brazil, Oct. 2007, pp. 1–8.
[10] J. Yang, Y. Li, Y. Tian, L. Duan, and W. Gao, “Group-sensitive multiple kernel learning for object categorization,” in *Proc. IEEE 12th Int. Conf.Comput. Vis.*, Kyoto, Japan, Sep./Oct. 2009, pp. 436–443.
[11] C.Cortes, M.Mohri, A.Rostamizadeh, “Learning non-linear combinations of kernels”, in: Advances in Neural Information Processing Systems, 2009, pp.396–404.
[12] M.Varma, and B.R.Babu, “More generality in efficient multiple kernel learning”, in: International Conference on Machine Learning , 2009, pp.1065–1072.

[13] S.V.N.Vishwanathan, Z.Sun, N.T.Ampornpunt, M.Varma, "Multiple kernel learning and the SMO algorithm",in: Advances in Neural Information Processing Systems, 2010, pp.2361–2369.
[14] F.R.Bach, "Exploring large feature spaces with hierarchical multiple kernel learning", in: Advances in Neural Information Processing Systems, 2008, pp. 105–112.
[15] Wei Luo, Jian Yang, Wei Xu, Jun Li and Jian Zhang, "Higher –level feature combination via multiple kernel learning for image classification"
[16] Jian-hua Yeh, Chen Lin, Yuan-ling Chang, 'Classification improvement based on feature combination and topic vector model." 2012 International Conference on Systems and Informatics (ICSAI 2012)
[17] Li Zhou, Zongtan Zhou and Dewen Hu, "Scene classification using multi resolution low level feature combination" L. Zhouetal./Neurocomputing122(2013)284–297
[18] S. Lazebnik, C. Schmid, and J. Ponce, "Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, New York, NY, USA, Jun. 2006, pp. 2169–2178.
[19] A. Oliva and A. Torralba, "Modeling the shape of the scene: A holistic representation of the spatial envelope," *Int. J. Comput. Vis.*, vol. 42, no. 3, pp. 145–175, 2001.
[20] J. Wu and J. M. Rehg, "Beyond the Euclidean distance: Creating effective visual codebooks using the histogram intersection kernel," in *Proc. IEEE 12th Int. Conf. Comput. Vis.*, Kyoto, Japan, Sep./Oct. 2009, pp. 630–637
[21] J. Hou and M. Pelillo, "A simple feature combination method based on dominant sets," *Pattern Recognit.*, vol. 46, no. 11, pp. 3129–3139,2013.
[22] M. Pavan and M. Pelillo, "Dominant sets and pairwise clustering," *IEEETrans. Pattern Anal. Mach. Intell.*, vol. 29, no. 1, pp. 167–172, Jan. 2007.
[23] Jian Hou, Huijun Gao, Qi Xia and Naiming Qi, "Feature combination and the kNN Framework in object classification" ieee transactions on neural networks and learning systems, Vol.27, No.6,June 2016
[24] Jian Hou, Huijun Gao and Xuelong li, "Feature combination via clustering" " ieee transactions on neural networks and learning systems, 2017.
[25] S. R. Bulò, M. Pelillo, and I. M. Bomze, "Graph-based quadratic optimization: A fast evolutionary approach," *Comput. Vis. Image Understand.*, vol. 115, no. 7, pp. 984–995, 2011.
[26] Lin Yuan, Fanglin Chen, Li Zhou and Dewen Hu, "Improve scene classification by using feature and kernel combination.
[27] Mehrdad Ahmadi Soofivand, Abdollah Amirkhani, Mohammad Reza Daliri, Gholamali Rezaeirad "Feature level combination for object classification" 2014 4th conference on International Conference on Computer and Knowledge Engineering.
[28] Taiwan Huang, Lei Li, Yazhao Zhang, Junqi Chi "Summarization based on multiple feature combination"
[29] Aibo Song, Wei Qian, Zhiang Wu, Jinghua Zhao "Discriminative Feature combination Selection For Enhancing Multiclass Classification." 2015 International Conference on Behavioral, Economic and Socio-Cultural Computing.
[30] D. G. Lowe, "Distinctive image features from scale-invariant keypoints,"*Int. J. Comput. Vis.*, vol. 60, no. 2, pp. 91–110, 2004.
[31] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, San Diego, CA, USA, Jun. 2005, pp. 886–893.
[32] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool, "SURF: Speeded up robust features," *Comput. Vis. Image Understand.*, vol. 110, no. 3, pp. 346–359, 2008
[33] T. Ojala, M. Pietikainen, and T. Maenpaa, "Multiresolution gray-scale and rotation invariant texture classification with local binary patterns," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 7, pp. 971–987, Jul. 2002
[34] L.-J. Li and L. Fei-Fei, "What, where and who? Classifying events by scene and object recognition," in *Proc. IEEE 11th Int. Conf. Comput.Vis.*, Rio de Janeiro, Brazil, Oct. 2007, pp. 1–8.
[35] Fei-Fei, L., Fergus, R., Perona, P.: Learning generative visual models from few training examples: an incremental bayesian approach tested on 101 object categories.In: CVPR, Workshop on Generative-Model Based Vision, p. 178 (2004)
[36] O. M. Parkhi, A. Vedaldi, A. Zisserman, C. V. Jawahar, "cats and dogs" IEEE conference on computer vision and pattern recognition, 2012.