Performance Analysis of Baseline Feature Combination Methods in Object Classification

Amitav Mahapatra

Research Scholar, *Biju Patnaik University of Technology, Rourkela, Odisha, India*

Prashanta Kumar Patra

Dean, SRIC, SOA University, Bhubaneswar, Odisha, India

Abstract:- This paper aims to assess the strategy of feature combination and demonstrate its effectiveness compared to using single features in terms of classification accuracy. Based on the results, it is evident that combining features by using different baseline feature combination methods is highly effective in a practical context. With the gradual addition of new features, the classification accuracy is enhanced. We assess the classification accuracy of various baseline feature combination methods using five distinct datasets that contain a variety of object types and classifiers. In comparison to other baseline feature combination methods, we have observed that weighted average methods perform exceptionally well.

Keywords:Feature Combination, Classification Accuracy

1. Introduction

The burgeoning field of computer vision offers tremendous research and innovation opportunities. Even with simple image collections, object categorization remains difficult after decades of effort. Intra-class variability and inter-class correlation cause large class differences and similarities, which makes this difficult.

To address these issues, SIFT [30], HOG [31], and SURF [32] have been developed. They excel in some object classes but cannot handle all. Multiple complementing characteristics must be fused to address this. A final feature with greater classification accuracy than any single characteristic is created by combining their strengths.

The structure of the paper is as follows. Section 2 provides a concise overview of significant research advancements in feature combination and demonstrates how these advancements serve as inspiration for our work in this study. Section 3 presents a range of baseline approaches for combining features, along with their respective algorithms. These methods are employed to assess the accuracy. In Section 4, we provide a detailed explanation of the procedure for calculating the accuracy using different combinations of baseline features.Various classifier used are explained in section 5. The experimental results and analysis are presented in Section 6. Section 7 serves as the concluding section and future workof the paper.

2. Related Works

While feature combination work has been used to several fields such as text classification, disease diagnosis, and human face identification, we will focus specifically on its application in 'Object classification'. We provide a comprehensive presentation of the literary works conducted in the past, including the methodologies used, comparisons made, advantages and disadvantages of certain approaches employed, and ultimately a summary. The literature works are as follows: An innovative approach to combining features using boosting is introduced [1]. Boosting is a technique that combines features in this context. Unlike conventional boosting methods, variation boosting trains weak classifiers using different sets of features. Weighted voting is a method that combines classifiers. Provide the classifier's output for that particular round. Studies demonstrate that this technique has the ability to integrate feature selection, communication, and classifier learning. In

1990, Schapire established the concept of boosting as a means to enhance the performance of weak learning algorithms. AdaBoost enhances boosting and voting classifiers by incorporating two more classes. In standard AdaBoost, weak classifier components are assigned a comparable feature vector. Despite the varying types of features, every training instance consists of a feature vector of fixed length, with similar qualities arranged in a predetermined order. As demonstrated in reference [1], AdaBoost exhibits superior performance. This boosting technique involves training a strong classifier by combining many weak classifiers on samples of each feature vector, utilising system-coded attributes at each iteration. The algorithm iterates through each round, employing weighted voting to identify combinations. AdaBoost and other boosting techniques, such as decision trees and neural networks, can utilise weak learning algorithms. This variant of boosting can be modified for multi-class scenarios, similar to generic boosting. The method proposed in [1] exhibits a specific classification approach, in contrast to AdaBoost, which employs a more general classification strategy. The variant of boosting shown significantly superior classification performance compared to conventional methods across three datasets. Enhancing performance can be achieved through boosting, optimising feature extraction, and adjusting neural network parameters.

Additional research conducted in [2] has validated the reliability and resilience of boosting by applying it to various multiclass scenarios. The paper [2] provides a comprehensive explanation of vivid kernel approaches for feature combination, including baselines, multiple kernel learning (MKL), and boosting. The study presented formulations that were derived from LPBoost. Specifically, two strategies have been suggested that draw inspiration from the MKL decision function. The specific datasets have been used to experiment with LPBoost and its multiclass versions, namely LP-β and LP-B. In the MKL solution, the combining coefficients are considered to represent the influence of features on a class. However, this assumption is incorrect in a multiclass scenario. Therefore, in multiclass decision-making, all features are given equal importance. LP-β selects a subset of three features from a total of seven, whereas other methods strive to choose all features. The results of the Oxford flower experiment indicate that the MKL and LP-β learning methods are resistant to irrelevant traits, but the CG-Boosting strategy deteriorates over time.

In order to avoid tedious and time-consuming learning methods in MKL, the researchers included existing knowledge into the process of kernel mixing [3]. The weight assigned to each attribute in combination is determined by its effectiveness in classifying the class. Therefore, [3] introduces several techniques that integrate local feature weights, as well as a novel approach for determining feature weights based on classification results. The problem of feature combination is addressed by utilising bag-of-words histogram features and kernel-based classifiers. Various literature surveys reveal the following research gap:

Limited research of Feature Fusion strategies:

Despite advancements in object categorization using feature extraction methods, there is a gap in comprehensive research and comparison of feature fusion strategies. Numerous studies focus on individual feature extraction approaches rather than integrating features for classification improvement.

Limited Research on Domain-Specific Feature Combination:

Image datasets include medical, satellite, and natural scene images. Domain-specific feature combination techniques customised to each domain's unique qualities and challenges are understudied.

So the primary objective of this study is to investigate and evaluate different feature combination strategies for object classification in image datasets, aiming to address the aforementioned research gap.

3. Baseline Feature Combination Methods And Algorithms

The process of integrating numerous types or sources of features derived from input data (such as images) to enhance the accuracy and robustness of object classification algorithms is referred to as feature combination in object classification.

1. Baselines Methods

In object classification, a baseline method of feature combination typically entails basic, easy-to-implement methods that offer a fundamental improvement over the use of individual features.

A) Concatenation Method

The concatenation method of feature combination entails the concatenation of feature vectors from various sources or types into a single, extended feature vector. This method maintains the individual representations of all original features without altering them. The algorithmic outline for the concatenation method of feature combination in the context of object classification is provided below.

Initialization:

- Let $n_i$denote the length of feature vector X_i for i=1,2,…,k
- Compute the total length of the concatenated feature vector: total_length=n_1+n_2+…+n_k
- Initialize concatenated_feature as an empty vector of size total_length

Concatenation Process:

- Initialize current_index to 0.
- For each feature vector X_i (where i ranges from 1 to k):
    - Concatenate X_i to concatenated_feature starting from current_index.
        - Copy elements of X_i to concatenated_feature starting from current_index up to current_index+n_i−1.
        - Update current_index=current_index+n_i

Output:

- concatenated_feature is the resulting feature vector obtained by concatenating all input feature vectors X_1,X_2,…,X_k

Ensure that all feature vectors have uniform lengths or handle variable lengths correctly in your implementation. The order of concatenation of X_1, X_2, ..., X_k may have an impact on the final feature vector. Verify that the order is in accordance with the intended approach for combining features The concatenation approach is applicable in preserving all original features clearly and is easily implementable. It is appropriate for situations where it is vital to preserve separate feature representations from several sources or kinds.

B) Product Method

The product method of feature combination entails the multiplication of corresponding features from various sources or varieties to generate a new combined feature vector. When attempting to document interactions or joint effects between features from various modalities, this approach may prove advantageous. The algorithmic framework for the product method of feature combination is provided below:

Algorithm for Product Method of Feature Combination

1. **Input:**
   - X_1, X_2,…,X_kFeature vectors from k different sources or types.
2. **Initialization:**

   Initialize an array combined_feature of size equal to the length of any feature vectorX_i

3. **Combination Process:**

   - For each feature index j from 1 to the length of X_i (assuming all X_i have the same length):
     - Initialize ***product*** to 1 (assuming a neutral starting point for multiplication).
     - For each feature vector X_i
       - Multiply ***product*** by X_i[j].

     Store the result in combined_feature[j]=***product***

3. **Output:**
   - combined_feature: The resulting feature vector obtained by multiplying corresponding features across all sources or types.

If any feature vector X_i has a value of zero at a certain position j, the resulting product combined_feature[j] will also be zero. Contemplate the manner in which to address such instances in accordance with the specific needs and specifications of the application. Normalisation is a process that involves scaling features to a similar range. This is done to ensure that each feature has equal weight before multiplication.

The product technique is applicable when there are interactions betweenfeature, especially when the features have a natural multiplicative relationship or where capturing joint effects is crucial.

C) Average Method

The average method of feature combination is a simple technique that involves combining features from various sources or types by calculating their average. Depending on the nature of the features and the task at hand, this method is particularly straightforward and can be effective. The following is a fundamental algorithmic framework for the implementation of the average technique of feature combination.

Algorithm for Average Method of Feature Combination

1. **Input:**

   X_1,X_2,…,X_kFeature vectors from k different sources or types.

2. **Initialization:**

   Initialize an array combined_feature of size equal to the length of any feature vector X_i.

3. **Combination Process:**

- For each feature index j from 1 to the length of X_i (assuming all X_i have the same length):
  - Initialize sum to 0.

- For each feature vector Xi:
    - Add Xi[j] to sum.
- Compute the average as average_feature[j]=sum/k

4. **Output**

- average_feature: The resulting feature vector obtained by averaging the corresponding features across all sources or types.

Make sure that all feature vectors have the same length or handle any differences in length appropriately. Normalisation is an important step to consider when dealing with features of different ranges. By scaling the features to a similar range, each feature is given equal weight, ensuring accurate averaging. The average method is straightforward yet powerful for merging features. It acknowledges the equal importance of each feature vector, but certain applications may necessitate more intricate weighting schemes.

D) Weighted Average Method

The weighted average approach of feature combination in object classification enables the incorporation of several feature sources or kinds with varying degrees of contribution to the final combined feature vector. This method is advantageous when specific characteristics are deemed more informative or dependable than others. Presented below is a systematic plan for executing the weighted average approach of combining features:

Algorithm for Weighted Average Method of Feature Combination

Steps:

1. **Initialization:**
    - Let n denote the length of any feature vector X_i.
    - Initialize weighted_average_feature as an empty vector of size n.
2. **Combination Process:**
    - For each feature index j from 1 to nnn (assuming all X_i have the same length):
        - Initialize weighted_sum to 0.
        - For each feature vector X_i:
        - Compute the weighted contribution: weighted_sum=weighted_sum+wi×Xi[j]
        - Compute the weighted average for the current index j: weighted_average_feature[j]=weighted_sum

3. Output:

- weighted_average_feature is the resulting feature vector obtained by weighted averaging the corresponding features across all sources or types.

Ensure that the weights $w_1, w_2,\ldots,w_k$ are chosen depending on the relevance or importance of each feature source. Depending on the specific application, it may be required to normalise the generated feature vector in order to assure uniform scaling. The weighted average method is applicable in object classification tasks as it allows for the incorporation of several sources of information, providing a sophisticated approach to feature combination.

4. Using baseline feature combination methods to determine accuracy

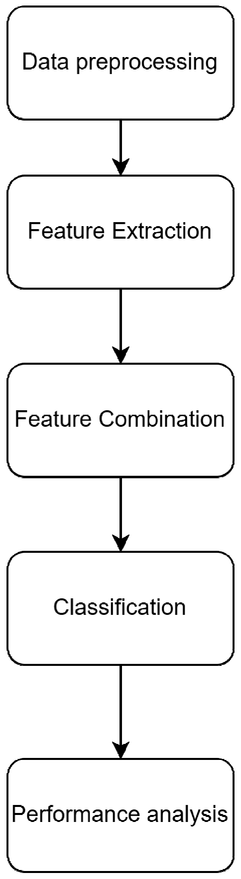

Fig.1 Flowchart of proposed model

Fig 1 shows the general outline of the proposed model. Below, we will cover the step-by-step approach for combining individual baseline features.

Concatenation is a frequently employed technique for combining features in object classification tasks involving picture datasets. The following are the sequential procedures for implementing the concatenation method:

1. Feature Extraction: Identify and extract pertinent features from theimages within the collection. These characteristics can be acquired through many methods, including manually designed feature descriptors (such as SIFT and HOG) or deep learning-based features derived from pre-trained convolutional neural networks (CNNs).

2. Normalization: Standardize the extracted features to ensure that they have comparable scales. Normalisation enhances classification performance.

3. Feature Concatenation: Combine the feature vectors obtained from various sources (e.g., multiple layers of a CNN, different feature extraction techniques). The process of concatenating the features produces a unified vector that represents each image in the dataset.

4. Classifier Training: Utilise the merged feature vectors acquired in the preceding step to train a classification model. Select an appropriate classifier, such as Support Vector Machines (SVM),

Random Forests, or Neural Networks. Partition your dataset into separate training and validation/test sets to assess the performance of the model.

5. Model Evaluation: Assess the performance of the trained classifier by using either the validation or test set. Evaluate the classification performance of the model by calculating performance metrics such as accuracy, precision, recall, and F1-score.

6. Fine-tuning (Optional): Depending on the classifier's performance, you have the option to refine the algorithm by modifying parameters like the normalisation technique, the selection of classifiers, or the feature selection procedure.

7. Final Model Selection: Choose the ultimate model based on its performance on the validation set. Additionally, it would be prudent to employ methods such as cross-validation to acquire more reliable assessments of the model's efficacy.

8. Model Deployment: After selecting the final model, proceed to deploy it for inference on new, previously unknown data. This may entail incorporating the model into a more extensive system or application for the purpose of object classification activities.

The average method is a simple and frequently successful approach to integrating features.:

1. Feature Extraction: Initially, the objects that are intended to be classified are used to extract features. Colour histograms, texture descriptors, or deep learning features derived from convolutional neural networks (CNNs) are among the numerous useful characteristics of the objects that can be included in the features.

2. Normalisation (Optional): It is recommended that the features be normalised prior to integrating them using the average approach. The ultimate outcome is guaranteed to be equally influenced by features with disparate scales through normalisation. Min-max scaling and z-score normalisation are among the most frequently employed normalisation techniques

3. Combination: Upon extracting and potentially normalising the features, it is possible to ascertain the average of each feature across all items in the collection. For example, when each entity has three characteristics, the mean of each characteristic is calculated individually for all entities.

4. Categorization: The average feature vector can be used as the input for your preferred classification technique after it has been calculated. A fundamental model, such as k-nearest neighbours (KNN), support vector machines (SVM), or more advanced models like deep neural networks, is one option for a classifier.

5. Evaluation: In the final analysis, you evaluate the efficacy of your classification model by assessing its performance on a unique test dataset using metrics such as precision, recall, accuracy, or F1-score. This assessment assists in determining the model's capacity to manage novel, unexplored data.

Calculating a weighted average of individual feature vectors is a key step in the algorithm for feature combination. By assigning weights to each vector, a unified feature vector can be generated. Here are the step-by-step instructions for implementing this method in the process of classifying objects using image datasets:

1. Extracting Features: Identify and isolate relevant characteristics from the photographs in the dataset. There are various methods to obtain these features, such as using manually designed feature descriptors like SIFT or HOG, or utilising deep learning-based features extracted from pre-trained convolutional neural networks (CNNs).

2. Standardising the extracted features is important to ensure that they have comparable scales. Optimising data organisation improves efficiency in categorization tasks.

3. Assigning Weights: Determine the importance or relevance of each feature vector and allocate appropriate weights accordingly. Weights can be determined either through empirical methods or by using techniques like feature selection or dimensionality reduction approaches.

4. In order to calculate the weighted average, you will need to multiply each feature vector by its corresponding weight. Add up the resulting products to get the weighted average of the feature vectors.

5. Training the Classifier: Use the combined feature vector obtained in the previous step to train a classification model. Choose a suitable classifier, like Support Vector Machines (SVM), Random Forests, or Neural Networks. Split your dataset into training and validation/test sets to evaluate the model's performance.

6. Evaluate the performance of the trained classifier by applying it to the validation or test set. Calculate performance measures, such as accuracy, precision, recall, and F1-score, to assess the classification performance of the model.

7. Optional fine-tuning: If the classifier's performance needs improvement, you can refine the algorithm by adjusting parameters like the weights assigned to the feature vectors or the choice of classifiers.

8. Selecting the Final Model: Determine the best model by evaluating its performance on the validation set. Furthermore, it is recommended to utilise techniques like cross-validation to obtain more accurate evaluations of the model's performance.

9. Deploying the Model: Once you have selected the best model, it's time to put it into action and make predictions on new and unfamiliar data. It may be necessary to integrate the model into a larger system or application specifically designed for object classification tasks.

Implementing the product method of feature combination in object classification requires combining features at the feature-level through a multiplication (or product) operation. Implementing this method can be relatively simple once you have extracted the necessary features from your input data. Here is a detailed procedure for implementing the product method:

1. Extracting Features

Extracting features: Begin by extracting features from your input data, such as images. These features can encompass a range of characteristics, from basic attributes such as colour histograms and texture features, to more complex ones like SIFT keypoints and HOG descriptors, and even advanced features derived from deep learning models such as CNNs.

2. Representing Features

Representation in Vector Form: Representing each object or image as a feature vector is essential.

3. Normalisation (optional).

Ensure Equal Contribution: Each feature vector is normalised to have zero mean and unit variance, guaranteeing that all features contribute equally during the combination process.

4. Combine Features: Perform the product operation to combine the features.

5. Classification

- Train Classifier: Use the combined feature vectors fi as input to train your classifier (e.g., SVM, Random Forest, etc.).
- Predict: For new objects, extract the same types of features, combine them using the product method, and use the trained classifier to predict the object class.

6. Evaluation

- Evaluate Performance: Evaluate the performance of your classifier using standard metrics such as accuracy, precision, recall, and F1-score to assess how well the combined features improve classification compared to using individual features alone.

5. Classifier algorithms used

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

A. KNN

Classification algorithm KNN is basic and instance-based.It forecasts a new data point's class using its K nearest neighbours' majority class. Number of neighbours (K) and distance metric (e.g., Euclidean distance) are important parameters.KNN prediction is straightforward but computationally costly, especially with large datasets.

B. Support Vector Machine (SVM)

SVM is a supervised learning technique used for classification and regression, mostly classification. SVM calculates the appropriate hyperplane to split data points by class.Maximum margin between hyperplane and closest data points (support vectors). Uses kernel functions like polynomial, RBF, or sigmoid to efficiently handle non-linear decision boundaries.SVMs work in high-dimensional domains with more characteristics than samples.

C.Random Forest

Ensemble learning approach Random Forest trains several decision trees.Each forest tree predicts independently using a bootstrap sample of data. Aggregating all tree predictions (most votes for classification, average for regression) yields final predictions.Random Forests give feature relevance scores, resist overfitting, and handle high-dimensional data. Their effectiveness and scalability in machine learning tasks make them popular.

D.AdaBoost (Adaptive Boosting)

AdaBoost uses ensemble learning to produce a strong classifier from weak ones.It trains weak classifiers on updated data repeatedly. Adjusts the weights of erroneously categorised instances to increase classifier focus on challenging cases.AdaBoost helps poor learners improve accuracy without overfitting. It's extensively utilised in classification jobs that benefit from several classifiers.

E.Gradient Boosting Classifier,

GBM, or Gradient Boosting Classifier, is a powerful ensemble learning algorithm used for classification:Gradient Boosting consecutively constructs decision trees that fix each other's faults.Adding trees that minimise loss optimises a loss function (usually gradient descent).Helps shallow decision trees become strong learners that can predict accurately.Gradient Boosting handles complex variable interactions well and performs well. Its versatility and capacity to simulate non-linear interactions make it popular in competitions and machine learning.

F.Bagging

It enhances machine learning algorithm stability and accuracy. Bagging creates several bootstrap samples from the original dataset. An independent base classifier (typically decision trees) is trained on each bootstrap sample.Averaging all base classifier predictions for regression tasks or voting for classification tasks yields final predictions.By combining predictions from models trained on diverse data subsets, bagging decreases variance and overfitting. Unstable models (sensitive to tiny training data changes) benefit from it and improve predicted performance.

G.Logistic Regression (LR)

Logistic Regression (LR) is a key supervised learning technique for binary classification.Logistic regression uses a logistic function to describe binary outcome probability, resulting in values between 0 and 1.It divides classes by fitting a linear decision boundary to feature space.Optimisation methods like MLE or gradient descent estimate model parameters (coefficients).When characteristics and goal variables are linear, Logistic Regression is easy to interpret, computationally efficient, and effective. Many fields use it because of its simplicity and efficacy in binary classification.

H.The Naïve Bayes Classifier

The Naïve Bayes Classifier is an effective technique for creating fast machine learning models that can effectively predict. It is a probabilistic classifier that evaluates object likelihood.

I. Decision Tree Classifier

A Decision Tree Classifier is a supervised algorithm for classification and regression tasks. It creates a decision tree by recursively partitioning data by feature values.The method picks the optimal feature to partition the data at each node, usually to maximise classification information gain or minimise regression variance. New data points travel from the root to a leaf node, which determines the class label (for classification) or predicted value (for regression) after training.Decision Trees evaluate non-linear interactions and capture complex decision limits. They can overfit, however pruning or ensemble approaches like Random Forests or Gradient Boosting can help.

6. EXPERIMENTAL RESULTS

The results and discussions of this paper are derived from extensive work conducted using python programming. The experiments were conducted on a computer with a Core i5 Processor, 4GB Memory, 500GB storage, and Ubuntu 16.04 Operating system.We have included here Five data sets. They are

Data set 1- Oxford Flower -17
Data set 2- Event 8
Data set 3 – Scene 15
Data set 4-TMA
Data set 5- Brain MRI

In the results, we initially assessed the classification accuracy by only using single features, without any feature combination.Next, we assessed the classification accuracy by combining various features to demonstrate the effectiveness of feature combination compared to using single features. We have assessed the same using various classifiers. The obtained results are transformed into graphs and then analysed. In order to obtain desirable results, it was made sure that the system remains stable.

6.1

Classification Accuracy

1. Flower-17 Data Set

The Oxford Flower-17 dataset is a highly regarded benchmark dataset in the field of computer vision, particularly for applications like image classification and object recognition. The collection contains 17 different floral types, each representing a specific species. The dataset contains a collection of floral photos, with varying numbers of photographs in each category. These images are typically high-resolution photographs taken in different situations.

Table1: Oxford Flower -17 Classification Accuracy

Feature Combination Method Used	Classifier Used								
	Decision tree	KNN	SVM	Random Forest	AdaBoost	Gradient Boosting	Bagging	Logistic Regression	Naïve Bayes
Concatenation	79.2	81.6	**83.25**	80.25	78.2	78.6	83	75.2	73.88
average	85.1	84.9	82.21	85.21	86.1	83.9	**87.53**	83.2	86.23
weighted average	85.7	86.5	85.44	85.44	86.7	84.5	**88.44**	83.11	87.25
Product	77.6	79.1	78.3	76.3	78.6	76.1	**79.3**	74.22	79.21
Best Single	76.5	**77.7**	76.4	75.4	77.3	74.4	75.3	73.2	77.4

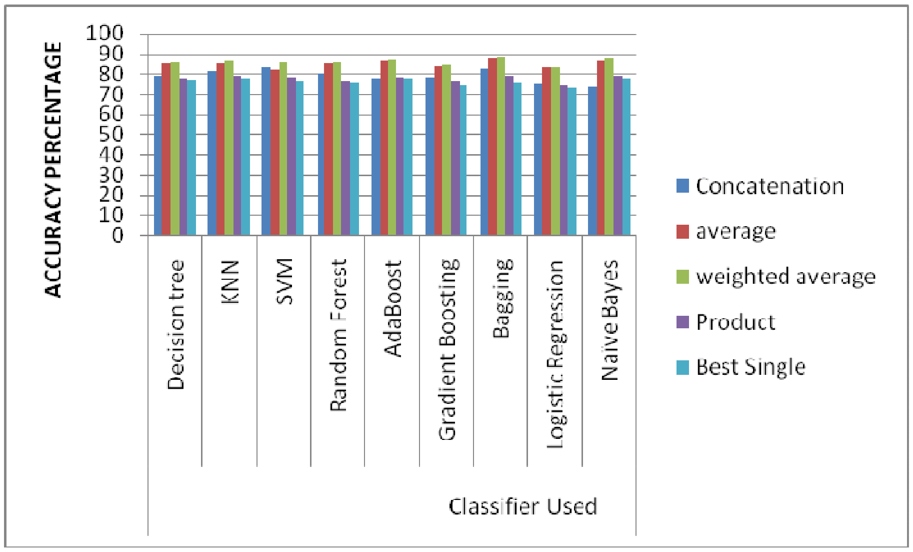

Fig-2 Performance comparison of Individual Feature and Baseline Combination Method for Flower 17 dataset

According to the data in Table 1, the highest accuracy of 88.44% was achieved using the weighted average method with the Bagging Classifier on the Flower 17 dataset.

2. Event 8 data set

The Event-8 dataset [14] comprises photos from eight sport event categories: badminton, bocce, croquet, polo, rock climbing, rowing, sailing, and snowboarding. Every category contains a range of 130 to 250 photos. In addition to categorising events depicted in still photos, the dataset also poses additional obstacles for classification. These challenges include the presence of complex and diversified backdrops, as well as the existence of various positions, sizes, and viewpoints of foreground objects.

Table 2 : Event 8 data set Classification Accuracy

Feature Combination Method Used	Classifier Used								
	Decision tree	KNN	SVM	Random Forest	AdaBoost	Gradient Boosting	Bagging	Logistic Regression	Naïve Bayes
Concatenation	77.6	81.6	84.25	81.25	77.2	77.6	**85**	73.2	74.88
average	83,2	84.9	85.21	83.21	88.1	82.9	84.52	83.2	**86.23**
weighted average	82.5	84.5	86.44	84.44	85.7	84.5	**89.43**	83.11	85.25
Product	76	77.1	**86.3**	75.3	75.6	74.1	74.3	73.22	74.21
Best Single	85.21	**86.21**	75.4	74.4	77.2	73.4	77	72.2	73.4

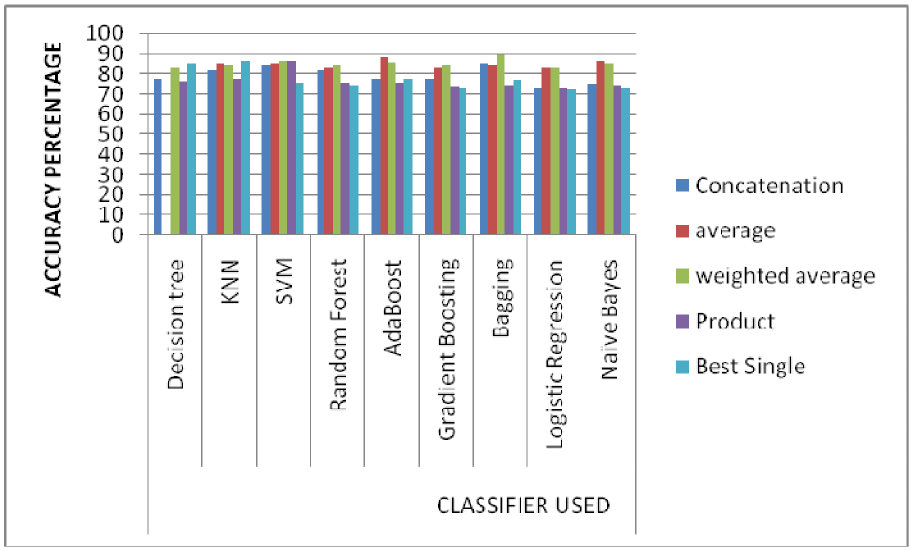

Fig-3 Performance comparison of Individual Feature and Baseline Combination Method for Event 8 data set

According to the data in Table 2, the highest accuracy of 89.43% was achieved using the weighted average method with the Bagging Classifier on the Event 8 dataset.

3. Scene 15 data set

The Scene-15 dataset is a widely used benchmark dataset in the fields of computer vision and machine learning for tasks related to classifying scenes. The dataset comprises a compilation of labelled photos that depict 15 distinct sorts of scenes. The dataset includes annotations for each image, specifying its relevant scene category.

Table 3: Scene 15 data set Classification Accuracy

Feature Combination Method Used	Classifier Used								
	Decision tree	KNN	SVM	Random Forest	AdaBoost	Gradient Boosting	Bagging	Logistic Regression	Naïve Bayes
Concatenation	74.6	81.6	84.25	82.25	75.2	77.6	**85.33**	72.2	73.88
average	83.2	83.9	**85.21**	84.21	82.1	82.9	84.51	84.2	84.23
weighted average	80.5	83.5	84.44	83.44	81.7	84.5	**89.42**	85.11	84.25
Product	83	76.1	**86.3**	85.3	77.6	84.1	84.2	77.22	72.21
Best Single	77.21	74.21	**79.4**	74.4	72.2	73.4	73	72.1	72.4

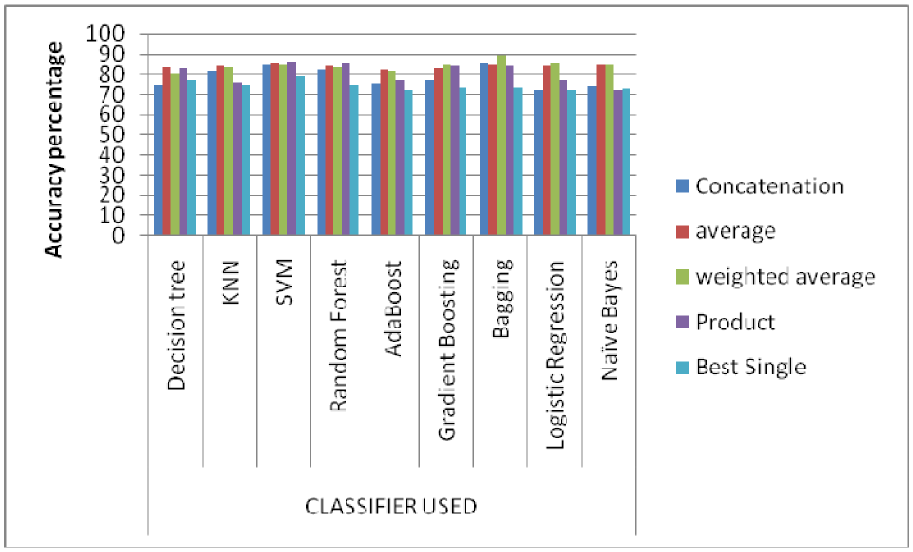

Fig 4 Performance comparison of Individual Feature and Baseline Combination Method for Scene 15 data set

According to the data in Table 3, the highest accuracy of 89.42% was achieved using the weighted average method with the Bagging Classifier on the Scene 15 dataset.

4. TMA data set

The initial dataset comprises 1272 images of tissue microarrays (TMA) depicting renal cell cancer. All the photos have dimensions of 80 X 80 pixels and are positioned at the centre of the cell nuclei. Out of the total number of images, 890 are labelled as benign and 382 are labelled as malignant.

Table 4: TMA data set Classification Accuracy

Feature Combination Method Used	Classifier Used								
	Decision tree	KNN	SVM	Random Forest	AdaBoost	Gradient Boosting	Bagging	Logistic Regression	Naïve Bayes
Concatenation	81.6	81.2	83.25	82.25	74.2	76.6	**85.33**	71.2	72.88
average	83.2	83.9	**85.21**	84.21	83.1	82.9	83.51	83.2	81.23
weighted average	80.5	83.5	84.44	83.44	81.7	84.5	**89.49**	85.11	83.25
Product	81.2	86.1	**86.3**	85.6	77.8	75.9	84.2	85.22	85.21
Best Single	74.21	**74.26**	72.4	74.4	72.2	73.4	73	73.1	72.3

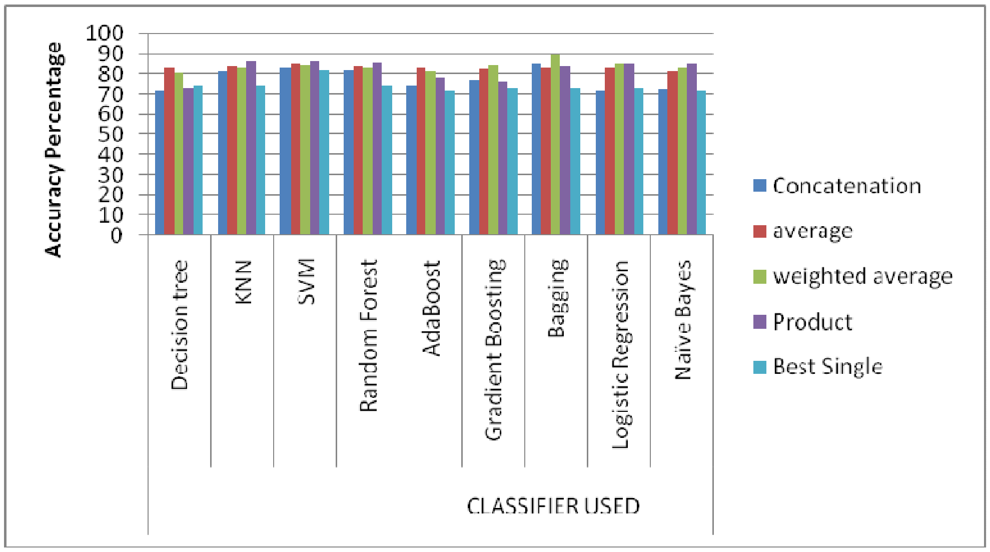

Fig 5 Performance comparison of Individual Feature and Baseline Combination Method for Scene 15 data set

According to the data in Table 4, the highest accuracy of 89.49% was achieved using the weighted average method with the Bagging Classifier on the TMA dataset.

5. Brain MRI data set

The dataset comprises magnetic resonance imaging (MRI) scans of 64 individuals diagnosed with schizophrenia and 60 individuals without any mental health conditions.

Table 5: Brain MRI Dataset Classification Accuracy

Feature Combination Method Used	Classifier Used								
	Decision tree	KNN	SVM	Random Forest	AdaBoost	Gradient Boosting	Bagging	Logistic Regression	Naïve Bayes
Concatenation	71.6	71.2	73.25	72.25	74.2	**76.6**	75.33	71.2	72.77
average	73,2	73.9	**75.21**	74.21	73.1	72.9	73.51	73.2	71.23
weighted average	70.5	73.5	74.43	73.44	71.7	74.5	**79.42**	75.11	73.25
Product	73	76.1	**77.9**	75.6	77.7	75.9	74.2	75.22	75.21
Best Single	71.21	**74.21**	72.4	74.4	72.2	73.4	73	73.1	72.3

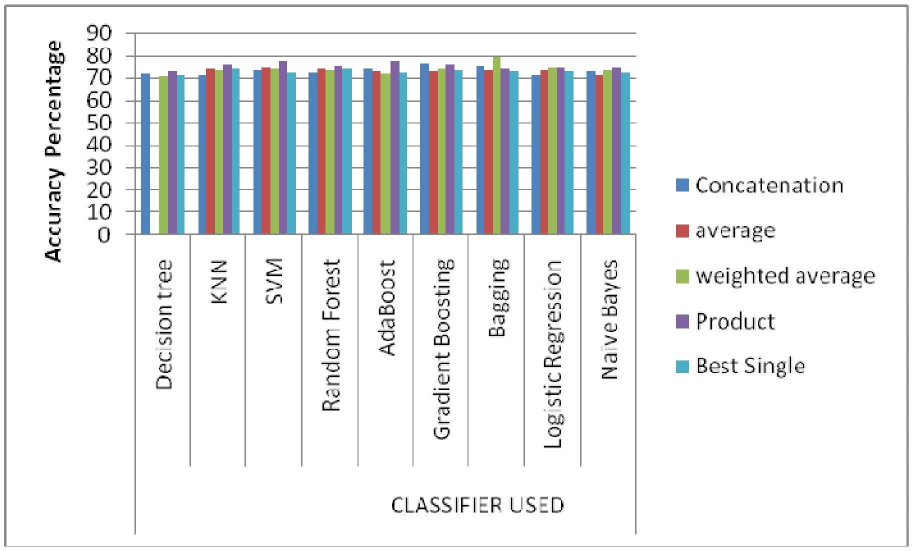

Fig 6 Performance comparison of Individual Feature and Baseline Combination Method for Brain MRI data set

According to the data in Table 5, the highest accuracy of 79.42% was achieved using the weighted average method with the Bagging Classifier on the Brain MRI Dataset.

Table 6 : Running Time Copmparision of highest accuracy percentage obtained by different classifiers for different datasets

Method used	Flower-17	Event-8	Scene-15	TMA	MRI
Concatenation	71.6	22.2	72.25	372.25	774.2
average	72.2	23.9	74.21	374.21	673.1
weighted average	67.5	21.5	72.43	245.44	571.7
Product	77.4	28.8	83.12	451.21	621.22

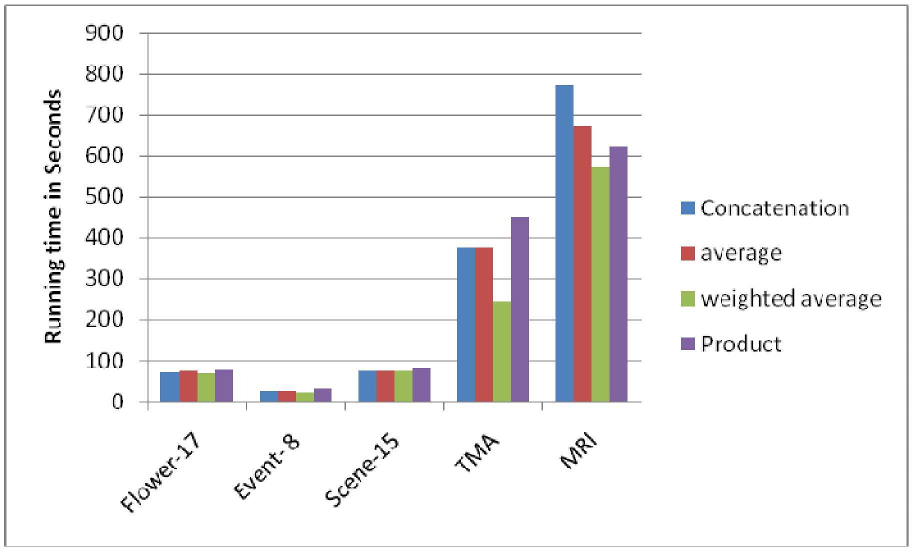

Fig 7 Running Time Copmparision of different feature combination method on different dataset

According to the data in Table 6, the minimum running time was achieved using the weighted average method in all data sets.Product method takes maximum running time in all the data sets. This may be because of the expansion of feature matrix that consumes processing time.

Conclusion & Future Work

Based on the results presented, it is evident that feature combination is highly effective in a practical setting. With the gradual addition of new features, the classification accuracy is enhanced. Therefore, these systems can provide valuable assistance from a practical standpoint and can be applied in everyday situations. If we can effectively manage the various parameters involved, we can improve the results of our application by expanding the use of feature combination. The success of classification accuracy is influenced by the number of aspects considered, so it is important to handle these parameters wisely.Many advancements have been made in feature combination, such as the introduction of Dsets clusters and other clustering algorithms. However, there is still much more potential for further progress in this area. Dealing with the dimension of the feature matrix can be quite challenging, especially when more features are added. This can significantly increase the processing complexity. Efforts can be made to minimise or reduce the dimensions of the feature matrix in order to streamline the processing. Modifying and optimising algorithms can greatly improve their performance. Developing more advanced feature descriptor algorithms can greatly improve 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" 2021 3rd International Congress on Image and Signal Processing (CISP2021)

[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, 2021, 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 holisticrepresentation 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, GholamaliRezaeirad "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-scaleand 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 fewtraining 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.