Comparative Analysis of Algorithms for Sports Image Classification

Introduction

Problem Statement

The aim of the project is to design and develop accurate and efficient sports image classification models for a diverse dataset. Model will be capable of recognizing and classifying various sports activities to aid in automating sports analysis and enhancing user experience for sports enthusiasts.

Dataset Description

The dataset contains images of 7 different classes such as Badminton, Cricket, Wrestling, Soccer, Swimming and Karate.
The dataset comprises of a total of 8227 images ensuring that we get an ample amount training data for building a robust classifier. A test set has been created with and 80%-20% split on the data. (training set has 6582 images and test set has 1645 images)
Each image is associated with specific class label.
The dataset includes a predefined training and testing data. Format of all images in .jpg.
The number of images belonging to a particular class in training set is as follows:-

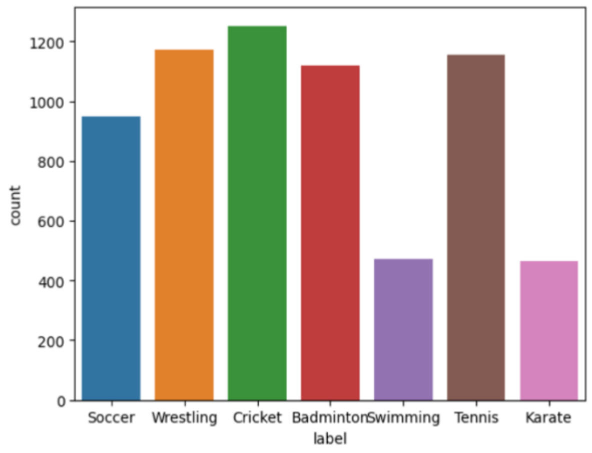


Fig1. Training set

The data has been segregated and organized into their respective folders based on specific classes in train and test folders.

Methodology

Preprocessing

The machine learning models apart from CNN require some preprocessing of the dataset before passing it in as input. The images were converted to grayscale as this step didn't result in loss of accuracy.

Algorithms

- **KNN (Nearest Neighbors)**
  It is a distance based algorithm and relies on the concept where similar data points tend to exist closely to each other. It measures the distance between 2 data points and its K nearest neighbors to make predictions.
- **Decision Trees**
  Decision tree is a simple classifier and creates a tree like model of decisions based on the features of the data. The algorithm makes predictions by following a path from root to leaf node.
  Performance of Decision tree and KNN may vary upon the dataset we choose, therefore we have executed both the algorithm for a better comparison.
- **Random Forest Classifier**
  Random forest is an ensemble method that combines multiple decision trees. (Bagged implementation of decision trees) Improvement is expected from the decision tree model as bagging takes majority voting and help in cancelling out the errors.
- **Gradient Boosting**
  Boosting combines multiple weak learners into a strong predictive model to provide better prediction accuracy.
- **SVM & Tuned SVM**
  Support Vector Machine model can also be used for classification as it can handle high dimensional feature space making it suitable for image classification.
- **CNN**
  Convolution Neural Networks are standard learners for dataset involving images. It uses filters which find correlation between neighboring pixels. Following implementations were used in CNN:

1. **RESNET50**

It is 50 layer deep CNN architecture.

2. **VGG16**
   Vgg16 is a 16 layer deep CNN architecture.
3. **VGG19**
   Vgg19 is a 19 layer deep CNN architecture.

Results and Discussion

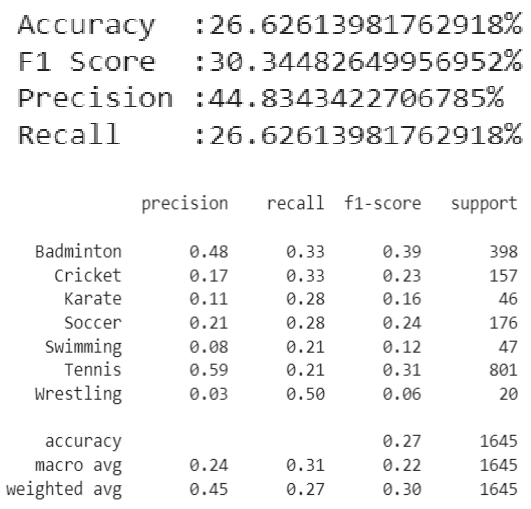
1.Logistic Regression

Logistic Regression has failed to converge , the reason can be that it failed to find correlation between the features and label.

```
/usr/local/lib/python3.10/dist-packages/sklearn/linear_model/_logistic.py:458: ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.
```

2.KNN (K nearest neighbors)

Images are high dimensional feature vectors and it is challenging for KNN to find meaningful neighbors because dimensionality is more pronounced. Images sometimes consider lot of noisy features which are irrelevant but KNN considers all features equally important. The precision, recall, and f1score values for most of the classes are quite low, indicating that the algorithm is struggling to accurately classify the sports images.
The acquired results show that KNN is less effective for image classification in our case. Dimensionality curse is more pronounced in this case.

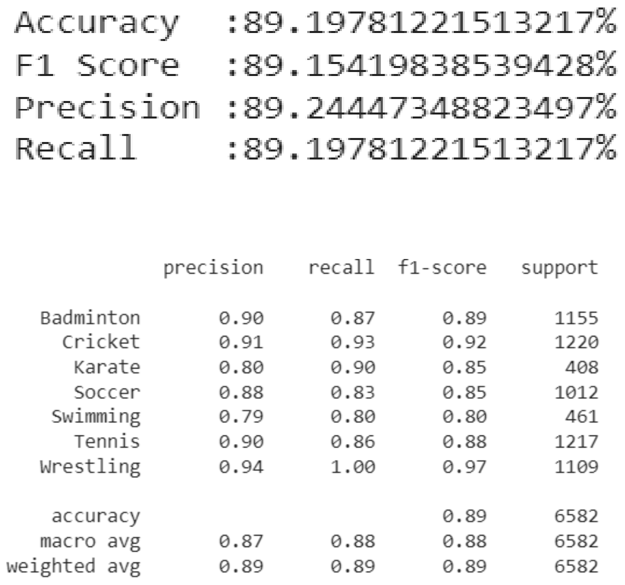


```
Accuracy  :26.62613981762918%
F1 Score  :30.34482649956952%
Precision :44.8343422706785%
Recall    :26.62613981762918%

              precision    recall  f1-score   support

   Badminton       0.48      0.33      0.39       398
     Cricket       0.17      0.33      0.23       157
      Karate       0.11      0.28      0.16        46
      Soccer       0.21      0.28      0.24       176
    Swimming       0.08      0.21      0.12        47
      Tennis       0.59      0.21      0.31       801
   Wrestling       0.03      0.50      0.06        20

    accuracy                           0.27      1645
   macro avg       0.24      0.31      0.22      1645
weighted avg       0.45      0.27      0.30      1645
```

Fig2. KNN

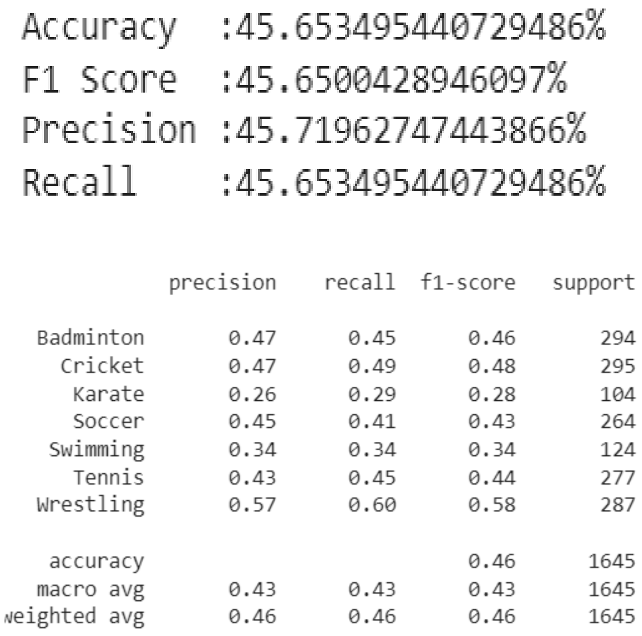
3.Decision Trees

Decision tree have showed improved performance as compared to KNN. The above figures show to performance metrics for the DTree model for train and test set. An accuracy of 89.19% was achieved on training set and 45.65% was achieved on testing set. The hyper parameter (depth of the tree) was set to 14 to reduce the overfitting and improve the generalization.



```
Accuracy  :89.19781221513217%
F1 Score  :89.15419838539428%
Precision :89.24447348823497%
Recall    :89.19781221513217%

              precision    recall  f1-score   support

   Badminton       0.90      0.87      0.89      1155
     Cricket       0.91      0.93      0.92      1220
      Karate       0.80      0.90      0.85       408
      Soccer       0.88      0.83      0.85      1012
    Swimming       0.79      0.80      0.80       461
      Tennis       0.90      0.86      0.88      1217
   Wrestling       0.94      1.00      0.97      1109

    accuracy                           0.89      6582
   macro avg       0.87      0.88      0.88      6582
weighted avg       0.89      0.89      0.89      6582
```

Fig 3. DTree_train



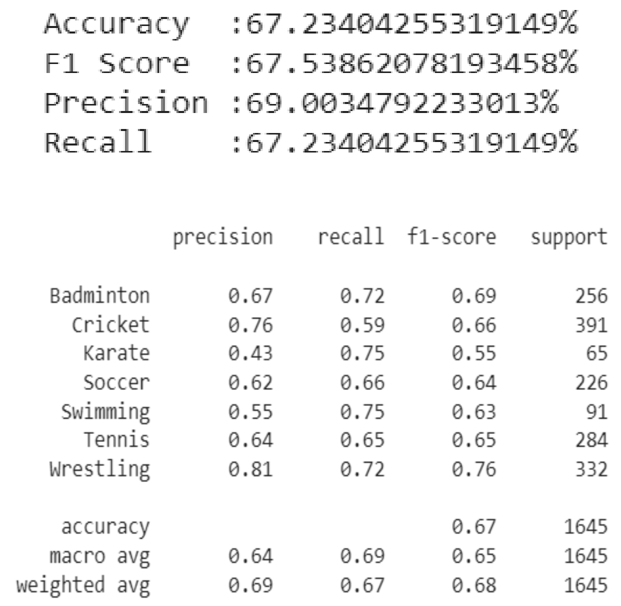
```
Accuracy  :45.653495440729486%
F1 Score  :45.6500428946097%
Precision :45.71962747443866%
Recall    :45.653495440729486%

              precision    recall  f1-score   support

   Badminton       0.47      0.45      0.46       294
     Cricket       0.47      0.49      0.48       295
      Karate       0.26      0.29      0.28       104
      Soccer       0.45      0.41      0.43       264
    Swimming       0.34      0.34      0.34       124
      Tennis       0.43      0.45      0.44       277
   Wrestling       0.57      0.60      0.58       287

    accuracy                           0.46      1645
   macro avg       0.43      0.43      0.43      1645
weighted avg       0.46      0.46      0.46      1645
```

Fig 4. DTree_test

4.Random Forest

Random forest has performed better than decision tree.

This shows that it reduces overfitting (reduces variance) by combining predictions from multiple trees and making a better generalization. Testing accuracy was found to be **67%**. Random forest improves the test accuracy by **21.4%** as compared to decision trees.



```
Accuracy  :67.23404255319149%
F1 Score  :67.53862078193458%
Precision :69.0034792233013%
Recall    :67.23404255319149%

              precision    recall  f1-score   support

   Badminton       0.67      0.72      0.69       256
     Cricket       0.76      0.59      0.66       391
      Karate       0.43      0.75      0.55        65
      Soccer       0.62      0.66      0.64       226
    Swimming       0.55      0.75      0.63        91
      Tennis       0.64      0.65      0.65       284
   Wrestling       0.81      0.72      0.76       332

    accuracy                           0.67      1645
   macro avg       0.64      0.69      0.65      1645
weighted avg       0.69      0.67      0.68      1645
```

Fig 5. Random Forest Test

4.XGBoost

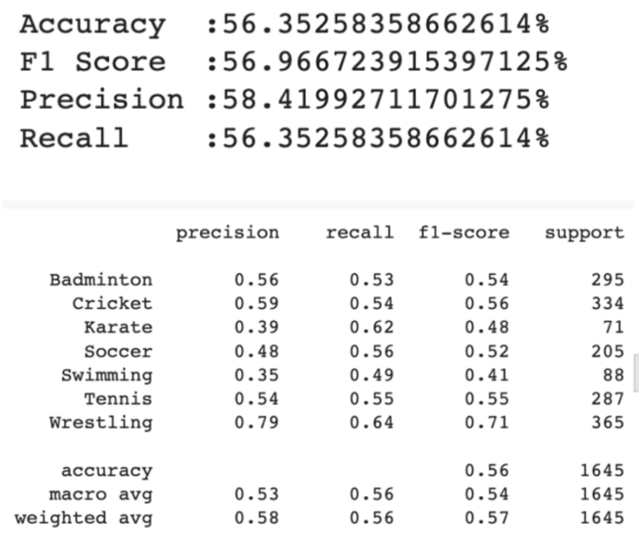
We incorporated the use of boosting shallow trees in the classification approach. The default model resulted in 100% training accuracy which shows its overfitting. Hyperparameter tuning was performed to optimize model's performance and specific parameters are mentioned in Table2 The comparative results are presented in the following table, indicating that while there is a general decline in evaluation metrics. However, an analysis of training metrics revealed reduction in overfitting, indication to improved generalization.

XGBoost Model	Accuracy	Precisionn	Recall	F1 Score
Default_train	100%	100%	100 %	100%
Default_test	67%	68%	67%	68%
Tuned_train	94%	95%	94%	95%
Tuned_test	60%	60%	59%	59%

Table 1. Metrics before and after Tuning

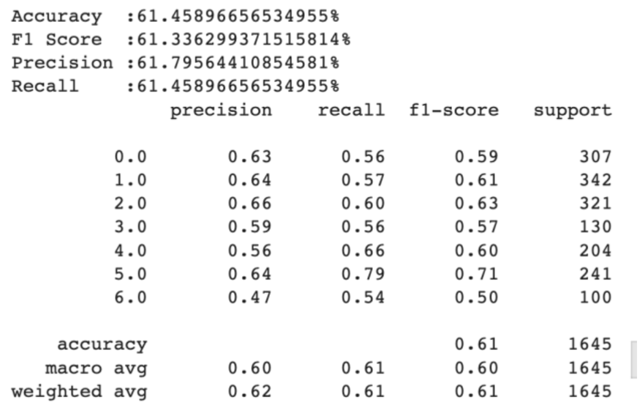
5.SVM and Tuned SVM

SVM had performance as shown in Fig 6. The test accuracy was 56%. After doing the hyperparameter tuning using randomized search, got optimal parameters as mentioned in Table 2. The provided C value (which is high)led to overfitting. There isn't any significant increase in the test accuracy which shows the poor generalization for the hyperparameters provided by the gridsearch.



```
Accuracy  :56.35258358662614%
F1 Score  :56.966723915397125%
Precision :58.41992711701275%
Recall    :56.35258358662614%

              precision    recall  f1-score   support

   Badminton       0.56      0.53      0.54       295
     Cricket       0.59      0.54      0.56       334
      Karate       0.39      0.62      0.48        71
      Soccer       0.48      0.56      0.52       205
    Swimming       0.35      0.49      0.41        88
      Tennis       0.54      0.55      0.55       287
   Wrestling       0.79      0.64      0.71       365

    accuracy                           0.56      1645
   macro avg       0.53      0.56      0.54      1645
weighted avg       0.58      0.56      0.57      1645
```

Fig6. Default_svm_test



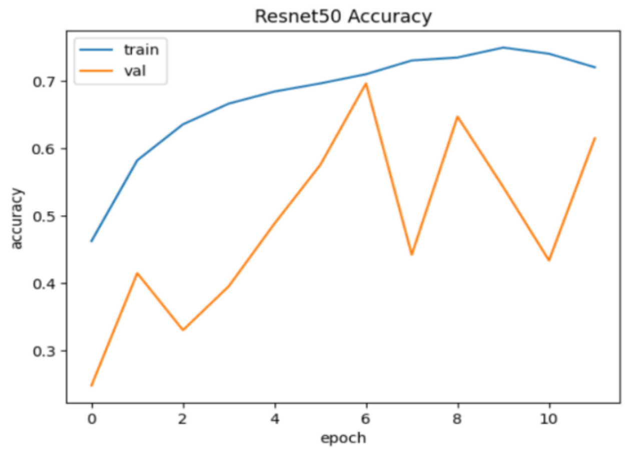
```
Accuracy  :61.45896656534955%
F1 Score  :61.336299371515814%
Precision :61.79564410854581%
Recall    :61.45896656534955%
              precision    recall  f1-score   support

         0.0       0.63      0.56      0.59       307
         1.0       0.64      0.57      0.61       342
         2.0       0.66      0.60      0.63       321
         3.0       0.59      0.56      0.57       130
         4.0       0.56      0.66      0.60       204
         5.0       0.64      0.79      0.71       241
         6.0       0.47      0.54      0.50       100

    accuracy                           0.61      1645
   macro avg       0.60      0.61      0.60      1645
weighted avg       0.62      0.61      0.61      1645
```

Fig7. Tuned_svm_test

6.CNN

CNN performed well on our dataset as compared to other models. It's built in convolution layer reduced the high dimensionality of the images without losing any of its information. CNN tries to find correlation between the pixels of the image. Generally the nearby pixels in an image are of similar value and this helps to classify an image. This enables CNN to be a very appropriate and fit network for image classification and processing. We compared to performances of ResNet50, Vgg16 and Vgg19. The respective training and test accuracy of the models has been listed in the table.

Model	**Training Accuracy**	**Testing Accuracy**	**Precision**	**Recall**
Resnet50	77%	62%	67%	58%
VGG16	97%	88%	88%	87%
VGG19	95%	86%	87%	86%



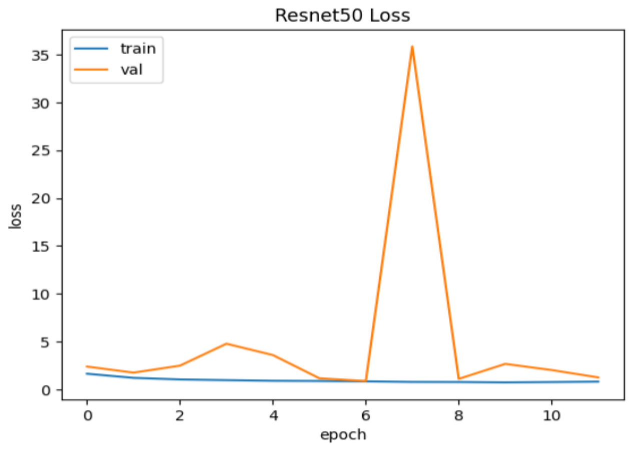
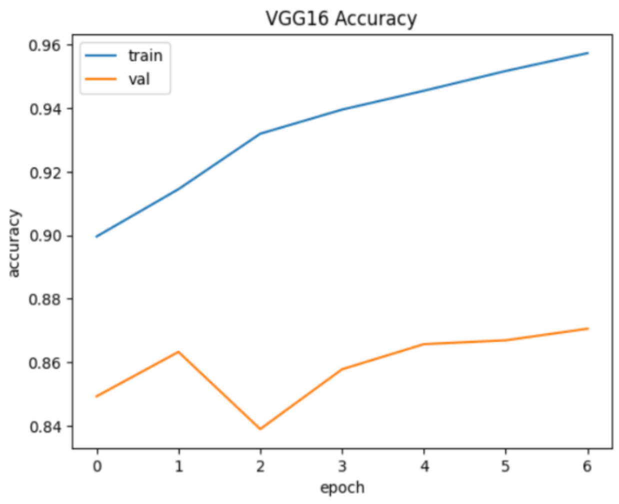


Fig 8. Resnet50 Accuracy and Loss



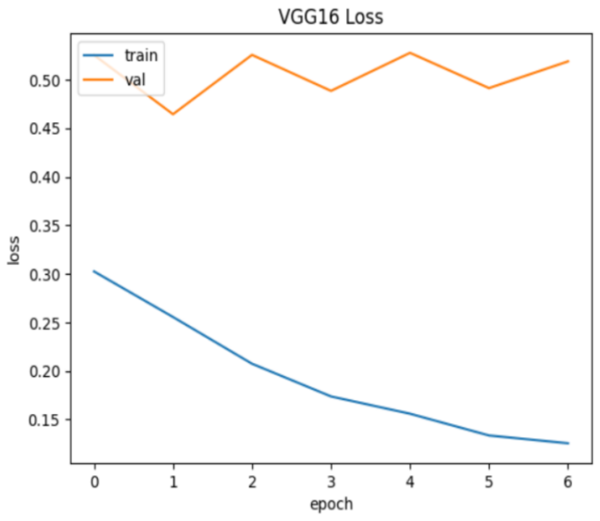
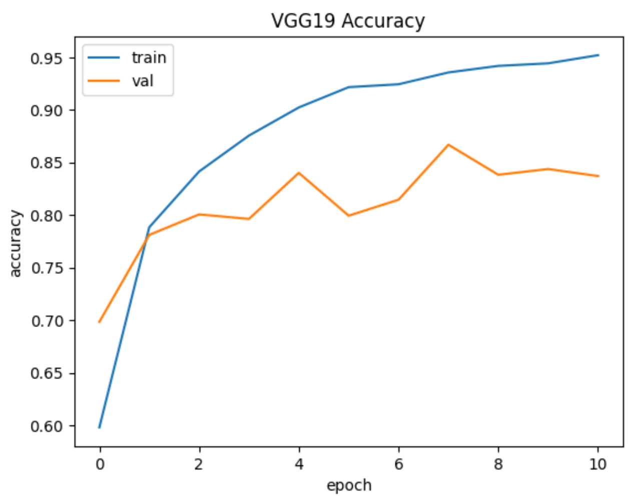


Fig 9. VGG16 Accuracy and Loss



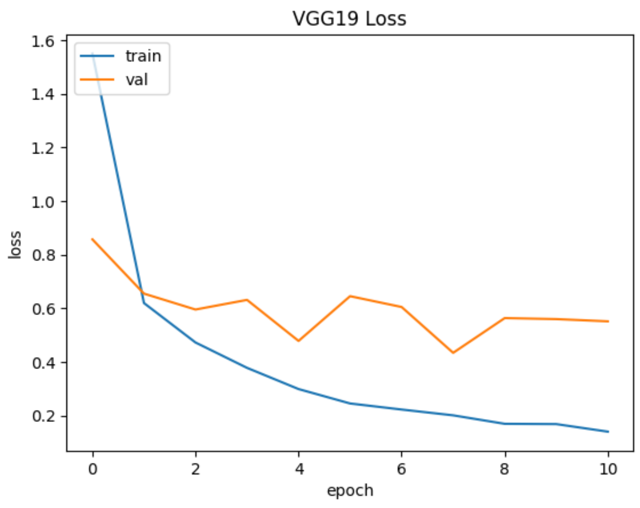
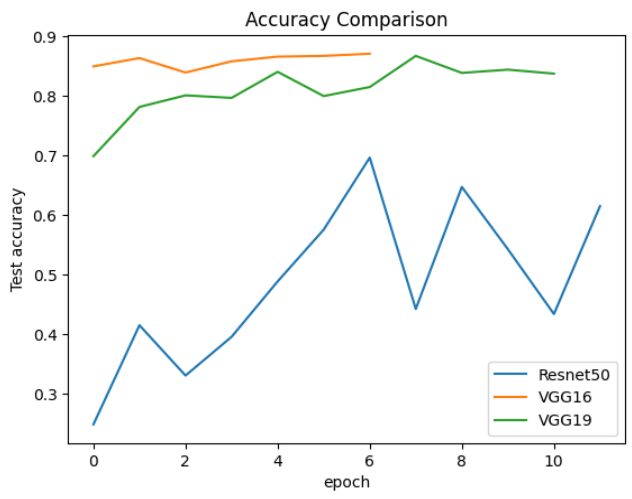


Fig10. VGG19 Accuracy and Loss



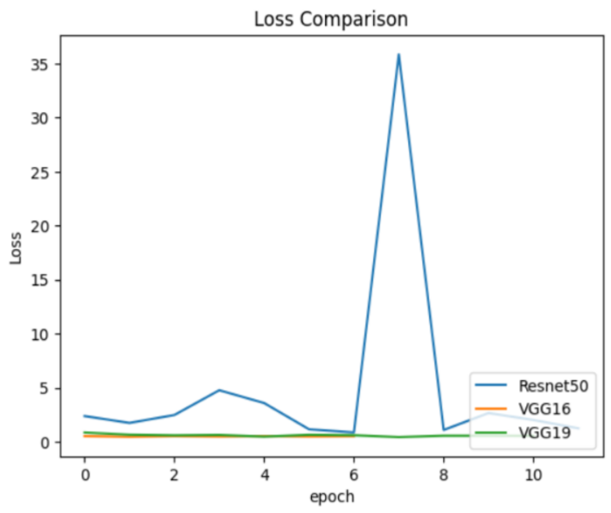


Fig 11. CNN Models Comparison

A surprising result was that Vgg16 performed better than Vgg19 and ResNet50. This might be due to the fact that the residual connections in VGG19 and Resnet50 are passing unnecessary information.

Model	**Tuned Hyperparameters**
Decision Tree	max_depth:14
XGBoost	Learning_rate:0.2,max_depth:4,n_estimators=50, booster:gbt ree
SVM	C:150, kernel: polynomial

Table2. Hyperparameters

Comparison of Models

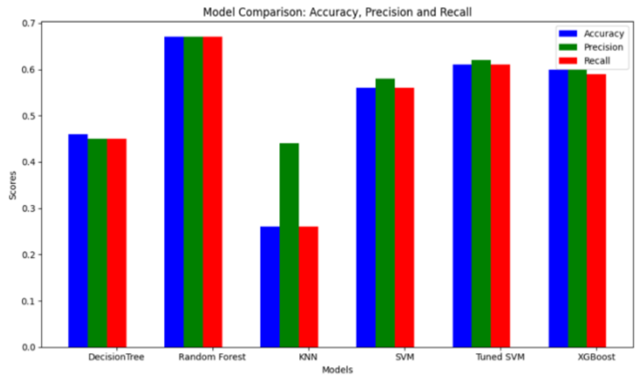


Fig 12. Comaparison of Models

Fig 11 and Fig 12 depict accuracies of the tuned models. Out of all Vgg-16 has the highest training and test accuracy. In contrast XGBoost,RandomForest, TunedSVM exhibited good training accuracies but poor generalization accuracy indicating that these were overfitting. Simpler models like Logistic regression, KNN and Decision Trees performed poorly due to the fact that they can't comprehend the correlation between feature set and the respective classes. Logistic regression failed to converge even with high number of iterations. Frequently misclassified images belong to the classes Karate and Swimming which is evident through precision and recall scores of all models belonging to these classes, the reason is that the training examples belonging to this particular set is relatively low compared to other classes as seen in Fig 1.

Conclusion

In this project, the objective was to compare different models and investigate the performance of CNNs in relation to their ability to achieve lower generalization error compared to other models. However, the results revealed a clear distinction: all models, except CNN, displayed either very low accuracy on the test set or high accuracy on the training set, indicating signs of overfitting. The key factor that set CNNs apart was their capacity to capture the local correlation among pixels in an image, which contributed to their superior generalization ability. On the other hand, the other models lacked this crucial characteristic, leading to difficulties in accurately classifying the test data.

References

Journal Article

Bansal, M., Kumar, M., Sachdeva, M., & Mittal, A. (2021). Transfer learning for image classification using VGG19: Caltech-101 image data set. Journal of Ambient Intelligence and Humanized Computing, 14, 3609 - 3620.

Conference Paper

Hussain, Mahbub & Bird, Jordan & Faria, Diego. (2018). A Study on CNN Transfer Learning for Image Classification.

Book Extraction

Vladimir Pestov. Is the k-NN classifier in high dimensions affected by the curse of dimensionality? Computers & Mathematics with Applications, Volume 65, Issue 10,2013, Pages 1427-1437

Article

Allison, Paul D. (2004) "Convergence Problems in Logistic Regression." Pp. 238-252 in Micah Altman, Jeff Gill and Michael P. McDonald (eds.), *Numerical Issues in Statistical Computing for the Social Scientist*. Hoboken, NJ: John Wiley & Sons.