**Background and Method Introduction: -**

K-Nearest Neighbours (KNN) is a simple and versatile algorithm for classification and regression. It operates on the proximity principle, which states that a data point's class or value is determined by the majority class or average value of its nearest neighbours in the feature space. KNN can be used for image classification by representing images as feature vectors and utilizing the Euclidean distance metric to identify nearest neighbours. In this implementation, image classification is performed using both a library based KNN classifier and a custom KNN classifier.

**Dataset and Tasks Description: -**

The CIFAR-10 dataset is made up of 60,000 32x32 colour images divided into ten classes, each with 6,000 images. Airplanes, automobiles, birds, cats, and other common objects are included in the classes. The dataset is divided into training and test sets.

Classification tasks: -

Use the KNN algorithm to classify images into 10 predefined categories. The dataset is pre-processed, and both library-based (scikit-learn) and custom KNN implementations are used for comparison.

**Algorithms Used: -**

1. Library-based KNN:  
A screenshot of a computer code

Description automatically generated

2. Custom KNN:

**Preliminary results and final results: -**

A screenshot of a computer program

Description automatically generated

we converted the image to NumPy array and reduce the dimension of the pixels to a single dimension.

We then use the scikit learn KNN classifier to train the model using the sample train data and test data we got.

once the model is trained, we test the model with the testing data sample and find its accuracy for given k

we then define a custom KNN model where we take the training data and for each testing image vector we get, we find the Euclidean distance between the testing image and every training image. Thus, we get a matrix of distances for each training image to each testing image.

The initial results include: -

**Library-based KNN:** Accuracy: 32%

**Custom KNN:**

Accuracy (without tuning): 10.5%

Best k after hyperparameter tuning: 5

Accuracy after tuning: 12%

Accuracy with different distance metric ('Manhattan'): 31.33%

Cross validation results: -

A screenshot of a computer

Description automatically generated  
  
**Methods of Improvements : -**

**Hyperparameter Tuning:**

Used a loop to test various k values and determined the best k based on accuracy comparison. Here in the hyper parameter tuning we check for various values of k and see which k gives the best accuracy for the testing dataset. The K which gives the best accuracy is considered as the tuned parameter.

**Different Distance Functions:**

Various distance metrics were used (Euclidean, Manhattan, and Minkowski) to assess their impact on classification accuracy. Initially we find the Euclidean distance between the testing image and to every training image. Thus, we get a matrix of distances for each training image to each testing image.

**Validations: -**

**Cross-validation:**

We used k-fold cross-validation to evaluate the model's performance across different subsets of the training data and chose the best k based on the average accuracy.

After hyperparameter tuning, the custom KNN implementation improved its accuracy and performed similarly to the library based KNN.

We had performed cross validation using various k values and derived the best results as   
A screenshot of a computer

Description automatically generated

**Visualization:** To provide a visual representation of the hyperparameter tuning process, I plotted a graph showing the accuracy for various k values.

A graph with blue lines and numbers

Description automatically generated

**Conclusions: -**

Here the KNN algorithm, whether implemented through a library or custom code, KNN is a simple yet effective method for image classification tasks. Careful consideration of hyperparameters, distance metrics, and cross-validation techniques improves the algorithm's overall accuracy and robustness. The custom implementation, while requiring more manual adjustments, demonstrates the ability to customize and adapt to specific requirements. Finally, the K-nearest neighbours (KNN) algorithm was used for image classification, with both a library-based and custom implementation. The scikit-learn KNN model served as a baseline, with a 32% accuracy on the CIFAR-10 dataset. The custom KNN model was significantly improved, including hyperparameter tuning, resulting in a 12% accuracy increase with the best k value of 5. Furthermore, various distance metrics, such as 'Manhattan,' demonstrated a notable accuracy of 31.33%. The use of cross-validation in the custom model selection process ensured robustness, and the best k found was used to achieve a reliable 31.33% test accuracy. The custom implementation provides flexibility but necessitates careful tuning, whereas the library-based approach provides efficiency.

I have used ChatGPT for reference where relevant according to policy.