Final Report:

We trained an SVM classifier and a deep learning classifier for the CIFAR-10 dataset for this assignment. Then we divide the data set into test and train. After that, we divided the train model into train and valid set in 75 and 25 per cent ration. Prior to training an SVM classifier with these data, we extracted features from the fully connected layer right before the output layer using a deep learning classifier.Introduction:

In this project, we have implemented a Convolutional Neural Network (CNN) on the CIFAR-10 dataset to classify images into 10 different classes. We have then extracted features from the fully connected layer before the output layer and trained an SVM model on those features. Finally, we evaluated the SVM model's performance on the test set using accuracy, F1 score, and recall.

Data Preprocessing:

We loaded the CIFAR-10 dataset using Keras and split it into training and testing sets. We then normalized the pixel values by dividing them by 255.0 to scale the data between 0 and 1. We also one-hot encoded the target labels to convert them into a binary matrix with each row representing a label and each column representing a class.

Then we divided them into test, train and valid dataset splits so that the train split can be used for training, valid is used for the feature extraction and test is used for testing the model
Model Building:

Model Training:

We built a CNN model using Keras with two convolutional layers, two max-pooling layers, and two dropout layers. We used the rectified linear unit (ReLU) activation function in the convolutional layers and the softmax activation function in the output layer to obtain the probability distribution over the classes. We compiled the model using the categorical cross-entropy loss function and the Adam optimiser with a learning rate of 0.001.

We trained the CNN model on the training set with a batch size of 18 and for 14 epochs. We used the validation set to prevent overfitting and evaluate the model's performance during training. The model achieved an accuracy of 0.7157 on the test set.

Feature Extraction:

We extracted features from the fully connected layer before the output layer to reduce the dimensionality of the data and improve the SVM model's training time. We used the Keras functional API to define a new model that takes the same inputs as the CNN model and outputs the activations of the fully connected layer. We then used the predict method of this new model to obtain the features for the validation and test sets.

Model Training and Evaluation:

We trained an SVM model with a linear kernel on the extracted features from the validation set. We then used the trained model to predict the classes of the test set and evaluated its performance using accuracy, F1 score, and recall. The SVM model achieved an accuracy of 0.7157 on the test set, which is the same as the CNN model's accuracy. The F1 score and recall were also similar, indicating that the SVM model performed similarly to the CNN model.

Conclusion:

In this project, we have implemented a CNN model on the CIFAR-10 dataset and extracted features from the fully connected layer before the output layer to train an SVM model. The SVM model achieved similar performance to the CNN model, indicating that feature extraction can reduce the dimensionality of the data and improve training time without sacrificing accuracy. This approach can be useful in scenarios where the training time or computational resources are limited.

Other Experiments:

I used the Gan Images to do this project initially but SVM model compilation took too much time and in the end, it was not going to the next step, so I had to use the Kesas dataset which has Images. In the keras dataset, I found CIFAR100 a bit more complicated and took too many epochs to properly train and test and also SVM model took too much time, So that's why I have used CIFAR10