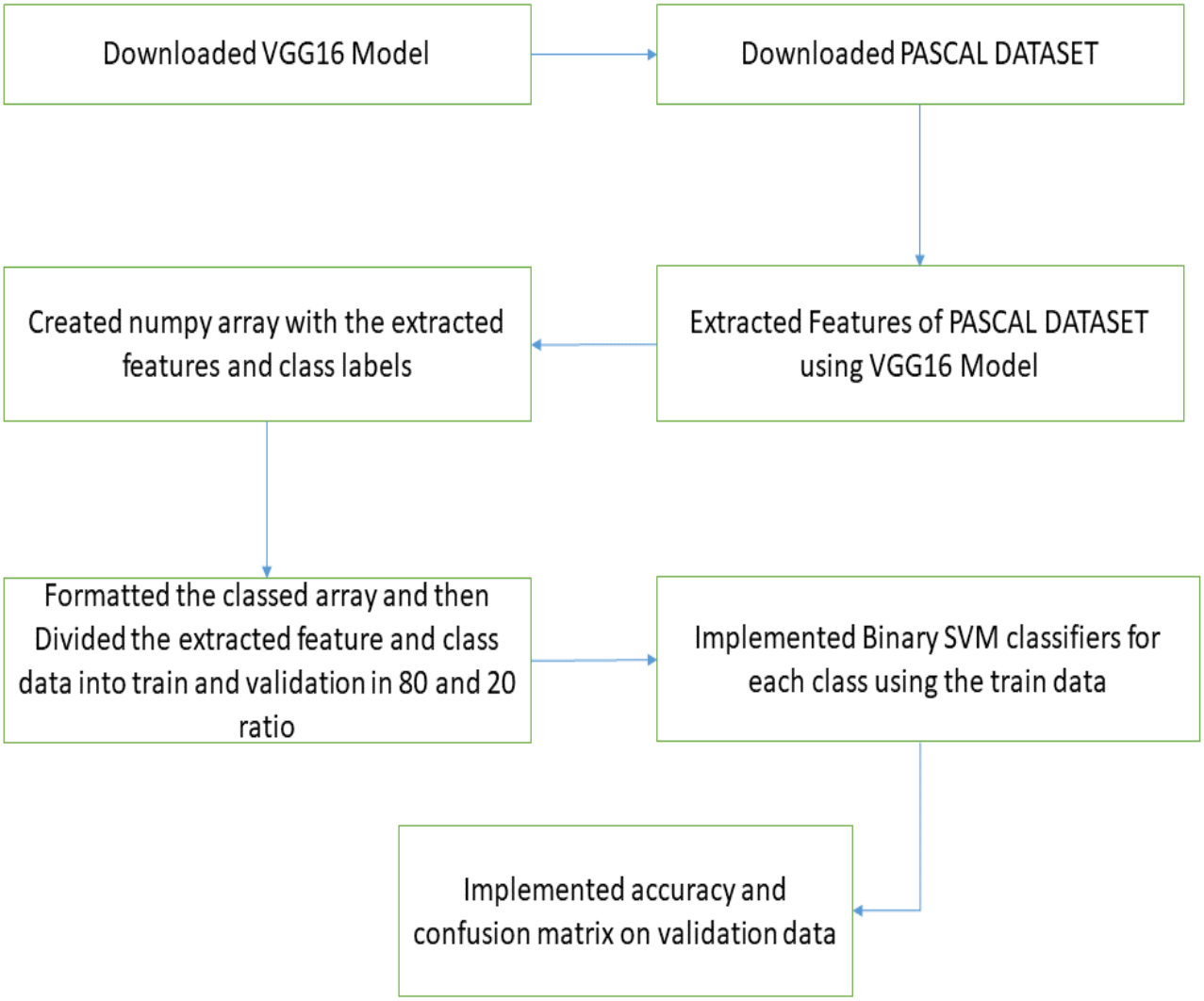
Programming Assigmnemt Report:

Name : Krishna Kumari Ravuri

Roll No: M22AI567

CNN model to train binary one-vs.-rest SVM classifiers:



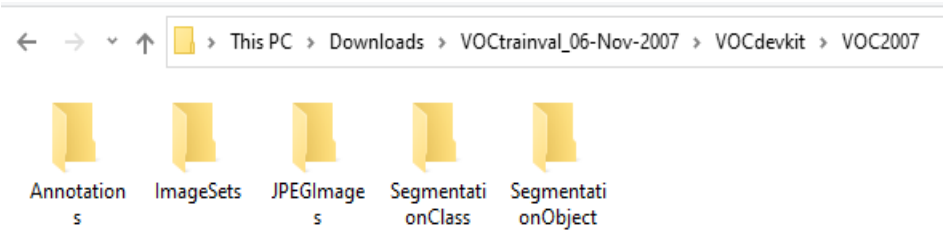
Step 1: Downloaded a pre-trained CNN model:

I used the **tf.keras.applications.VGG16** function to load the pre-trained **VGG16** model with weights pre-trained on the ImageNet dataset.Used weights='imagenet' argument to specifies that I want to download the pre-trained weights. By default,I have added include_top=True includes the fully connected layers at the top of the model.

After downloading the model, I saved it using the save method. The model saved as a Hierarchical Data Format (HDF5) file with the name 'vgg16.h5'.

Step 2: Download the PASCAL VOC 2007 dataset:

downloaded the PASCAL VOC 2007 dataset from the link provided and extract the dataset into a local directory.



The dataset is provided in the PASCAL (Pattern Analysis, Statistical Modeling, and Computational Learning) VOC (Visual Object Classes) data format, which includes image files along with corresponding XML files for annotations. The XML files contain information about the object classes, bounding box coordinates, and additional metadata.

Step 3: Extract features using the pre-trained CNN model

I have loaded the pre-trained VGG16 model, define the paths to the dataset, and specified the desired image size, then list all JPEG image files in the "JPEGImages" directory.

Implemented the code to loops over each image file, loads the image, and preprocesses it using the "vgg16_model.preprocess_input" function. It extracted the features using the pre-trained VGG16 model and appends them to the features list.

To extract the class label, I have implemented code to parse the corresponding annotation file. The annotation files are in XML format and contain information about the bounding box and class label for each object in the image. I used an XML parsing library like "xml.etree.ElementTree" to extract the class label.

Finally, the extracted features and labels are converted to NumPy arrays and saved using the np.save function.

Step 4: Train binary SVM classifiers:

I used the train_test_split function from scikit-learn to split the data into training and validation sets. I set the test size to 0.2, which means 20% of the data will be used for validation, and the remaining 80% will be used for training.

Then, I iterate over each unique class label in the training set, prepare binary labels for the current class, by using scikit-learn's SVC (Support Vector Classifier) to train binary SVM classifiers for each class in a one-vs.-rest fashion nand train a binary SVM classifier.

Using a library like scikit-learn, train binary SVM classifiers for each class in a one-vs.-rest fashion. For each class, prepare the training data by assigning labels: 1 for samples belonging to the class and 0 for samples not belonging to the class. Then, train a separate SVM classifier for each class using the extracted features.

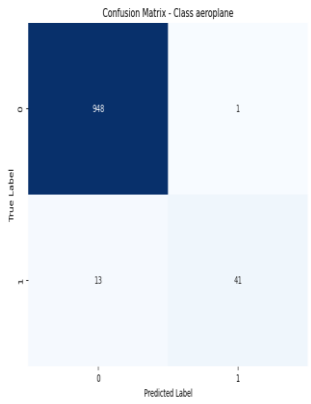
Step 5: Evaluate classification accuracy and confusion matrix

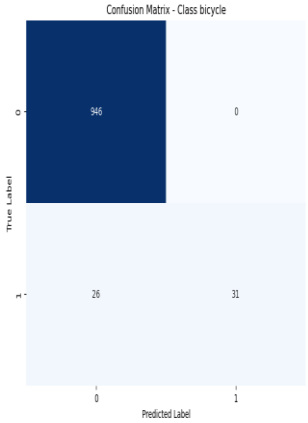
After training the SVM classifiers, we evaluate their performance on the validation set. We iterate over each unique class label in the validation set, extract the validation features for that class, and use the corresponding SVM classifier to predict the class labels. The predicted labels are assigned to the predictions array.

Calculated the overall classification accuracy and created a confusion matrix to analyze the performance of the classifiers for each class.

```
Accuracy for class aeroplane: 0.9860418743768694
Accuracy for class bicycle: 0.9740777666999003
Accuracy for class bird: 0.9830508474576272
Accuracy for class boat: 0.9880358923230309
Accuracy for class bottle: 0.9581256231306082
Accuracy for class bus: 0.9810568295114656
Accuracy for class car: 0.9501495513459621
Accuracy for class cat: 0.9850448654037887
Accuracy for class chair: 0.9302093718843469
Accuracy for class cow: 0.9850448654037887
Accuracy for class diningtable: 0.9680957128614157
Accuracy for class dog: 0.9680957128614157
Accuracy for class horse: 0.9850448654037887
Accuracy for class motorbike: 0.9850448654037887
Accuracy for class person: 0.9082751744765702
Accuracy for class pottedplant: 0.959122632103689
Accuracy for class sheep: 0.9850448654037887
Accuracy for class sofa: 0.9451645064805583
Accuracy for class train: 0.9920239282153539
Accuracy for class tvmonitor: 0.9651046859421735
```

I have implemented Confusion matrix for each class as sample I have added only Confusion matrix for 2 classes





Conclusion

Overall, the classifiers show relatively high accuracy values, ranging from around 90% to 99%. This indicates that the classifiers are performing well in distinguishing between different object classes.

Some classes, such as "train" and "boat," achieve particularly high accuracy values above 98%, suggesting that the classifiers are highly effective in correctly identifying instances of these classes.

On the other hand, some classes, such as "person," "chair," and "sofa," have relatively lower accuracy values around 90%, indicating that the classifiers struggle more with correctly classifying instances of these classes. These classes may have more inherent complexity or similarity with other classes, making them more challenging to classify accurately.

From the confusion matrix, I observed that some classes have a significant number of misclassifications with other classes, while others have minimal confusion.

References:

Pascal DataSet:

https://datasets.activeloop.ai/docs/ml/datasets/pascal-voc-2007-dataset/

https://cv.gluon.ai/build/examples_datasets/pascal_voc.html

https://mlhive.com/2022/02/read-and-write-pascal-voc-xml-annotations-in-python

https://towardsdatascience.com/coco-data-format-for-object-detection-a4c5eaf518c5

https://www.youtube.com/watch?v=VU0pGQfDZHM

use the imagenet pretrained model extract features from image:

https://www.youtube.com/watch?v=LGk2SfHLhGo

https://www.youtube.com/watch?v=7DFod37T3l4

https://fairyonice.github.io/Object_detection_with_PASCAL_VOC2012_cnn_feature_extraction.html

svm binary classifier one vs rest:

https://github.com/christianversloot/machine-learning-articles/blob/main/creating-one-vs-rest-and-one-vs-one-svm-classifiers-with-scikit-learn.md

https://machinelearningmastery.com/one-vs-rest-and-one-vs-one-for-multi-class-classification/

https://scikit-learn.org/stable/modules/generated/sklearn.multiclass.OneVsRestClassifier.html

Thank You