Report-CBIR implementation:

Introduction:

This report is demonstrating an overview of a content-based image retrieval (CBIR) implementation. The goal of this project is to test existing pre-trained neural network architectures, either trained on ImageNet or several remote sensing datasets and draw comparisons on their performance under the assumption of “varying spatial resolution” in different sources of data.

Content-based Image Retrieval is a system that retrieves images given features such as colour, shape and texture. This involves in addition to store the images in an index that is utilised for quicker search and retrieval. Although, due to the existence of many large image databases the traditional image indexing methods have been insufficient. CBIR’s implementation is by storing the features of each image on the database and then apply the matching with the features of the query image. Furthermore, it is applicable in multiple fields including medical diagnosis, security checks and crime prevention [7].

Depending on the requirements of each researcher, a CBIR can be implemented in various ways. Some famous systems that are still applicable until today, such as the QBIC developed by IBM and Almaden Research Centre, allows users to pose and refine queries based on visual features such as colour, texture and shape [8]. In addition, a system developed by the Beckham institute at the university of Illinois, known as MARS, supports colour, spatial layout, texture and shape matching [8].

In determining the latent representations and extracting the most useful information from image data has been a challenging problem for researchers. Several characteristics can make an image unique for a CBIR system being capable of interpreting correctly its content. The key characteristics, such as colour, texture, and shape, can be determined by methods and algorithms such as colour histograms or discrete wavelet transforms (DWTs) which are used to extract specific features unlike with a CNN.

Hany F. Atlam *et al.* [2] presented two approaches at which on the first they extracted texture features using Gray Level Co-Occurrence Matrices (GLCM) at which they were used for search and retrieval of relevant images. Then, they selected the features with the highest retrieval accuracy and combined them with DWT. The second approach was to combine texture and colour features together, at which a HSV colour histogram was used for the colours.

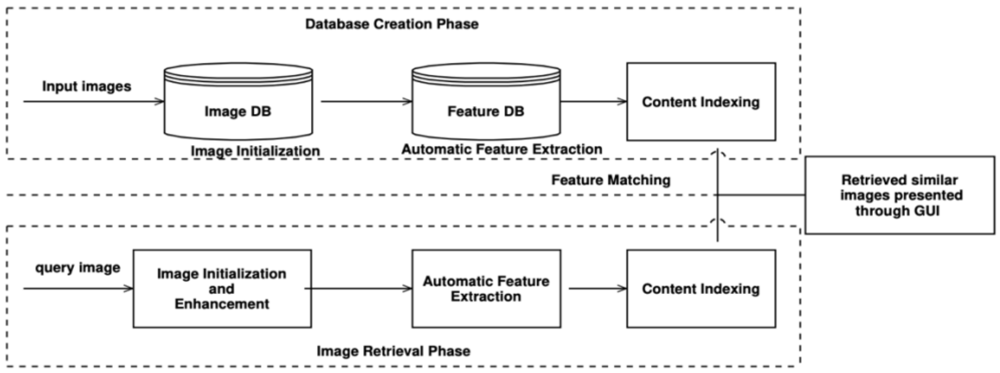
Ramesh K. Lingadalli et al. [3] have used again a colour histogram at which they converted images from RGB to HSV and additionally they formed feature vectors by normalising the histogram to be used for search and retrieval. Additionally, they used a GLCM at which they converted the images from RGB to grayscale and by applying the algorithm they extracted several statistical features such as energy, homogeneity and correlation that were combined to be used for searching. Unlike with other researchers, they also extracted shape features by using morphological gradients and moment invariants (known as Hu moments) combined together.

Ivica Dimitrovski *et al.* [4] have utilised deep learning approaches such as traditional CNN architectures on multiple remote sensing datasets. In addition, they also trained more recent architectures that are quite different from traditional methods. They used the ViT and Swin transformers which consist of an attention mechanism, as well as the MLPMixer which is more of an MLP-based approach. They attempted to tackle several tasks apart from image classification, such as object detection, image semantic segmentation and crop-type prediction. Image semantic segmentation again aims to identify objects in an image but does it differently with respect to an object detection system. Here, it assigns a label to each pixel on the image with a corresponding class of what the pixel represents. In crop-type prediction, unlike semantic segmentation, a temporal component is involved.

Xiaodong Zhang *et al.* [5] proposed region-based CNN at which candidate regions are generated first and then classifying and locating the objects that exist in these regions. Due to the complexity in images such as been overlarge and having uneven size in training samples, they utilised a framework known as DM-FPN where it combines low resolution with high resolution features together.

Motivation for current work:

The approaches that will be followed for feature extraction will be existing pre-trained CNN architectures for features representation. The main architectures that will be used are EfficientNet-b0 and ResNet50. Since the goal of this task is to fine-tune our datasets with the pre-trained architectures, we will apply some "freezing" to several parts of the models. Freezing is the technique that is used to avoid retraining from scratch specific model layers that are chosen. We applied this on the fully connected layers of EfficientNet and ResNet since we insert manually our own. With this applied, by visualising the model summary of each model the number of trainable parameters is inspected, and this is 4 million and 23 million for EfficientNet and ResNet respectively. To keep more sensible numbers for ResNet the fourth block of layers is frozen and that drops the number to 8 million trainable parameters. A KD tree is used as the data structure index where feature vectors will be stored and to be used for quick search and efficient retrieval and finally, we will use simple Euclidean distances for measuring the similarity between the feature vectors and the query image's feature vector after pre-processing has been implemented to it. A figure from [3] is shown below to demonstrate the framework.



Methodology:

Data Pre-processing:

The three datasets will be split separately in portions of 60%, 20%, 20% for training, validation and testing sets respectively. Then, several data augmentation techniques will be applied to each batch of the training data to create a variety of examples at which the network will be learning.

These include transformations such as random horizontal and vertical flips with a 0.5 probability, rescaling of the pixels from a [0, 255] range to [0,1] and finally normalisation of the pixels according to the respective mean and standard deviation of each of the (R, G, B) channels. The reason rescaling and normalisation are implemented is to make learning for the CNN easier in the sense of not treating any extreme values of the [0, 255] scale as "outliers" and to reduce high variability in the predictions the network will output, known as avoiding the "overfitting effect". For validation and test data, we only apply rescaling and normalisation.

Particularly, regarding the FAIR_1M dataset, it's been observed that there were some inconsistencies regarding the dimensionality of the image. For instance, as stated in the instructions the dimensionality of this dataset should have been 1500x1500 although most of the images had 1000x1000. Furthermore, for the number of channels usually we may expect a dimensionality of 3 which represent the (R, G, B) channels. Again, in most images a fourth channel has been observed which represent the Alpha channel. This channel represents a colour component at which it controls the level of transparency in pixels, and it determines how pixels render when blended [1]. For these exceptions, we applied two extra transformations in the data augmentation pipeline at which we resized the 1500x1500 images to 1000x1000 and converted the RGBA images to RGB.

For data labelling, the names for each object were replaced with numeric values. For the Sentinel_2 dataset, 0 is represented as "NoShip" objects and 1 for "Ship". For RESISC_45, 0 depicts "bridges", 1 shows "airplanes" and 2 for "ships." Likewise for the FAIR_1M, 0 shows "neighbourhoods", 1 shows "airplanes" and 2 shows "ships". The final predictions on the testing sets are demonstrated as numbers.

Model implementation:

CNNs have been recognised as a great approach for many computer vision tasks, due to several features that are implemented in their structure. In the beginning of the network, they are composed of convolutional layers. These layers are the ones responsible for extracting useful features that exist in an image, such as colour, contours, and shapes. This is done by applying filters/kernels of lower dimensionality that apply the convolutional operation to retrieve multiple feature maps of a lower dimension and including the extracted features. In addition, pooling layers follow up right after convolutional layers. These layers are used to drop the dimensionality of the feature maps to a smaller one, by preserving the most useful information.

Along with these layers, an activation function is used to apply a transformation to the input and obtain it in the output. The most applicable one is the ReLU at which is useful to address non-linearity in the data as well as avoiding effects in backpropagation such as the "vanishing gradient".

After features are extracted, the layers are vectorised as a 1-D vector in the form of a fully connected layer where the final output is retrieved. Here, depending on the specified task, a different activation function is used. For binary classification problems a sigmoid is used. This function is used a lot in applications such as logistic regression where the final output to be retrieved is binary and it is used to predict the probability of classes. Depending on the highest probability, the respective class is chosen. For multi-class classification problems, a softmax is used, which is a generalisation of the sigmoid. It forms a linear combination for multiple sigmoids which formalises a probability distribution that is suitable for more than two classes. In the case of RESISC_45 and FAIR_1M datasets, the softmax activation is used and for Sentinel_2, the sigmoid is used.

In addition to layers, some regularisation techniques are applied to address effects such as overfitting and vanishing gradient. These include dropout and batch normalisation layers, where some neurons are deactivated during training and batches are used in backpropagation for specific activation functions to avoid diminishing gradients.

Finally, for the model's compilation the cross entropy loss is chosen and the Adam optimiser with a fixed learning rate of 0.001. An early stopping callback with a 5 epoch patience is also app using the validation loss as the criterion. This is used to prevent overfitting and obtain the set of weights until the terminal epoch. Further hyperparameters such as epochs and batch size are chosen to be 10 and 18 respectively. To evaluate the performance of the models, we compute the loss, accuracy and f1-score on both the training and validation sets and we save them in the training history which is used to create learning curves across the number of epochs.

Image database and similarity measuring:

As mentioned above, a KD tree is used as the feature index to where the resulting feature vectors will be stored after they are extracted from the CNN. A KD tree is a binary tree at which each node on the tree has two children nodes. It recursively partitions the dataset in two equal-sized subsets, where the nodes in the left subtree should be smaller than the node in that dimension and vice versa for the right subtree [6]. In terms of how it is constructed, firstly a dimension is chosen to split the datapoints. The median in that dimension is selected as the root of the tree. Then recursively, one left and one right subtree are constructed from that root node.

For searching given a query image, after features are extracted, starting from the root node the similarity measuring is applied using Euclidean distances. Traversing the tree and computing the distance between the query image and the current node, always keeping track of the closest points. With this in regard, we can make the search algorithm more efficient by pruning the tree with nodes that have larger distances than others. The process is repeated until all points have been traversed.

Results:

A few details regarding the inspection that was done to the datasets, metrics and visualisations depicting the CNNs' model performance and some demonstration when the search function is applied between the query image and the image database.

Data analysis:

1) Sentinel_2:
With this dataset, we see that it may lead to confusion. The reason is because the "no ship" pictures do not entirely show a blank space where only the sea is visible. There are white spots which show waves, and the system may think that it is a ship.

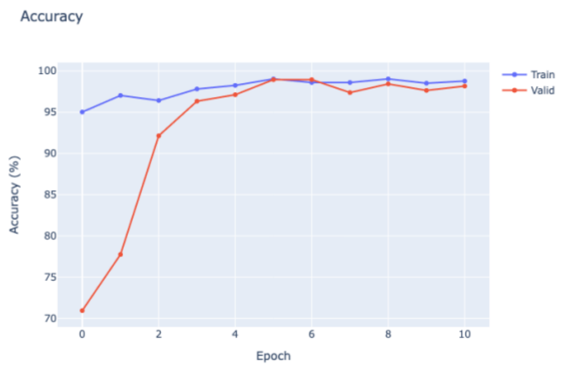
2) RESISC45:
With this one, objects are clear enough and it seems that the system can make the classification more confidently. In these set of pictures, there are multiple objects of the same object. There may be the option when the system mis-classifies bridges as ships due to the similar shape that they seem to have from that angle of the sensor.

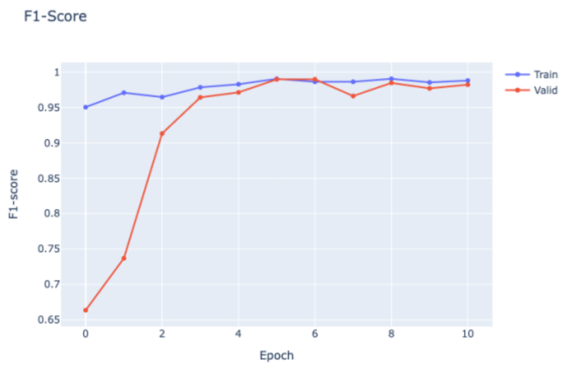
3) FAIR1M:
Images are clear enough, due to the high resolution. Neighbourhood is shown as a highly dense/occupied space that the system can easily understand. Airplanes are clear enough, again we foresee the case of multiple objects in the same picture. Likewise with the ship picture. Now, here we may have confusion between ship and airplane pictures with the neighbourhood pictures. The reason is because, in all pictures we see high density such that in ships not only the sea is visible but there is visible space on the shores, depicting land space. Likewise, with airplane pictures. The most common example is the trees or any green space that may appear in all pictures.

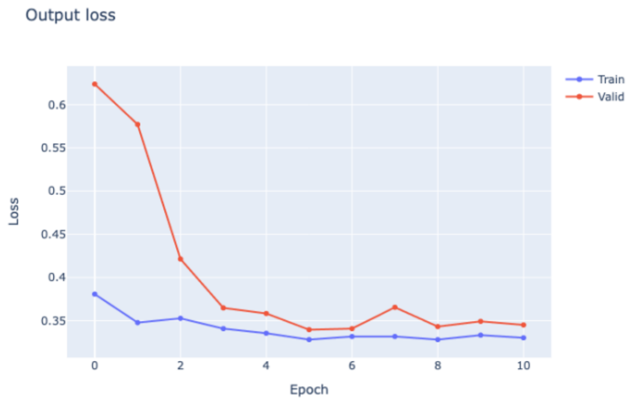
While inspecting the pixels directly it seems that there are irregularities in spatial resolution for each separate class. The length/width dimensions are inconsistent (eg. airplane and neighbourhood are 1000 and ship is 1500) as well as there is an extra channel for neighbourhood and ship pictures depicting the Alpha channel.

CNN:

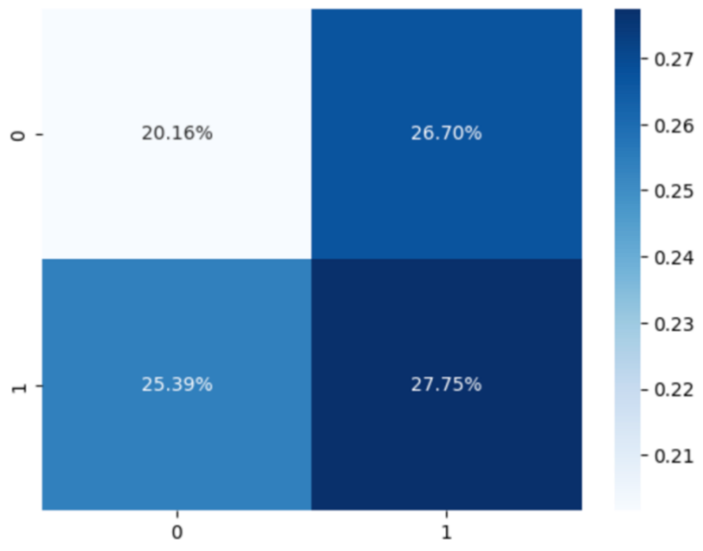
As explained above, the EfficientNet and ResNet were used to examine model performance on the three datasets. The EfficientNet on the Sentinel_2 dataset was the one to achieve the best performance on the training and validation data. The highest scores were about 0.35, above 95% and very close to 1.0 for loss, accuracy and f1-score respectively. The learning curves are shown below.







To obtain predictions on the test set, a confusion matrix is chosen generated in a heatmap form. Apart from the model's performance for the training and validation set, EfficientNet couldn't obtain more correct predictions. The respective heatmap indicates that the proportion of both correct classifications and misclassifications is equal. The heatmap is demonstrated below. That may be explained due to the facts that were stated when analysing the Sentinel_2 dataset, where in "NoShip" images the system incorrectly classifies them as "Ship" images. The waves that are shown in some pictures have similar shape to a ship. The difference may not be high enough but the proportion of incorrectly classifying "NoShip" as "Ship" is slightly higher than the proportion of "NoShip" correctly being classified.



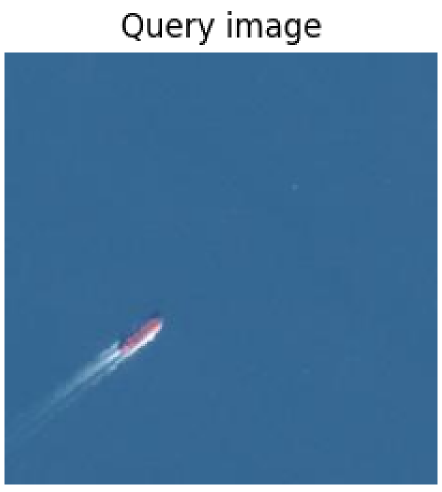
For ResNet, the overall metrics were similar for this dataset although the learning curves were not depicting smoother convergence. For several consecutive epochs, the loss remained stationary depicting that during backpropagation the weights were not updating and it can be explained that the gradient was diminishing.

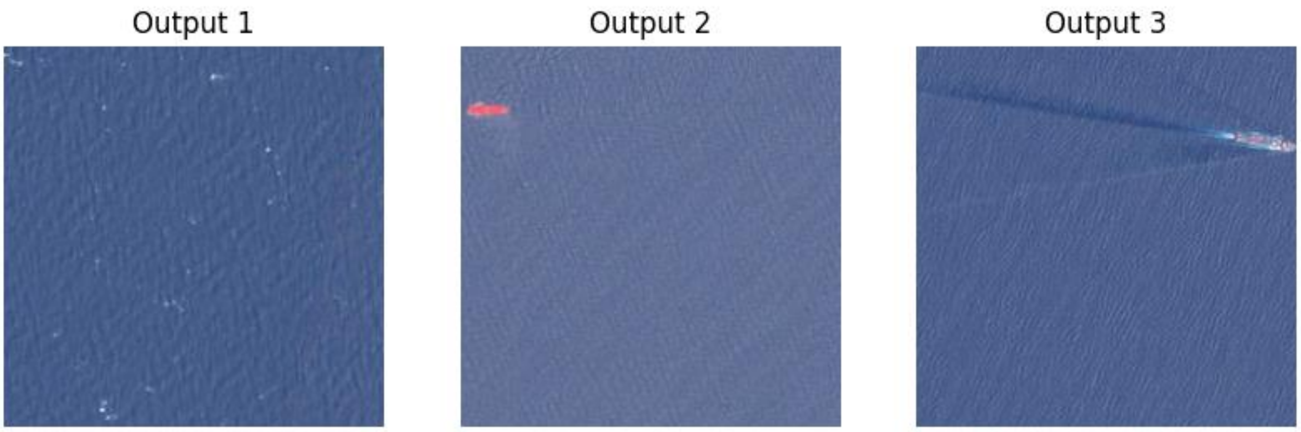
Regarding the rest of the performances of EfficientNet and ResNet on other datasets, it wasn't great enough and the proportion of misclassifications was greater than from the Sentinel_2 dataset. This may be from the fact that the number of overall classes is greater as there are 3 objects to be classified instead of 2.

KD-Tree:

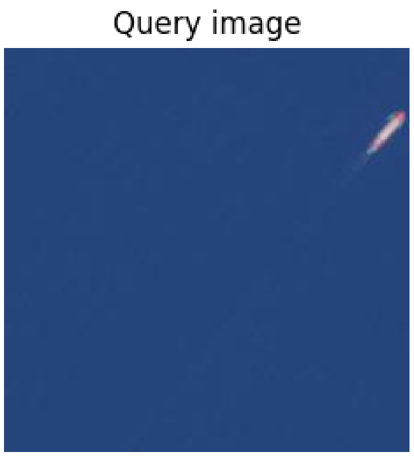
To demonstrate the effectiveness of the KD-tree, a visualisation is created with the query images and the representative top-3 candidate images. The tree was constructed with feature vectors obtained from EfficientNet's performance on the Sentinel_2 dataset. An image from this dataset is used for searching. A few examples shown below from the search performance.

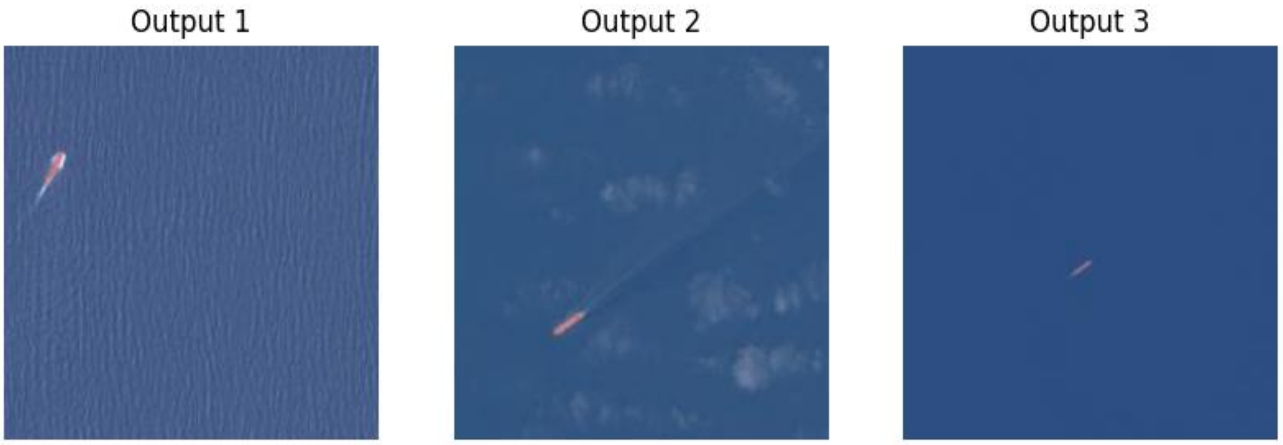
Example 1





Example 2



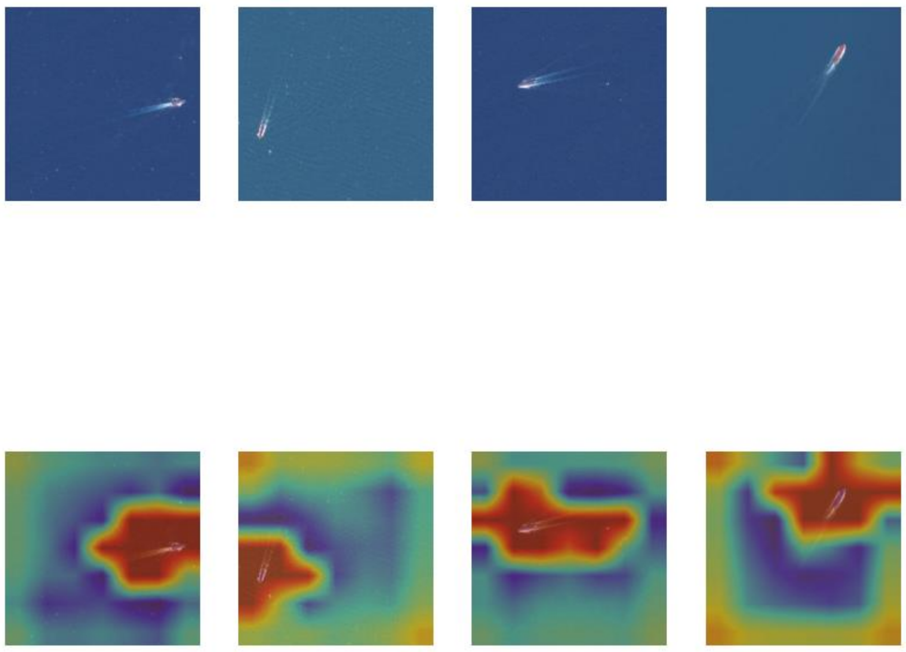


Again, due to how the model was effective on the testing set would have led in something similar regarding misclassifications. From the first example, it is shown that the search algorithm incorrectly returned a “NoShip” picture.

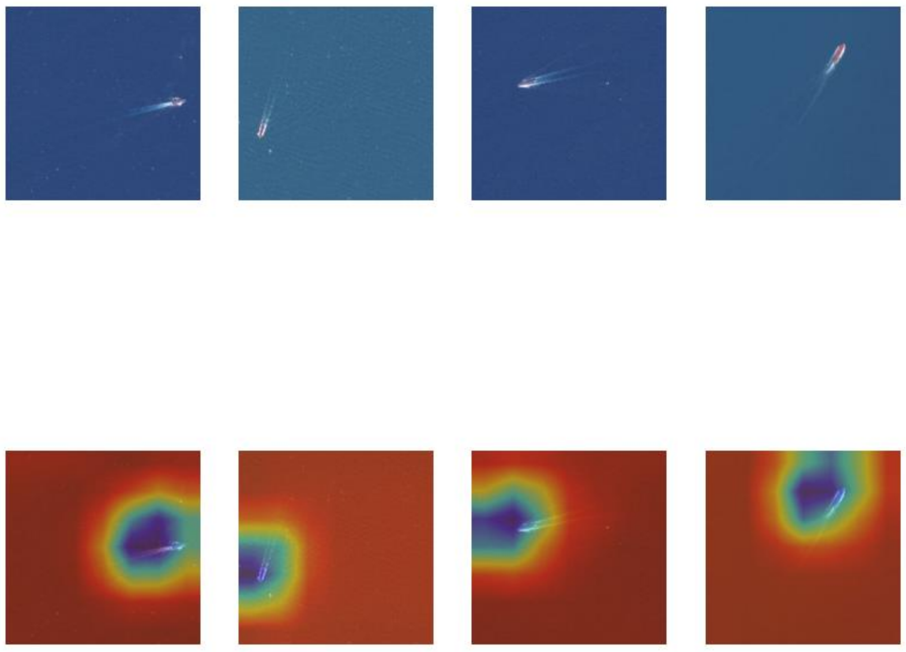
Final Remarks:

Overall, the system implementation had impact on the given datasets. Of course, there is room for improvement to be done. To demonstrate how effective the system is on query images, heatmaps are generated aiming in the detection of the object to be classified. This heatmap is known as the GradCam. Below, the two figures show a set of images from Sentinel_2 and their corresponding GradCam for EfficientNet and ResNet respectively.

EfficientNet



ResNet



The inconsistencies regarding model performance on each dataset can be explained by several reasons. Firstly, as spatial resolution was increasing in terms of length and width of the images, the performance was not great enough. The Sentinel_2, has images with the lowest resolution (224x224) and was the one with the best performance. The RESISC_45, has slightly higher resolution from Sentinel_2 images (256x256) and achieved similar metrics but its performance was more inconsistent. The FAIR_1M dataset with (1000x1000) pixel size didn't perform well neither it had consistent performance.

Secondly, the mismatch in the amounts of data that were used. For the FAIR_1M dataset, there was a total of 90 images to be examined which was way lower compared to the 2000 images that Sentinel_2 and RESISC_45. For neural networks training, it is essential to feed the network with loads of training data as it will underfit with only a few examples. For this to be addressed, a generative model can be used to create synthetic images and augment the training dataset. A second case that may lead to improvement is the testing of more pretrained models except from EfficientNet and ResNet, to examine if different architectures will indeed obtain better performances and more consistent predictions on the testing sets.

Moreover, the image database could be re-implemented using the testing data from all three datasets as well as storing features vectors that are extracted from both networks to increase its effectiveness.

For further research in the future, we can apply a combination from several approaches regarding feature extraction. This can include a pre-trained CNN as well as the methods used for specific features such as colour histograms and GLCMs. Once obtained, they will be combined in the form of a single feature vector and is stored in the image database to be retrieved using query images.

References:

[1] Bonafè, B., Botta, M., Cavagnino, D. and Pomponiu, V., 2017. Alpha Channel Fragile Watermarking for Color Image Integrity Protection. Journal of Imaging, 3(4), p.53

[2] Atlam, H.F., Attiya, G. and El-Fishawy, N., 2017. Integration of color and texture features in CBIR system. Int. J. Comput. Appl, 164(3), pp.23-29

[3] Lingadalli, R.K. and Ramesh, N., 2015. Content based image retrieval using color, shape and texture. International Advanced Research Journal in Science, Engineering and Technology, 2(6), pp.40-45

[4] Dimitrovski, I., Kitanovski, I., Panov, P., Kostovska, A., Simidjievski, N. and Kocev, D., 2023. Aitlas: Artificial intelligence toolbox for earth observation. Remote Sensing, 15(9), p.2343

[5] Zhang, X., Zhu, K., Chen, G., Tan, X., Zhang, L., Dai, F., Liao, P. and Gong, Y., 2019. Geospatial object detection on high resolution remote sensing imagery based on double multi-scale feature pyramid network. Remote Sensing, 11(7), p.755

[6] Abam, M.A., De Berg, M. and Speckmann, B., 2007, June. Kinetic kd-trees and longest-side kd-trees. In Proceedings of the twenty-third annual symposium on Computational geometry (pp. 364-372)

[7] Kumar, K.K. and Gopal, T.V., 2010, August. CBIR: Content based image retrieval. In National Conference on Recent Trends in information/Network Security (Vol. 23, p. 24th)

[8] Venters, C.C. and Cooper, M., 2000. A review of content-based image retrieval systems. JISC Technology Applications Programme, http://www.jtap.ac.uk/reports/htm/jtap-054.html