CSC5029Z – Computer Vision

Programming Project

Scott Hallauer (HLLSCO001)

Wavelet Search

1. Introduction

Content-Based Image Retrieval (CBIR) is a problem that has seen massive investment from field of Computer Vision over the past few decades. Many approaches have been proposed and refined over the years, with incremental gains in performance being achieved. Modern methods usually make use of convolutional neural networks to achieve image understanding by the automated extraction of features through supervised learning processes. However, in contrast, early approaches relied more heavily on static, mathematical representations of images. In this report, we investigate the implementation of one such mathematical approach that makes use of the Discrete Wavelet Transform. We call our system Wavelet Search.

Our results demonstrate generally good matching performance for queries using images in the dataset and worse results for those not in the dataset. Additionally, the inclusion of the original RGB colour channels in the feature vector used to calculate image similarity is shown to significantly improve query output.

2. System

2.1 Design

The system design for Wavelet Search consists of three major components: (1) Dataset Manager, (2) Query Manager and (3) User Interface. There are also some modules which are shared between these components, including a Dataset Processor, Image Processor and Image Comparator. The diagram below gives an overview of the system design and the interaction between components, where arrows indicate usage dependencies.

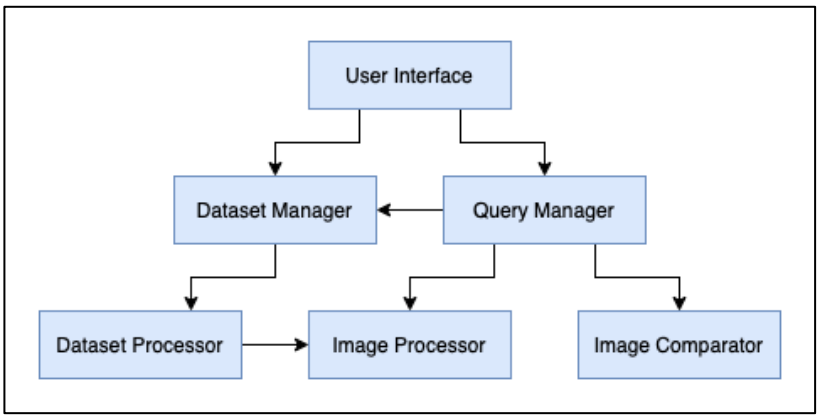

Figure 1. System design for the Wavelet Search application.

The **Dataset Manager** provides a programming interface to manage different datasets within the application. This functionality allows the user to import multiple, independent image datasets and easily switch between them for testing queries. When importing a new dataset, all image processing is done once (via the **Dataset Processor** module) through a batch copy, resize, vectorize pipeline. The resulting feature

vectors are serialised and stored in a database file, allowing for quick dataset loading on future application launches.

The **Query Manager** provides a programming interface to execute queries on a selected dataset and to view or export query results. Queries are passed in as an image path (to the query image) and a set of parameters. The parameters include various weights to be used in the comparison distance calculation as well as a *percent* and a *threshold* to improve the efficiency of the search (explained in section 2.2), and a *limit* to select how many results to display. This module also provides functionality to export the results from a query to an image.

Finally, the **User Interface** provides a user-friendly front-end to interact with the application and submit queries. The interface consists of a left-hand "Query" panel and right-hand "Results" panel. In the "Query" panel, the user can browse, select and view a desired query image, as well as set all the aforementioned query parameters via simple sliders. The currently selected dataset is also indicated in this panel. Upon submitting a search query, the results are populated in the "Results" panel arranged in rows ordered from most relevant (top left) to least relevant (bottom right). If the user hovers their mouse pointer over a result image, the calculated distance from the query image is displayed as a tooltip. The query statistics, including time taken to execute and total number of results returned, is shown at the top of the panel. An "Export" button is also displayed, allowing the user to easily save the results (along with dataset name and query parameters) to an image file.

2.2 Implementation

The Wavelet Search application is built in Python [1] using the NumPy [2], Matplotlib [3], OpenCV [4], PyWavelets [5] and PySimpleGUI [6] libraries. The Content-Based Image Retrieval (CBIR) method is based on using the Discrete Wavelet Transform (DWT) to extract a feature representation from images. This method is described in detail in the paper "Content-based image indexing and searching using Daubechies' wavelets" by Wang *et al.* [7].

All image processing is completed by the **Image Processor** module, which uses OpenCV for image manipulation and PyWavelets for extracting DWT feature vectors. When converting an image into a feature vector, the image is first resized to 128 × 128 pixels using bilinear interpolation before extracting its RGB and component colour channels. As described in the paper, the component colour space is derived from the RGB colour space and more closely resembles human perceptual colour distance (with intensity and perceived contrasts). Unlike the paper's implementation, Wavelet Search makes use of *both* the original RGB and component colour channels when matching images to improve results. These colour channels are then fed through a 5-level DWT using a Biorthogonal-1.3 wavelet, unlike the Daubechies-8 wavelet used in the paper (also in an effort to improve results). The final extracted feature vector for an image thereby consists of six 16 × 16 matrices obtained from the DWT of each of the colour channels (each consisting of four 8 × 8 sub-matrices, for a total of 24 sub-matrices), as well as the standard deviations of the first (upper-left) sub-matrix for each of the three component colour channels.

As mentioned in section 2.1, dataset processing (performed by the **Dataset Processor** module) is done once for each imported dataset and the computed image feature vectors are serialised and stored in a database file. This database is just a Python dictionary containing a list of images, where each image is represented as a dictionary with its original file name and its feature vector. Serialisation is achieved by using Python's pickle module.

Query processing is handled in a very similar way to the paper. This is managed by the **Query Manager** module in conjunction with the **Image Comparator** module. The provided query image undergoes the same resize and feature extraction workflow as described above. A three-stage comparison is then performed for each image in the database. These three stages are included to improve the efficiency of the search algorithm. The first stage is identical to that in paper, where the standard deviations of the corresponding component matrices in the query and candidate images are compared to see if they fall within a close range of each other (using the *percent* parameter to configure the size of this range). If they do not fall within the specified range, the images are marked as not matching and the system moves on to the next image. The second and third stages are almost identical to the that in the paper, besides the inclusion of RGB channels. The second stage checks if any of the differences between corresponding RGB or

component matrices exceed a certain threshold (set with the *threshold* parameter) and marks any violating images as not matching. The third stage computes a weighted Euclidean distance between the two images' matrices. This calculation includes weights for each of the image channels (six weights) and each of the sub-matrices (four weights). Final matching query results are sorted in order of increasing distance and returned to the user.

3. Analysis

3.1 Datasets

Three different datasets have been used in the testing of Wavelet Search, including the University of Washington Image Database [8], the Caltech 256 Image Dataset [9] and the Flickr30k Image Dataset [10].

1. The **University of Washington Image Database** (Washington) is the smallest dataset being tested, consisting of 963 images. These images are drawn from themes (such as "Spring flowers" and "Swiss mountains") or places (such "Italy" and "Greenland"). Although it is a small corpus, it provides a useful test bed with groups of similar photos for matching purposes. This serves as the primary dataset for experiments in this report.
2. The **Caltech 256 Image Dataset** (Caltech) consists of 30,607 images. These images are drawn from a diverse collection of 257 object categories, ranging from "blimp" to "triceratops". This dataset is useful in testing how well the DWT feature vectors generalise to matching object types in visually different photos.
3. The **Flickr30k Image Dataset** (Flickr) consists of 31,783 images. These images are randomly drawn from user-uploaded content on Flickr and have traditionally been used in testing sentence-based image description algorithms. Although this functionality is not relevant to Wavelet Search, the included images provide a much larger, more diverse collection than the Washington dataset for testing more extensive query result matching performance.

3.2 Experiments

Two sets of experiments have been run to evaluate the general matching performance of Wavelet Search and the effect of modifying certain parameters.

All experiments were executed on a 2018 MacBook Pro with the following specifications:

- **Processor:** 2.3 GHz Quad-Core Intel Core i5
- **Memory:** 8 GB 2133 MHz LPDDR3
- **Graphics:** Intel Iris Plus Graphics 655 1536 MB

Wavelet Search was tested on ten query images using the Washington dataset (Figure 2). Five of these images appear in the dataset (along with other visually similar images) and five do not appear in the dataset (although there are still other visually similar images).

These query images were submitted to the search engine with the following parameters kept constant:

$w_{1,1} = w_{1,2} = w_{2,1} = w_{2,2} = w_{c1} = 1$

$w_{c2} = w_{c3} = 2$

$percent = 50$

$threshold = 50000$

The only factor changed between experiment sets was whether or not the RGB colour channels were used in the comparison distance calculations. As such, these were the differing parameters:

Experiment #1 (-RGB): $w_r = w_g = w_b = 0$

Experiment #2 (+RGB): $w_r = w_g = w_b = 1$

The tests were run through application interface, with the first 20 results being rated for subjective relevance on a scale of 1 (completely irrelevant) to 5 (perfectly relevant). These ratings are used to calculate performance metrics (precision @ N and normalised discounted cumulative gain @ N).

Additionally, the first four results for each query have been exported for presentation in the report (Figure 3 and Figure 4).

Figure 2. Test query images for Washington dataset, where images 1-5 appear in the dataset and images 6-10 do not appear in the dataset.

3.3 Results

Table 1. Precision at 10 and 20 results for test queries on Washington dataset with and without RGB channels used.

	Precision @ 10		Precision @ 20	
	- RGB	**+ RGB**	**- RGB**	**+ RGB**
Query 1	0.70	0.70	0.65	0.65
Query 2	0.90	1.00	0.50	0.85
Query 3	0.90	1.00	0.90	0.90
Query 4	0.80	0.60	0.80	0.60
Query 5	0.40	0.80	0.50	0.60
Query 6	0.70	0.60	0.55	0.45
Query 7	0.40	0.60	0.35	0.45
Query 8	0.20	0.40	0.15	0.20
Query 9	0.20	0.60	0.30	0.60
Query 10	0.70	0.90	0.70	0.85
Average	**0.59**	**0.72**	**0.54**	**0.62**

Table 1. Normalised discounted cumulative gain (NDCG) at 10 and 20 results for test queries on Washington dataset with and without RGB channels used.

	NDCG @ 10		NDCG @ 20	
	- RGB	**+ RGB**	**- RGB**	**+ RGB**
Query 1	0.867	0.870	0.946	0.944
Query 2	0.938	0.918	0.984	0.971
Query 3	0.765	0.875	0.928	0.956
Query 4	0.816	0.768	0.927	0.904
Query 5	0.515	0.652	0.740	0.813
Query 6	0.785	0.635	0.937	0.796
Query 7	0.446	0.802	0.654	0.932
Query 8	0.210	0.652	0.373	0.652
Query 9	0.354	0.466	0.621	0.671
Query 10	0.506	0.747	0.725	0.917
Average	**0.620**	**0.739**	**0.784**	**0.856**

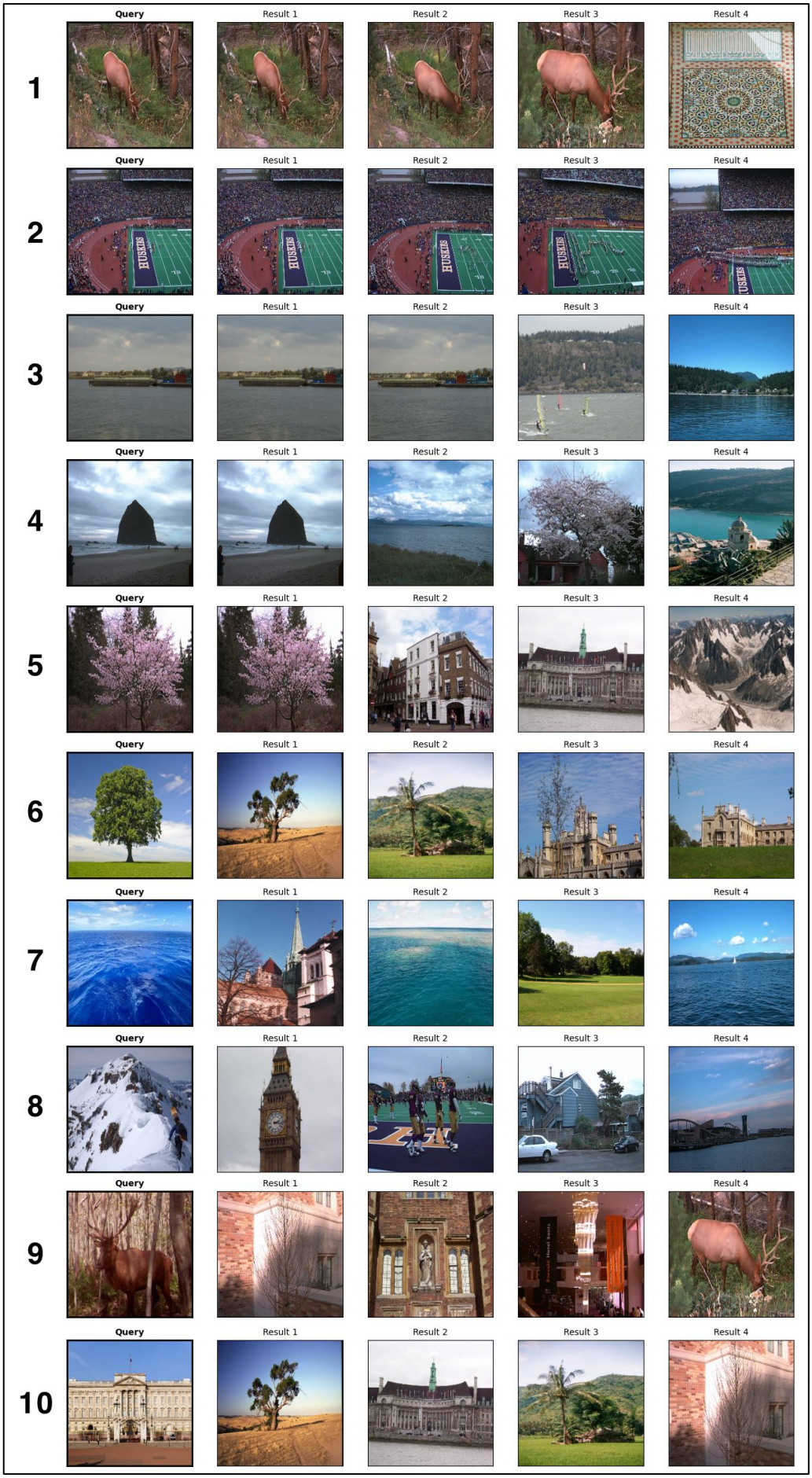

Figure 3. Test queries and results on Washington dataset (*without* RGB channels).

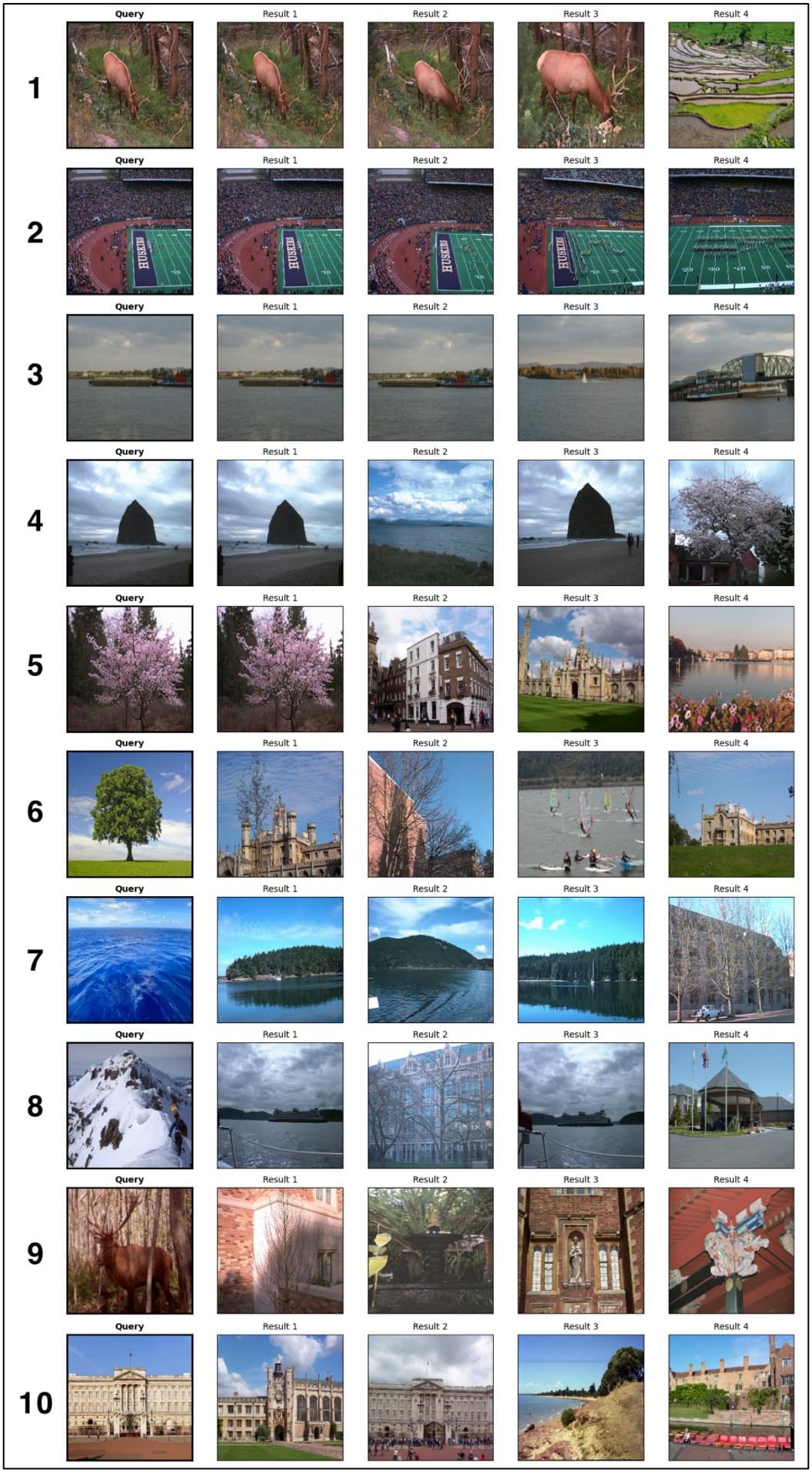

Figure 4. Test queries and results on Washington dataset (*with* RGB channels).

3.4 Discussion

As indicated in the results, the performance of Wavelet Search on our test corpus is generally good but still unpredictable. There are many examples of good, expected output results from the test queries but also some bad failure cases. We will discuss these results below.

Firstly, our experiments validate that, for images in the dataset (test queries 1-5), Wavelet Search always surfaces the exact match as the first result in the output (with a zero distance). This is a sanity test of the algorithm which ensures that the theorical approach has been implemented correctly.

Secondly, we observe a notable improvement in the performance of the algorithm when RGB colour channels are included in the comparison distance calculations. The average precision for 10 results is measured at 0.59 and 0.72 without and with RGB, respectively, and for 20 results at 0.54 and 0.62, respectively. The relative ordering of relevant results also improves when RGB is included. This is demonstrated by the increase in NDCG from 0.620 to 0.739 for 10 results and 0.784 to 0.856 for 20 results (without and with RGB, respectively). We also note a subjective improvement in the quality of results. For example, queries 7 (ocean horizon) and 10 (palace façade) both produce a completely irrelevant first result without RGB, but then produce a reasonably relevant first result with RGB.

Thirdly, we observe a reasonable subjective assessment of the first four results for most queries. The notable failure cases include queries 5, 8 and 9. For example, query 5 (cherry blossom) represents an image in the database which correctly matches for its first image, but then does not produce any relevant results for the next three. Using RGB slightly improves the results to include some flower bushes as result 4, but this is still not great given that there are many other very similar cherry blossom trees in the dataset. Through parameter tweaking, we find that we can get two more cherry blossom trees to appear in the first four results by setting $w_{1,1} = 5, w_{1,2} = 2, w_{2,1} = 1$ and $w_{2,2} = 2$. Similar alterations of parameter can improve results for the other failure cases, but this is not generalisable.

Overall, we note better matching performance for query images in the dataset (test queries 1-5) and worse performance for images not in the dataset (test queries 6-10). Furthermore, the inclusion of the RGB colour channels in the feature vector used for distance calculations produces significantly better output on average.

Conclusion

The Wavelet Search application presents a simple solution to problem of Content-Based Image Retrieval. Unlike more advanced methods, approaches using the Discrete Wavelet Transform do not require copious amounts of training data and, rather, function based on the implicit semantic information in each independent image.

Through experimentation, we have demonstrated the decent matching performance achieved using this method. However, we have also presented some notable failure cases which highlight the limitations of such a simple mathematical model. The inclusion of RGB colour channels in the vector used to compare images was shown to significantly improve result relevance. Furthermore, altering certain parameters was found to improve results on a query-by-query basis, but these changes were unfortunately not generalisable across all queries.

Possible future extensions could investigate implementations that include additional features (such as alternative colour spaces) in the vector used for image comparison. More exhaustive testing should also be performed using different permutations of query parameters to find the optimal parameters which generalise across images.

References

[1] “Python,” [Online]. Available: https://www.python.org. [Accessed 4 November 2021].

[2] “NumPy,” [Online]. Available: https://numpy.org. [Accessed 4 November 2021].

[3] “Matplotlib,” [Online]. Available: https://matplotlib.org. [Accessed 4 November 2021].

[4] “OpenCV,” [Online]. Available: https://opencv.org. [Accessed 4 November 2021].

[5] “PyWavelets,” [Online]. Available: https://pywavelets.readthedocs.io. [Accessed 4 November 2021].

[6] “PySimpleGUI,” [Online]. Available: https://pysimplegui.readthedocs.io. [Accessed 4 November 2021].

[7] J. Z. Wang, G. Wiederhold, O. Firschein and S. X. Wei, “Content-based image indexing and searching using Daubechies' wavelets,” *International Journal on Digital Libraries,* pp. 311-328, 1997.

[8] University of Washington, “Object and Concept Recognition for Content-Based Image Retrieval,” [Online]. Available: http://imagedatabase.cs.washington.edu. [Accessed 4 November 2021].

[9] J. Li, “Caltech 256 Image Dataset,” [Online]. Available: https://www.kaggle.com/jessicali9530/caltech256. [Accessed 4 November 2021].

[10] Hsankesara, “Flickr Image Dataset,” [Online]. Available: https://www.kaggle.com/hsankesara/flickr-image-dataset. [Accessed 4 Novemeber 2021].