EEE3032

Computer Vision & Pattern Recognition

Coursework Assignment

By: Dhruvraj Singh Rawat

Table Of Content

Abstract

This report investigates the implementation and evaluation of various visual search techniques, including global colour histograms, spatial grid-based descriptors, and bag-of-visual-words (BoVW). The core objective is to effectively retrieve images based on visual similarity.

Global colour histograms were employed as a baseline, with quantization levels explored to optimize performance. Spatial grid-based descriptors were implemented, combining colour and texture features from localized regions. BoVW was used to represent images as distributions of visual words, enabling efficient retrieval.

The evaluation used precision-recall curves and area under-the-curve (AUC) metrics. The impact of quantization levels, distance measures, and dimensionality reduction techniques (PCA) on retrieval performance was also analyzed.

Experimental results demonstrated the effectiveness of spatial grid-based descriptors, particularly when combined with angular quantization. BoVW also exhibited strong performance, highlighting its potential for robust image retrieval. Future work will delve deeper into feature engineering, exploring advanced descriptors and distance measures to further enhance retrieval accuracy.

Overview of Implemented Visual Search Techniques

Requirement 1: Global Colour Histogram

The global colour histogram is implemented in the `create_global_color_hist` function, which iterates through each BMP image in the dataset to compute and save a colour histogram descriptor. Each image is normalized between 0 and 1 and is passed to the `colour_histogram` function, where I perform the core histogram computation. Firstly, each pixel's RGB values are scaled to the range [0, Q−1] through multiplication by Q, flooring, and clipping to ensure the values stay within the expected range. Then, the quantized image is divided into three separate colour channels: `red_channel`, `green_channel`, and `blue_channel.`

For each channel, a frequency histogram of Q bins is computed using `np.bincount`, which counts occurrences of each quantized colour value across the channel. After generating these histograms, it is combined into a single descriptor vector called `overall_bin`. This is achieved by weighting each channel's histogram by powers of Q to ensure unique encoding for each colour combination. These encodings are then saved in the HDD.

The colour histogram descriptors are then loaded using the `load_descriptors` function, which returns two arrays: `ALLFILES` and `COLOR_HIST_ALLFEAT`. `COLOR_HIST_ALLFEAT` contains the colour histogram descriptor for each image, while `ALLFILES` stores the actual image paths of these images. The index in each array corresponds to a specific image, linking each descriptor with its respective image path. Next, a random query image is selected, and the **Euclidean distance (L2 norm)** between the query image's descriptor and the descriptors of all other images is computed using the `cvpr_compare` function. The top 15 images with the closest distance to the query image are then returned, representing the most similar matches based on colour histogram distribution.

Requirement 2: Evaluation Methodology: Precision-Recall Curve

To determine if relevant images are retrieved, I first extract the class label of each image based on the increasing distance from the query image. I extract the class of each image using the `extract_class_and_file` function from the `cvpr_toolset` module, which extracts the class using the image file path. These class labels are stored in the `response_class_no` list, where each element represents the class of an image in order of similarity to the query. Since the Query image would have 0 distance with itself, it would always be at the 0th index in the `response_class_no` list, the class at the 0th index is assigned to `query_class` and entries are stored in `response_class.`

The `plot_precision_recall_curve` function is designed to calculate and display the precision-recall (PR) curve for evaluating the performance of the image retrieval system. In all the analysis I have used the entire dataset of 591 images to calculate the PR curve, thus ensuring that both precision and recall values range from 0 to 1.

Within `plot_precision_recall_curve`, the function begins by initializing two lists, `precision` and `recall`, to store these values at each retrieval step. It also tracks `relevant_retrieved`, which counts the number of relevant images retrieved so far, and calculates `total_relevant` which contains the total count of relevant images in the dataset that match the query class.

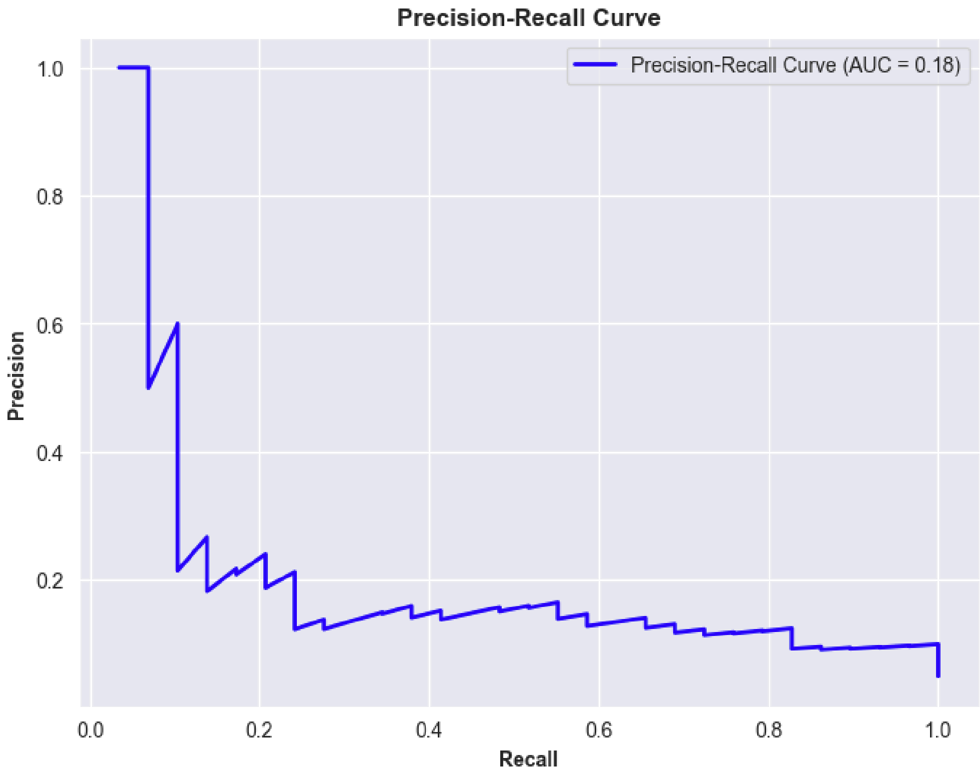


Image 1: Precision-Recall Curve obtained by using “IMG 7_9_s.bmp” as Query Image. Global Colour Histogram as Descriptor with a Quantization Level of 8

As the function iterates through each retrieved image in `response_class` (which contains the classes of images ordered by increasing distance from the query), it checks whether the class of the current image matches the query's class. If it does, `relevant_retrieved` is incremented, contributing to the relevant retrievals at that point. The current precision and recall values are then calculated:

- **Precision** at each step is computed as the ratio of `relevant_retrieved` to the total number of images considered so far (i.e., `i + 1`).
- **Recall** at each step is calculated as the ratio of `relevant_retrieved` to `total_relevant`, providing the fraction of relevant images retrieved relative to all relevant images in the dataset.

These values are appended to the `precision` and `recall` lists, building a detailed view of the retrieval performance as more images are examined.

Finally, I also implemented the method to calculate the Area Under Curve (AUC) for each PR curve using the `auc` function from `sklearn.metrics`, providing a single numerical score that summarizes the overall retrieval effectiveness.

Requirement 3: Spatial Grid (Color and Texture)

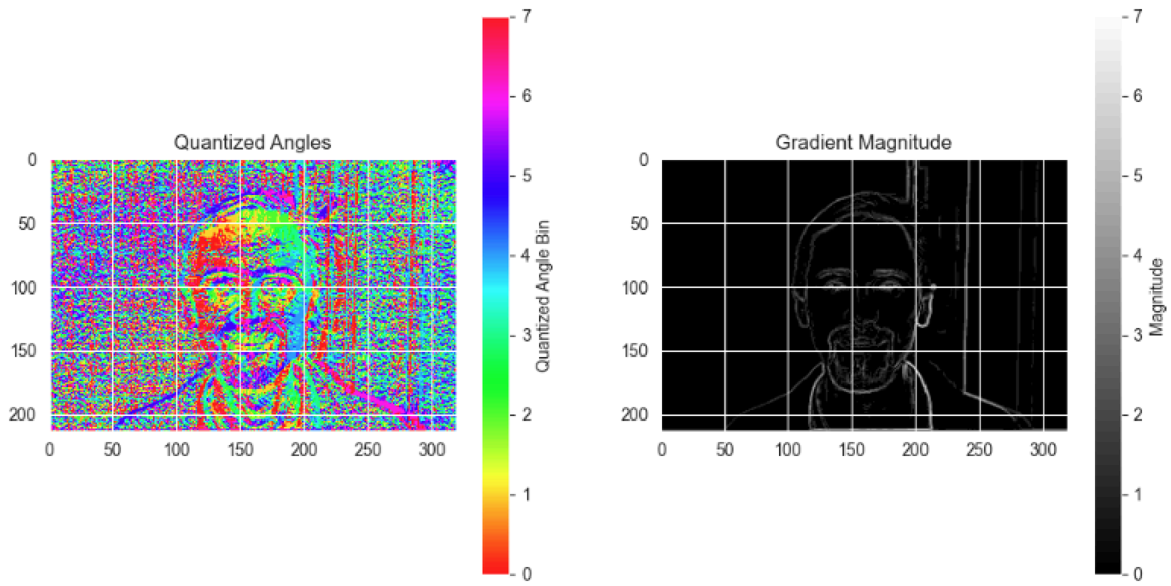


Image 2: This image shows the result of a Sobel filter applied to a sample image, displaying quantized angle orientations and gradient magnitudes. The angle orientations and gradient magnitudes are categorized into 8 levels

In this section, I implement gridding to extract detailed colour and texture features from localized regions within each image and then combine them into a comprehensive descriptor. The process is handled by the `color_texture_grid_descriptor` function, which divides each image into multiple cells and computes colour histograms and Sobel-based texture descriptors for each cell. This gridded approach allows for capturing local variations in colour and edge orientation, creating a more robust and expressive descriptor.

In `color_texture_grid_descriptor`, each image is split into mentioned grid_size (eg 8x8 grid), and descriptors are calculated for each cell. For colour features, the `colour_histogram` function generates a colour histogram by quantizing pixel intensities into Q bins per channel.

Texture features are extracted using the Sobel filter, with gradient orientation and magnitude quantized to create a histogram of edge features. This is performed in the `sobel_quantization` function, where Sobel gradients are computed in both x and y directions, yielding gradient `magnitude` and `angle` matrices. The angle matrix is normalized to the range of 0 to 360 degrees, then quantized into `num_bins` angular bins. Each angle is assigned to an angular bin, capturing the edge direction distribution within the cell. Similarly, the magnitudes are normalized to [0,1] and then quantized into Q bins, forming a histogram of edge strengths.

Requirement 4: PCA and Mahalanobis Distance

Principal Component Analysis (PCA) is used to reduce the dimensionality of the image descriptors, allowing for a more efficient and computationally manageable representation. I applied `PCA` from `sklearn.decomposition` to the image descriptors, with the goal of retaining 90% of the variance in the data. This is achieved by specifying `n_components = 0.9`.

After transforming the descriptors into the lower-dimensional space using PCA, in the next step, I computed the Mahalanobis distance between the query image's descriptor and all other image descriptors. Unlike the Euclidean distance, which treats all dimensions equally, the Mahalanobis distance takes into account the covariance structure of the data. The covariance matrix of the reduced descriptors is computed based on all the image descriptors and its inverse is calculated using `np.linalg.inv`, which is then used in the Mahalanobis distance formula.

Now like in the previous section for each image, the Mahalanobis distance to the query image is calculated, and the images are sorted by their distance to the query. The sorted distances are then used to display the top 15 most similar images to the query image.

Requirement 5: Additional descriptors and Distance measures

I explored different descriptors and distance measures to evaluate their effect on performance. I experimented with additional texture and colour-based descriptors such as Gabor and Haralick and evaluated the impact of various distance metrics like L1 norm, Chi-squared, and Cosine similarity on the retrieval results.

The **Gabor descriptor** was calculated by applying Gabor filters with varying orientations and frequencies to the image. I used OpenCV (`cv2`) to apply the Gabor filters and compute the filtered image for each orientation. The `gabor_descriptor` function uses four different orientations (0°, 45°, 90°, and 135°) to capture texture information at different scales and directions. The filtered images are processed, and the average value of each filtered image is used as the feature representation.

Additionally, I explored **Haralick's features** to capture the statistical properties of texture. I wrote the `haralick_features` function, in which I used `skimage.feature` library computes the Gray-Level Co-occurrence Matrix (GLCM) of the image at various angles and distances, and extracts properties such as contrast, correlation, and energy. These features are then concatenated into a single feature vector. Haralick features are commonly used to characterize the spatial arrangement of pixels in texture analysis.

For the **distance measures**, I explored multiple alternatives to the standard Euclidean distance. The **L1 norm** (Manhattan distance) was tested. The **Chi-squared distance** was also considered, as it is particularly effective for comparing histograms. Lastly, I experimented with **Cosine similarity**, which measures the cosine of the angle between two vectors, often used when the magnitude of the vectors is less important than their direction.

Requirement 6: Bag of Visual Words Retrieval

For the BoVW-based image retrieval system, I coded the following functions `extract_sift_features`, `create_codebook`, `compute_bovw_histogram`, `compute_bovw_representation`, and `run_bovw_system`.

The process begins with **feature extraction** using the function `extract_sift_features`, which uses the SIFT detector. This function identifies key points and descriptors for each image, creating robust local features that are invariant to scale and rotation. For each input image, SIFT extracts a set of distinctive 128-dimensional feature vectors, which collectively represent visually informative regions.

The retrieval process begins with **feature extraction** using the `extract_sift_features` function, which applies the Scale-Invariant Feature Transform (SIFT) to detect and describe key points in each image. For each input image, this function generates a set of 128-dimensional SIFT descriptors, capturing key visual details that are invariant to transformations like scale and rotation. These descriptors form the foundational data for BoVW, representing visually significant regions across the image set.

Once the descriptors are extracted, the **codebook generation** step, handled by the `create_codebook` function, organizes these local descriptors into clusters that form the BoVW vocabulary. In this step, all descriptors from the dataset are concatenated and clustered using the k-means algorithm, with each cluster representing a unique visual pattern. The `n_clusters` parameter defines the vocabulary size or the number of "visual words." The cluster centres then serve as the standardized codebook, allowing any image to be represented consistently through a set of common visual words.

To quantify each image's unique distribution of visual words, the **BoVW histogram** is computed via the `compute_bovw_histogram` function. For each image, its descriptors are mapped to the closest codebook clusters, and a histogram is constructed based on the frequency of each visual word. This histogram is normalized to ensure a uniform scale across images, resulting in a vectorized representation of each image based on its visual word distribution.

The **primary function** **`run_bovw_system`** then triggers the entire BoVW process, from loading the images to generating the BoVW histograms. This function first calls `extract_sift_features` to gather SIFT descriptors for each image. It then applies `create_codebook` to establish a visual vocabulary by clustering all descriptors into visual words. Finally, it computes BoVW histograms for each image by calling `compute_bovw_histogram` within `compute_bovw_representation`. The output is a BoVW matrix where each row represents an image's visual word distribution, ready for image retrieval tasks.

Experimental Results, Observations & Proposed Future Experiments

Please note that, unless stated otherwise, the image "IMG 7_9_s.bmp" is taken as the query image for all the experiment results mentioned below.

Requirement 1: Global Colour Histogram

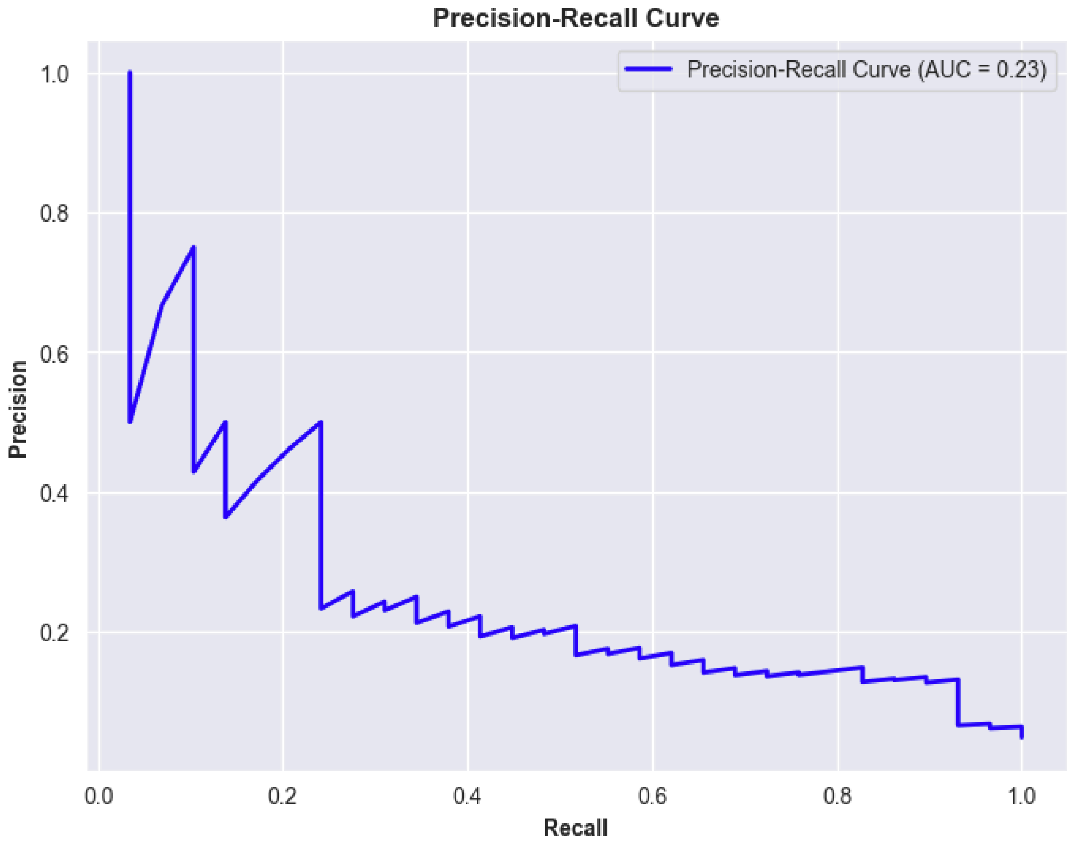
Experiment 1.1: Compute the Color Histogram for different quantisation levels and identify the level at which the PR curve's maximum AUC (Area Under the Curve) is achieved.

Level of Quantization	Descriptor	Distance Measure	Performance Measure: AUC
2	COLOR_HIST	L2	0.076
4	COLOR_HIST	L2	0.090
6	COLOR_HIST	L2	0.155
8	COLOR_HIST	L2	0.180
10	COLOR_HIST	L2	0.172
12	COLOR_HIST	L2	0.183
14	COLOR_HIST	L2	0.206
16	COLOR_HIST	L2	0.219
18	COLOR_HIST	L2	0.211
20	COLOR_HIST	L2	0.224
22	COLOR_HIST	L2	0.223
24	COLOR_HIST	L2	0.229
26	COLOR_HIST	L2	0.234
28	COLOR_HIST	L2	0.232
30	COLOR_HIST	L2	0.231

Table 1

1. The performance (AUC) of the colour histogram descriptor improves steadily as the quantization level (Q) increases, indicating that higher levels of quantization capture more discriminative colour information.

2. The AUC reaches its peak at Q=26 with a value of 0.234, suggesting that low quantization levels may lose critical colour details, while at a higher level of Quantization, the performance stabilizes.



The above Precision-Recall Curve corresponds to the optimal quantization value, Q = 26, as identified in the previous table.

Experiment 1.2: Determining the Class in Which the Colour Histogram Performs Best Based on the Optimal Level of Quantization Found in the Previous Step (Q = 26)

Q	Class	Descriptor	Distance Measure	Performance Measure: AUC
26	1	COLOR_HIST	L2	0.145
26	2	COLOR_HIST	L2	0.047
26	3	COLOR_HIST	L2	0.037
26	4	COLOR_HIST	L2	0.037
26	5	COLOR_HIST	L2	0.052
26	6	COLOR_HIST	L2	0.132
26	7	COLOR_HIST	L2	0.159
26	8	COLOR_HIST	L2	0.157
26	9	COLOR_HIST	L2	0.070
26	10	COLOR_HIST	L2	0.103
26	11	COLOR_HIST	L2	0.034
26	12	COLOR_HIST	L2	0.039

26	**13**	**COLOR_HIST**	**L2**	**0.226**
26	14	COLOR_HIST	L2	0.089
26	15	COLOR_HIST	L2	0.030
26	16	COLOR_HIST	L2	0.056
26	17	COLOR_HIST	L2	0.053
26	18	COLOR_HIST	L2	0.051
26	19	COLOR_HIST	L2	0.102
26	20	COLOR_HIST	L2	0.182

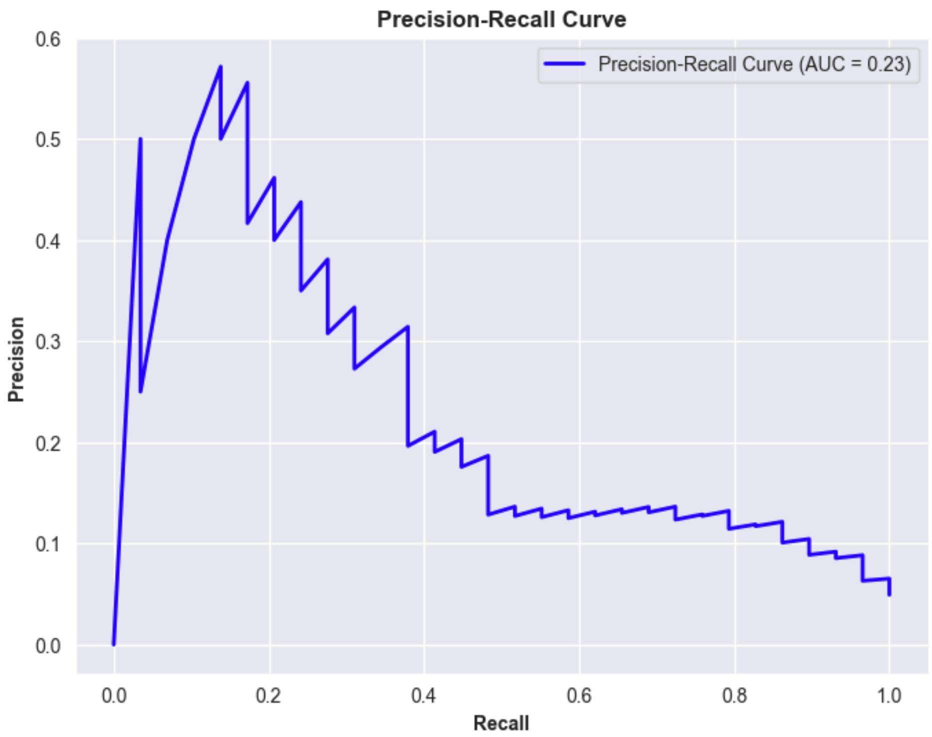
Table 2

In the above table, the performance (Area under the PR Curve - AUC) achieved for each class is shown by using the first item of each class as the query image. In this experiment, the value of Q was kept constant at 26, as determined in the previous step. The following observation can be drawn:

1. Performance is Item Dependent: In the previous experiment, using the 9th image as a query image of Class 7 resulted in an AUC value of 0.234, whereas in this case, the AUC dropped to 0.159 when we used the first image. This indicates that the choice of the query image within a class significantly impacts the overall performance.

2. Based on the table above, the colour histogram-based visual search algorithm performs well for classes 7, 8, 13, and 20 because the dominant colour distribution in these images is distinctive and consistent within their respective classes. The vibrant red car (Class 7), the teal bicycle (Class 8), the colourful book (Class 13), and the blue-green ocean waves (Class 20) all show unique colour patterns, making them easily distinguishable in the feature space. Thus we can conclude that colour histograms are effective for image classes with strong and characteristic colour signatures.

The above image collage displays the first image from each class where the Global Colour Histogram achieved the best performance, providing an overview of representative images for each class.



This image represents the Precision-Recall (PR) curve for the best-performing class, class 13, as identified in Table 2. The curve was generated using the Global Colour Histogram with a Quantization Level of Q = 26.

The image illustrates the top 15 results retrieved using the 1st image of class 13 as a query image, employing the Global Colour Histogram with Quantization Level Q = 26. All the returned images share a similar colour composition and variety like the query image.

Additional Experiments that could be implemented involve iteratively determining the optimal quantization value for each class, rather than applying the best-performing quantization value identified for a single class to all others. This approach would allow us to observe whether a smaller variety of colours within a specific image class impacts the optimal quantization value.

Requirement 3: Spatial Gridding

Experiment 3.1: Gridding-based experiment with Colour and Textual Features

In this experiment, each image is divided into **4×4 grids**, and the **Colour Histogram**, **Angular Quantization**, and **Magnitude Quantization** are computed for each grid individually. The descriptors from all the grid components are then combined to form an overall image descriptor.

For this experiment:

- **The Colour Quantization Level** is set to **26**, as determined to be optimal in a previous experiment.
- **Angular and Magnitude Quantization Levels** are set to **8**.

The following combinations of descriptors were tested, and the **Precision-Recall (PR) Curve** and **AUC** were computed for each:

1. Fig 1: Global Colour Histogram combined with Textual Descriptors (Angular and Magnitude).
2. Fig 2: Global Colour Histogram combined with Angular Quantization-based Descriptor.
3. Fig 3: Global Colour Histogram combined with Magnitude Quantization-based Descriptor.

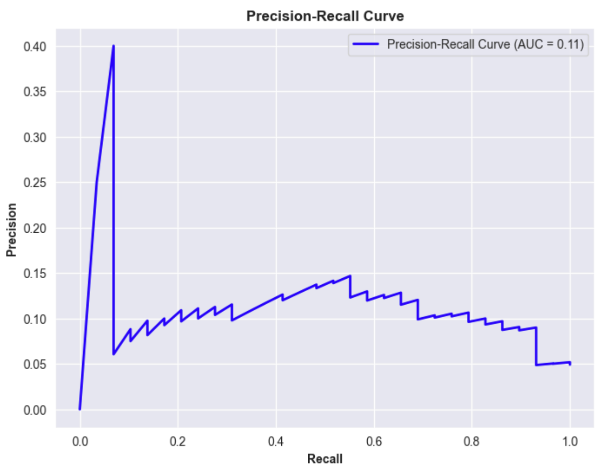


Fig 1

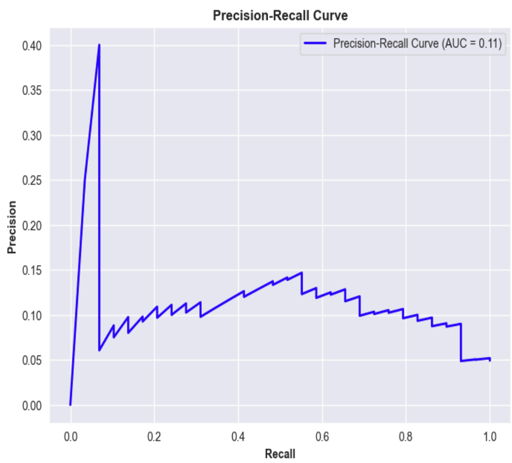


Fig 2

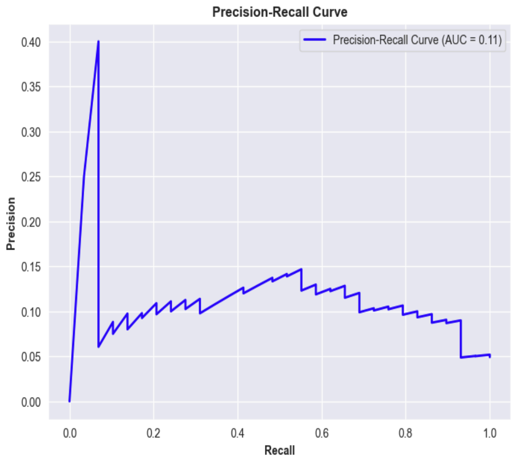


Fig 3

As observed in Figures 1, 2, and 3, the similar performance across all three descriptor combinations can be explained by several factors. **One possible reason** is the dominance of the Global Color Histogram, which may overshadow the contributions of the Angular and Magnitude features. This dominance likely explains why the inclusion of different textual features (Angular and Magnitude) has a negligible impact on overall performance. **Additionally,** the quantization levels chosen for the Angular and Magnitude descriptors (set to 8) may not be fine-grained enough to capture meaningful variations within the dataset, reducing their discriminative effectiveness. **Furthermore**, the Angular and Magnitude descriptors could be encoding redundant or highly correlated information when paired with the Global Color Histogram, neutralizing their individual contributions. This redundancy might account for the identical Precision-Recall curves and AUC values observed across all three combinations.

Experiment 3.2 - Experiment with Angular Quantization: Compute the Angular Quantisation for different levels of quantization and identify the level at which the maximum AUC (Area Under the Curve) for the Precision Recall curve is achieved.

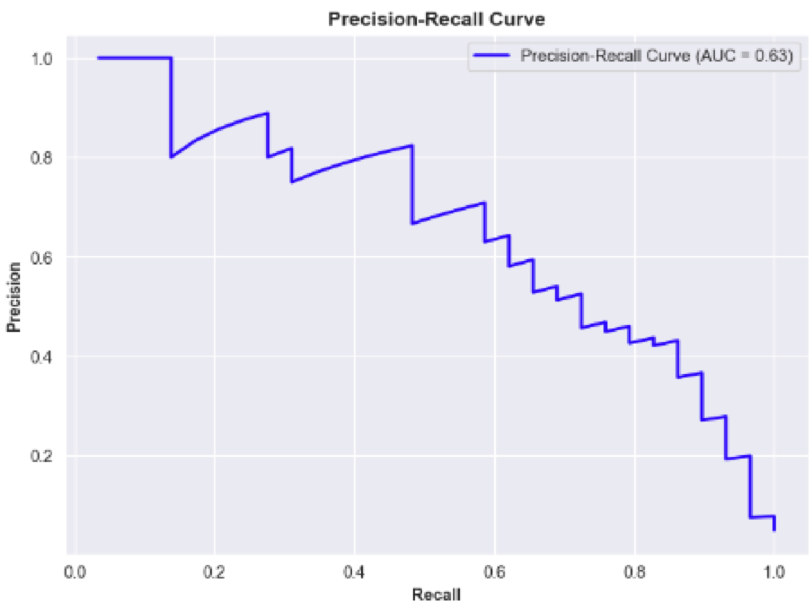
Angular Quantization	Descriptor	Distance Measure	Performance Measure: AUC
8	ANG_QUANTISATION	L2	0.160
16	ANG_QUANTISATION	L2	0.491
24	ANG_QUANTISATION	L2	0.556
32	ANG_QUANTISATION	L2	0.606
40	ANG_QUANTISATION	L2	0.609
48	ANG_QUANTISATION	L2	0.606
56	ANG_QUANTISATION	L2	0.592
64	ANG_QUANTISATION	L2	0.619
72	ANG_QUANTISATION	L2	0.614
80	ANG_QUANTISATION	L2	0.626
88	ANG_QUANTISATION	L2	0.615
96	ANG_QUANTISATION	L2	0.627

AUC Performance Across Varying Angular Quantization Levels

As observed from the table above, the performance begins to plateau at a quantization level of 32, with further increases resulting in only minimal improvements. This indicates that higher quantization levels provide diminishing returns beyond this point.

However, it is important to note that the optimal quantization level MIGHT be query image specific. For instance, in this experiment, the class used cars, which likely benefited from the chosen level of angular quantization. If we were to use a different class, such as water or textures with less distinct angular features, the optimal quantization level might vary significantly. Thus similar experiments can be done by choosing query images from other classes to justify this hypothesis.

Top 15 Responses achieved using the Best Performing Angular Quantization of 96



PR Curve for Angular Quantization of 96

The Additional Experiments that could have been performed are :

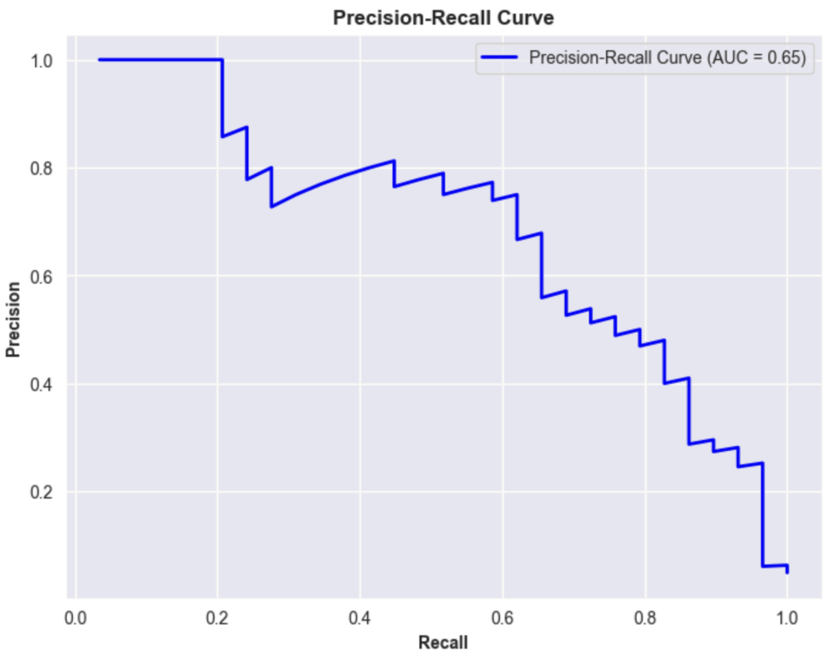
1. Experimenting with query images from different classes to determine if the **optimal level of quantization is image-specific**: Does the optimal quantization level vary significantly if the image has more **distinct angular features**?

2. **Balance Descriptor Contributions:** Adjust the weighting of the Global Color Histogram and textual descriptors to ensure that the influence of Angular and Magnitude features is not overshadowed, allowing for a more balanced contribution to the combined descriptor.

3. **Investigate the Impact of Grid Dimensions:** Explore how varying grid dimensions (e.g., 2×2, 8×8) affect the performance of individual descriptors, such as Global Color Histogram, Angular Quantization, and Magnitude Quantization, in image search tasks.

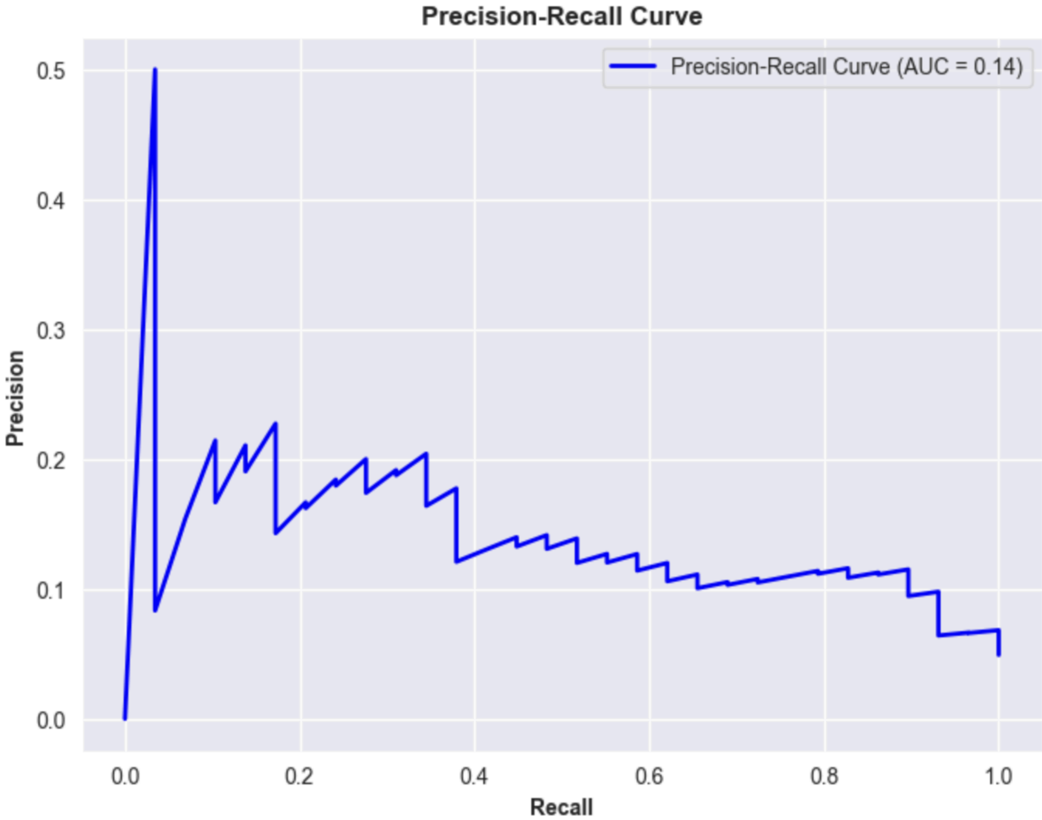
Requirement 4: Use of PCA

Experiment 4.1: PCA and Mahalanobis Distance Applied to the Best-Performing Angular Quantization (Q = 96)



The above PR Curve is obtained after applying PCA and using the Mahalanobis distance measure on the best-performing angular descriptor from Experiment 3.2. Using PCA, we reduced the dimensionality from 96 to 4 while retaining 90% of the variance in the data. The Mahalanobis distance was then used to compute the similarity between the query and other images to identify similar ones. Based on the PR Curve from Experiment 3.2 and the mentioned above, it is evident that applying PCA and the Mahalanobis distance has improved the performance. The AUC of the PR Curve increased from 0.63 to 0.65.

Experiment 4.2: PCA and Mahalanobis Distance Applied to the Grid Colour Angular Magnitude-based descriptor.



Like in the previous experiment, in this experiment, I used the descriptor generated in Experiment 3.1, which combines colour and textual features (angular and magnitude). After applying PCA, the dimensionality of the dataset was reduced from 672 to 28 while preserving 90%of variance. This dimensionality reduction resulted in an overall performance improvement, with the AUC increasing from 0.11 to 0.14, demonstrating the effectiveness of PCA in enhancing the descriptor's performance.

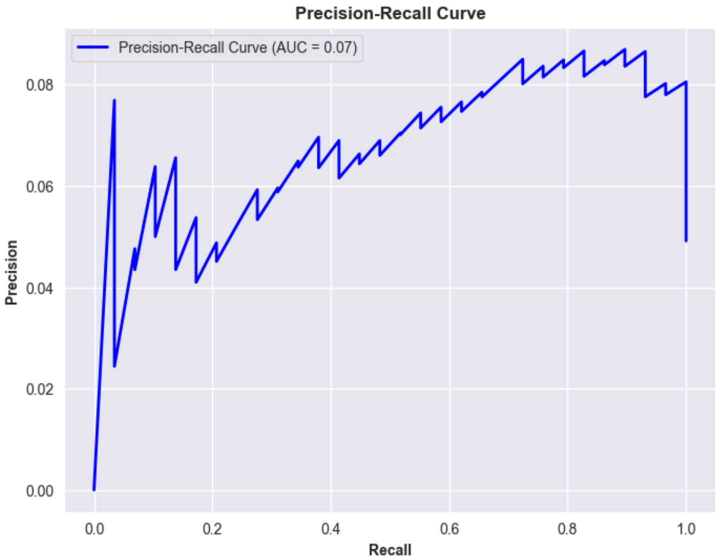
Additional Experiments That Could Be Explored

1. **Evaluating Mahalanobis Distance Across Quantization Levels**: Investigate whether using only the Mahalanobis distance as a distance measure leads to better results at specific quantization levels. Additionally, compare its performance relative to other distance measures to determine if it provides superior results at lower or higher levels of quantization.
2. **Applying PCA Across All Descriptors**: Apply PCA to all the descriptors generated so far to evaluate its impact. Identify cases where PCA does not improve performance, providing insights into which descriptors are inherently robust or lose critical information during dimensionality reduction.

Requirement 5: Different descriptors and distance measure

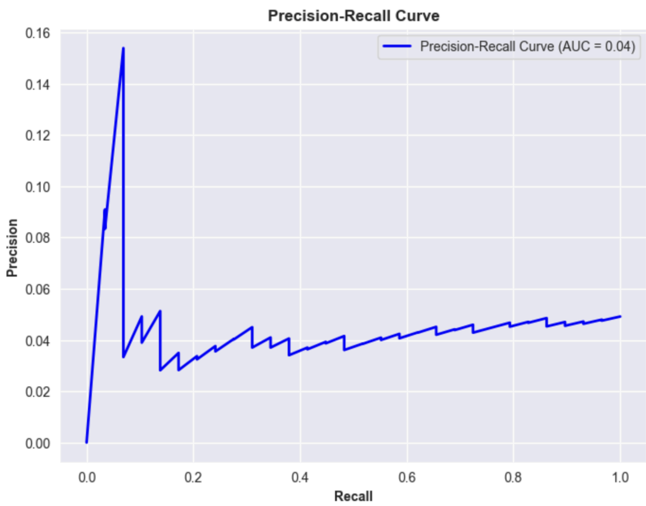
Exp 5.1: Experimenting with Different Descriptors

a) **Gabor Filter :**



The above PR curve is obtained for the Gabor filter with 4 orientations. The performance achieved with the Gabor filter is comparable to that of the Global Color Histogram (refer to the table in Experiment 1.1) at a quantization level of 4. Further experiments could involve iteratively testing different numbers of orientations to identify the optimal value that maximizes the AUC of the PR curve, thereby improving performance.

b) **Haralick Filter :**



The above PR curve is obtained using the Haralick filter with 256 grey levels. At this level of quantization, the performance of the Haralick filter is worse than that of the Gabor filter with 4 orientations. Further experiments are required to determine the optimal number of grey levels that can maximize the AUC, thereby improving the performance of the Haralick filter.

Exp 5.2: Experimenting with Different Distance Measures

Descriptor	Quantization	Distance Measure	Performance Measure: AUC
HARALICK	256 gray levels	L1	0.169
HARALICK	256 gray levels	L2	0.176
HARALICK	256 gray levels	Chi-Squared	0.162
HARALICK	256 gray levels	Cosine	0.126
GABOR	4 orientations	L1	0.062
GABOR	4 orientations	L2	0.067
GABOR	4 orientations	Chi-Squared	0.068
GABOR	4 orientations	Cosine	0.146
COLOR_HIST	26	L1	0.267
COLOR_HIST	26	L2	0.234
COLOR_HIST	26	Chi-Squared	0.286
COLOR_HIST	26	Cosine	0.173
GRID_COLOR_TEXT	Grid: 4x4 | Colour bin : 8 | Ang + Mag Bin : 8	L1	0.162
GRID_COLOR_TEXT	Grid: 4x4 | Colour bin : 8 | Ang + Mag Bin : 8	L2	0.113
GRID_COLOR_TEXT	Grid: 4x4 | Colour bin : 8 | Ang + Mag Bin : 8	Chi-Squared	0.149
GRID_COLOR_TEXT	Grid: 4x4 | Colour bin : 8 | Ang + Mag Bin : 8	Cosine	0.102
ANGULAR_HIST	96	L1	0.625
ANGULAR_HIST	96	L2	0.627
ANGULAR_HIST	96	Chi-Squared	0.665
ANGULAR_HIST	96	Cosine	0.613
GRID_COL_ANG_HIST	Grid: 4x4 | Colour bin : 8 | Ang Bin : 8	L1	0.158
GRID_COL_ANG_HIST	Grid: 4x4 | Colour bin : 8 | Ang Bin : 8	L2	0.113
GRID_COL_ANG_HIST	Grid: 4x4 | Colour bin : 8 | Ang Bin : 8	Chi-Squared	0.146
GRID_COL_ANG_HIST	Grid: 4x4 | Colour bin : 8 | Ang Bin : 8	Cosine	0.102
GRID_COL_MAG_HIST	Grid: 4x4 | Colour bin : 8 | Mag Bin : 8	L1	0.162
GRID_COL_MAG_HIST	Grid: 4x4 | Colour bin : 8 | Mag Bin : 8	L2	0.113
GRID_COL_MAG_HIST	Grid: 4x4 | Colour bin : 8 | Mag Bin : 8	Chi-Squared	0.149
GRID_COL_MAG_HIST	Grid: 4x4 | Colour bin : 8 | Mag Bin : 8	Cosine	0.102

Using the descriptors generated in previous requirements, I recalculated their performance at specific quantization levels with different distance measures to observe how the results vary. For this experiment, I considered additional distance measures, which include: L1, Cosine, and Chi-Squared.

Based on the table above following are the observations :

1. The **COLOR_HIST** descriptor performs best with the Chi-Squared distance due to its sensitivity to small differences in histogram bin values, aligning well with the discrete

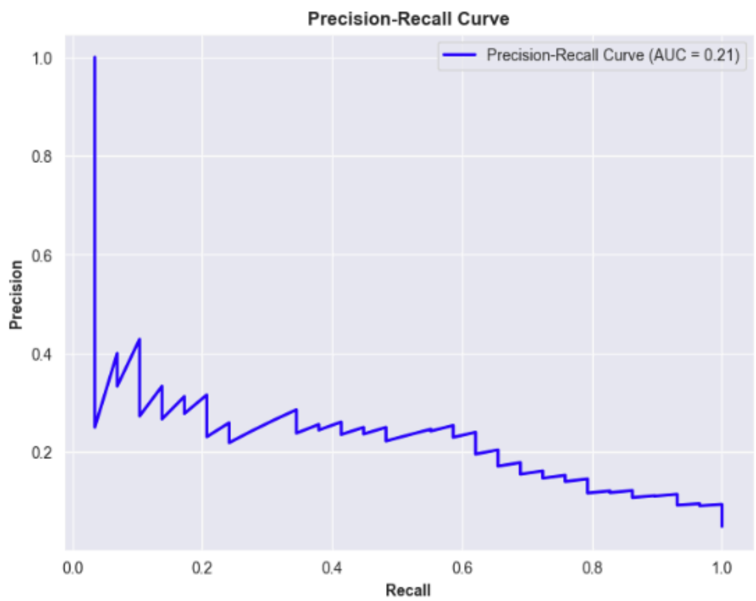
nature of colour histograms.

2. The GABOR descriptor benefits significantly from the Cosine distance measure, likely due to its emphasis on directional similarity, which aligns well with the orientation-based features of Gabor filters, resulting in a 2x improvement in performance compared to other measures. In contrast, the Cosine distance measure performs poorly for the Haralick descriptor, suggesting that this feature lacks a strong directional component. This is consistent with the fact that the Haralick descriptor quantizes the image into 256 grey levels, focusing on texture patterns rather than directional information.

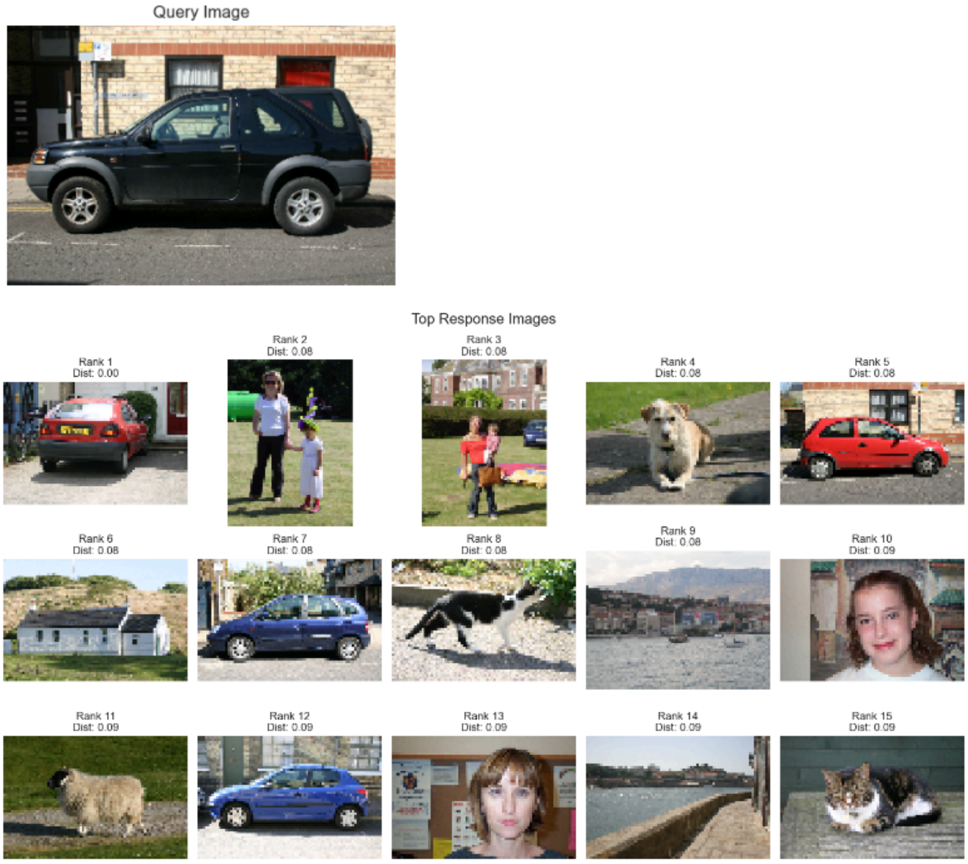
3. The observed performance highlights that no single distance measure is universally optimal. Instead, the choice of the most effective measure depends on the structural characteristics of the descriptors—whether they represent textures (continuous values), histograms (discrete distributions), or orientation-based features. This emphasizes the importance of tailoring the distance metric to the descriptor's properties for robust performance.

In additional experiments, we could investigate which distance measure yields better results at different quantization levels, as our current experiments have focused on performance at a fixed quantization level. Specifically, for the Gabor and Haralick descriptors, we have only evaluated performance at a single quantization level. Running experiments at varying quantization levels could provide a deeper understanding of these descriptors' performance across a broader range of settings. Additionally, a comparative analysis could be conducted to evaluate the performance of each distance measure relative to others, helping to determine whether a specific measure provides superior results at lower or higher levels of quantization.

Requirement 6: Bag of Visual Word retrieval



PR Curve achieved using BoVW (n_clusters = 100)



Top 15 Responses achieved using BoVW (n_clusters = 100)

The above PR curve and the top 15 responses are generated using the BoVW algorithm with the number of clusters set to 100 in the K-means algorithm. What is particularly notable is that the performance achieved by BoVW is quite impressive, especially considering that I randomly selected the number of clusters. In fact, it is the second-best result I obtained, after the Angular Quantization (Exp 3.2), where we iteratively ran the code to find the best angular quantization value. This suggests that the BoVW algorithm's performance is relatively robust, despite my random initial choice of clusters.

It is also worth noting that for each run of the BoVW, the performance slightly changes. This variation is due to the K-means algorithm, which involves a cluster assignment step in the initial phase, leading to different results depending on the random initialization. Additionally, the runtime of my implementation of the BoVW algorithm is quite high, which made it challenging for me to run further experiments to find the optimal number of clusters, especially since my laptop is not very powerful.

Additional experiments, that could be run with the BoVW descriptor include investigating how the performance of BoVW varies when query images from different classes are used. This would help assess the descriptor's generalizability across various categories and better understand its robustness in handling diverse data. Furthermore, if I had access to a more powerful system, I could experiment with different values of *n_clusters* in the K-means algorithm. By systematically varying the number of clusters, I could explore the true limits and potential of the BoVW descriptor.

Challenges

1. **Computational Constraints:** High quantization and clustering levels significantly increased the computational cost, slowing down the testing process. To address this, I reduced the range of quantization levels and the number of clusters in certain tests to optimize performance within hardware limits.

2. **Data Quality Issues:** During debugging, I found inconsistencies in the dataset, such as images like 1_26_s.bmp and 9_10_s.bmp, both depicting goats, being mislabeled into different classes. This mislabeling affected our ability to correctly measure descriptors' performance.

3. **Descriptor Integration:** In Experiment 3.1, combining angular and magnitude features with the Global Color Histogram showed no improvement, likely due to the colour histogram dominating the features. Thus I recommended Future work to explore descriptor weighting or additional feature extraction techniques to balance this.

Testing During Descriptor Implementation

1. **Edge Cases in Histogram Binning:** While implementing `colour_histogram`, I observed incorrect bin values for images with extreme pixel intensities (near 0 or 1). This was resolved by explicitly clipping the values during quantization to ensure they stayed within the range [0, Q−1].

2. **Gradient Normalization and Quantization in Sobel Features:** During the `sobel_quantization` development, I initially encountered uneven bin distributions for gradient angles due to inconsistent handling of negative angles. I fixed it by converting all angles to a [0, 360] degree range.

3. **Grid Partition Alignment Issues:** While implementing `color_texture_grid_descriptor`, I noticed that non-divisible image dimensions caused errors in grid cell extraction. This was resolved by dynamically adjusting the grid cell size for edge cases, ensuring all pixels were included.

Reference

1. Coursework Lecture Week 3 - Visual Search, Features and Matching
2. Computer Vision: Algorithms and Applications” (Szeliski)
    a. Ch. 4 (4.1)
    b. Ch.14 (14.4-5)
3. Andrew NG Deep Learning course on Coursera

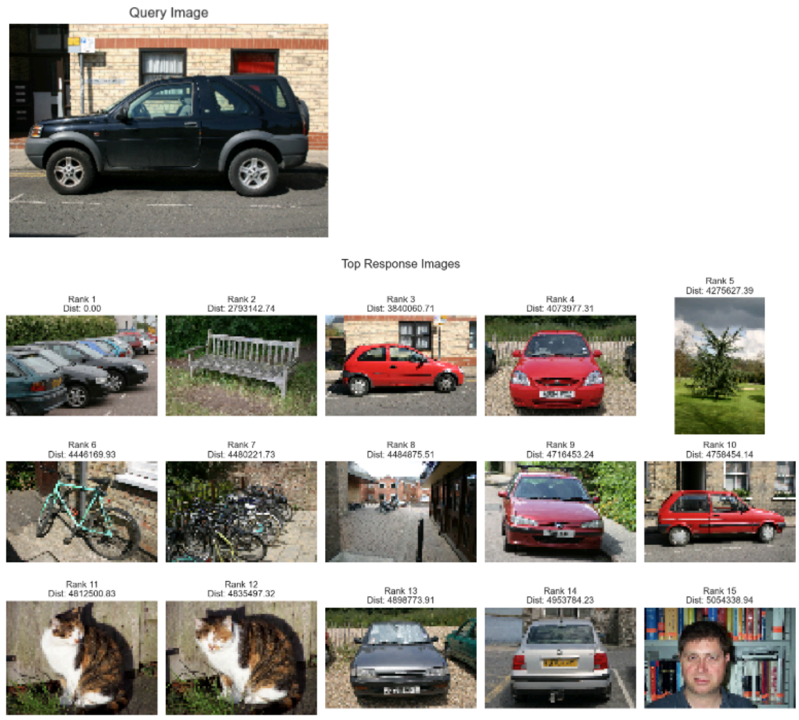
The End.

Appendix

Visual Results of Top Retrieved Images

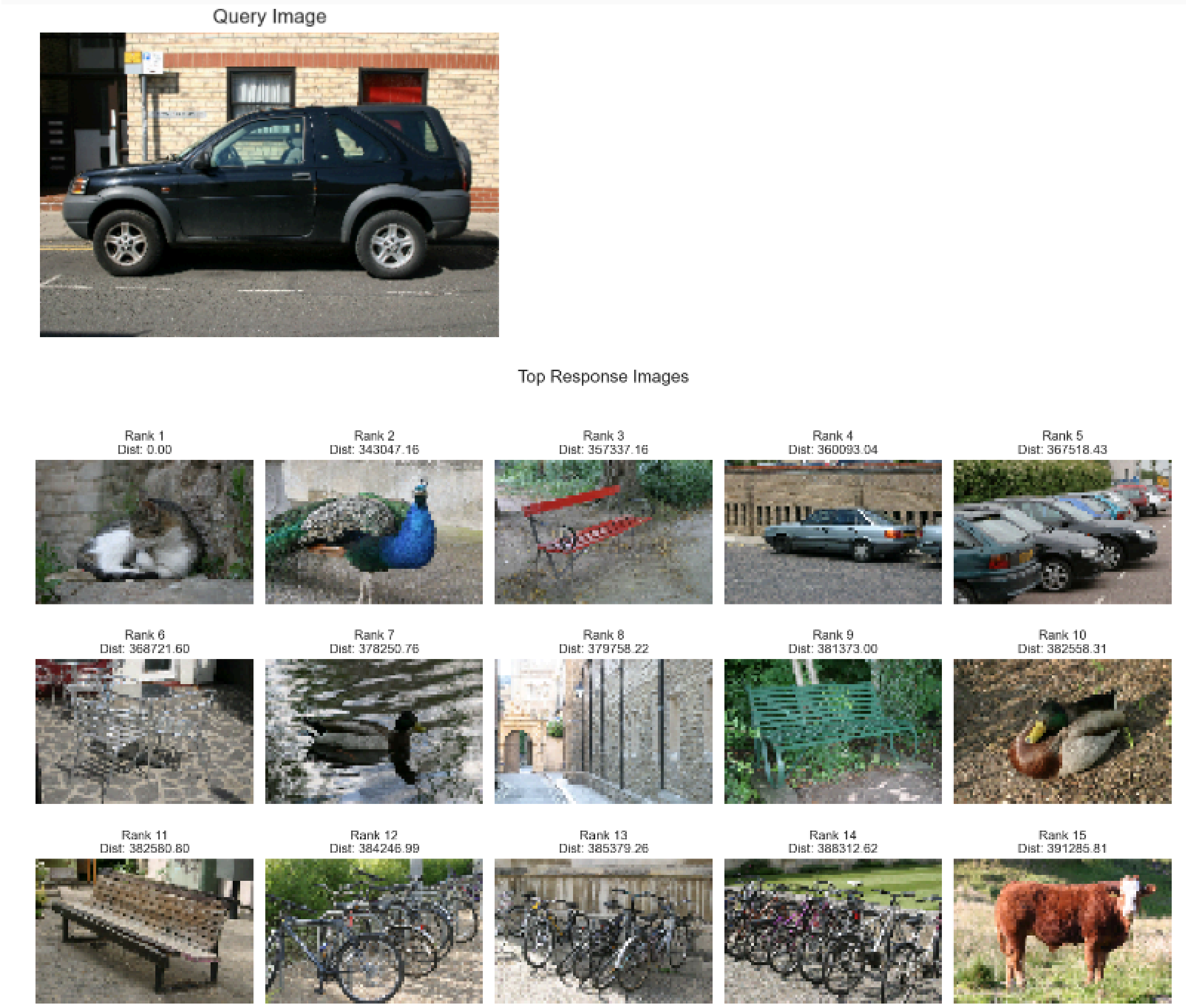
In this appendix, I present the top 15 images retrieved by each descriptor tested in the study. For a few descriptors, these results were excluded from the main body due to space constraints but are essential for providing a visual understanding of the descriptors' performance and retrieval accuracy. Notably, images are included only for those quantization levels or descriptor configurations for which Precision-Recall (PR) curves were presented in the main document.

Requirement 1: Global Colour Histogram



Experiment 3.1: Gridding-based experiment with Colour and Textual Features

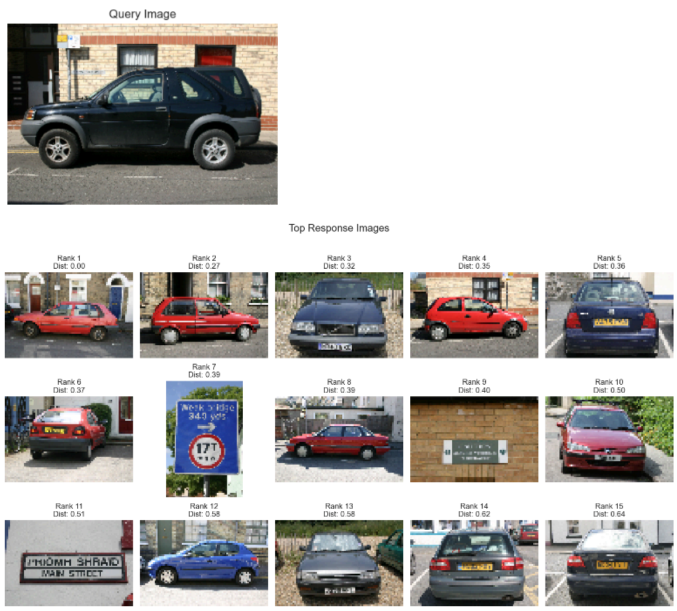
- Global Colour Histogram combined with Textual Descriptors (Angular and Magnitude).



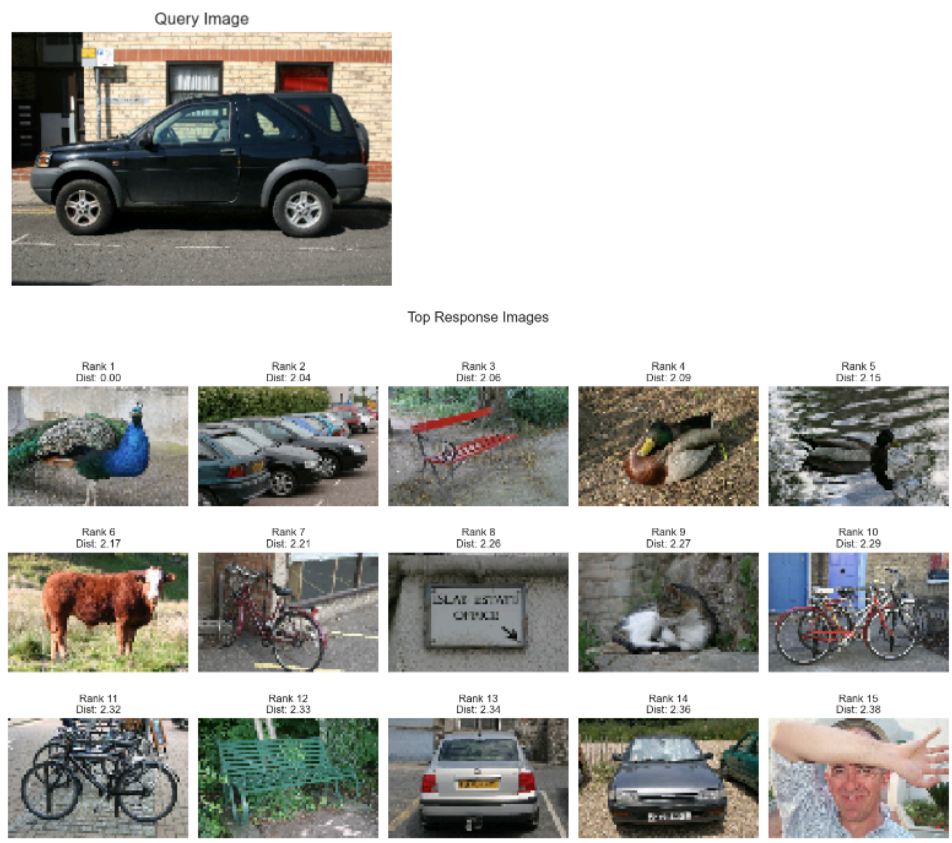
- Fig 2: Global Colour Histogram combined with Angular Quantization-based Descriptor.

- Fig 3: Global Colour Histogram combined with Magnitude Quantization-based Descriptor.

Experiment 4.1: PCA and Mahalanobis Distance Applied to the Best-Performing Angular Quantization (Q = 96)

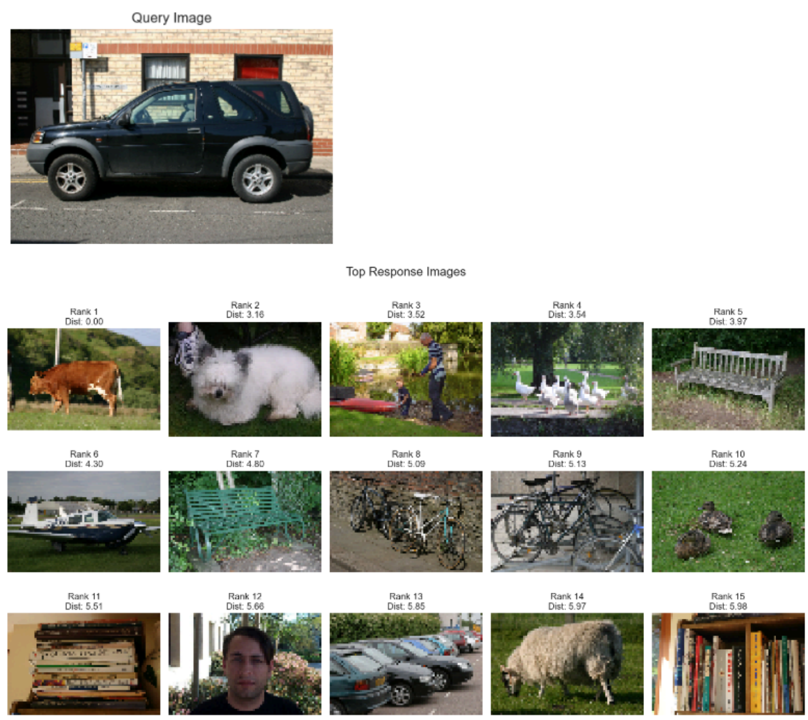


Experiment 4.2: PCA and Mahalanobis Distance Applied to the Grid Colour Angular Magnitude-based descriptor.



Exp 5.1: Experimenting with Different Descriptors

a) Gabor Filter



b) Harallick Filter