Efficient Image Similarity Search with Quadtrees (Student Abstract)

Yifan Zhang[1], **Jeff Heflin**[1]

[1] Dept. of Computer Science & Engineering
Lehigh University
Bethlehem, PA 18015, USA
yiz521@lehigh.edu, jeh3@lehigh.edu

Abstract

In this paper, we present a new image similarity search algorithm designed to enhance traditional information retrieval (IR) by adding an image search capability. Our approach uses a quadtree data structure to organize image data, significantly reducing search space and improving retrieval efficiency. We describe an indexing strategy and two query algorithms that can be implemented in any IR system. We tested our method on a 70K material microscopy image dataset, achieving a 25-times improvement in retrieval speed with only a 20% reduction in ranking accuracy.

Introduction

Traditional methods for image similarity search such as Pairwise and K-nearest neighbors (KNN), while effective for small datasets, struggle with scalability due to their high computational cost. As dataset sizes grow, these methods become impractical. The quadtree is a well-established spatial data structure, widely used in GIS and computer graphics, that recursively subdivides 2D space for efficient management. Though successful in tasks like image compression and collision detection, its potential for large-scale image similarity search remains underexplored (Ibrahim, Wahab, and Almaaly 2022).

Recent advancements in image similarity search focus on dimensionality reduction and Approximate Nearest Neighbor (ANN) methods, such as LSH and HNSW graphs, which improve retrieval speed but involve trade-offs between accuracy and storage (Li et al. 2019). These methods also lack integration with traditional information retrieval (IR) systems to support both image and metadata searches.

This paper introduces a novel quadtree-based image similarity search algorithm to address these challenges. We organize high-dimensional datasets into a quadtree for fast similarity searches and integrate quadtree indexing with IR to support both image and metadata retrieval. Our method shows significant improvements in retrieval speed with reasonable accuracy trade-offs compared to Pairwise and KNN.

Methodology

Before we can index images, we use a two-step process to represent them with useful 2D vectors. First, we need to represent an image as a meaningful vector in continuous space. We can use a generic embedding model pre-trained on natural images (e.g., ImageNet) or train a domain-specific embedding model. Second, embedding vectors are projected into a 2D space using dimensionality reduction. We found that UMAP provides the most useful projection to 2D, as it minimizes information loss during dimension reduction.

To efficiently retrieve the top n similar images for a given query from a dataset, our methodology consists of four primary steps: 1. constructing a quadtree to organize the dataset, 2. a novel tag scheme for quadtree paths that enables indexing by an IR system, 3. identifying the target quadtree regions (nodes) that contain the n most similar images, and 4. retrieving the final set of images via IR.

Direct comparison of the query image with each image in the data set is computationally inefficient, resulting in a time complexity of O(n). By leveraging the quadtree data structure to manage the 2D representation of the image dataset, we reduce the retrieval time complexity to O(log n). The quadtree achieves this by recursively partitioning the dataset into smaller regions. Initially, the dataset is represented by a single root node covering the entire space. Each leaf node can store a limited number of images, and when a new image is added, the quadtree checks whether the image's coordinates fall within the node's boundaries. If the node has available space, the image is stored there; otherwise, the node splits into four child nodes, and the existing images are redistributed to the appropriate child nodes. This process ensures that similar images are efficiently grouped together in leaf nodes.

IR systems typically create an inverted index for various fields, such as titles and content, where fields can differ in type and tokenization. To index the dataset with the quadtree structure in the IR system, our approach adds an additional field for each image: *quadtree path* which records the route from the quadtree root to the leaf node where the image is stored. The *quadtree path* is generated recursively. Starting at the root node, the quadtree ID is initialized as 0. Each time the quadtree splits into four child nodes, the ID is extended based on the child's location: 0 for northeast, 1 for southeast, 2 for southwest, and 3 for northwest. As the image moves deeper into the tree, new digits are added, creating a unique quadtree ID that traces the image's position in the tree. As the dataset grows and nodes are subdivided, images

may move to different nodes, altering their quadtree path. To address this, we assume a maximum tree depth of 10 (which can include up to 1,048,576 nodes) and pre-generate quadtree IDs for all subpaths to that depth. This approach ensures that the image's location is accurately tracked as the tree grows, and allows queries at different levels of abstraction.

We propose two strategies to find regions with the n most similar images: a window-based and a distance-based search algorithm. For the window-based strategy, the core idea is to focus on nodes (regions) closest to the query image first and expand the search outward if necessary. The algorithm dynamically adjusts the search window around the query point. Initially, it calculates an appropriate window size based on the dataset's global (the root node) density and the desired number of similar images. It then traverses the quadtree, selecting nodes that intersect with the window as candidate regions for similar images. If the initial window does not contain enough relevant images, the window is expanded based on the local (candidate nodes) density until the desired number of similar images is found. For the distance-based method, the algorithm begins at the root node of the quadtree and recursively explores its child nodes. The visited nodes are sorted based on the distance between each visited node and the query image. The nodes that are closest to the query image and contain a number of images close to n are selected as candidate regions. To balance between accuracy and retrieval speed, the algorithm ensures that the candidate regions are not too large (which could reduce accuracy by including irrelevant images) and not too small (which could slow down the retrieval by requiring more regions to be searched).

Experiments

To evaluate the performance of our proposed quadtree-based image retrieval methods, we conducted experiments on datasets of 30K, 50K, and 70K images taken from the Materials Project database[1]. We created embeddings using a symmetry-aware model trained by Drexel University. We compared five models: pairwise in full embedding space (Pairwise Full), Pairwise in 2D space (Pairwise 2D), KNN in full embedding space (KNN Full), Quadtree Distance-based search in 2D space (QT Distance), and Quadtree Window-based search in 2D space (QT Window).

Our evaluation task was to find the top 20 similar images to each of 1000 random query images. We used the Normalized Discounted Cumulative Gain (NDCG) score to assess accuracy, comparing the quadtree-based methods to Pairwise Full, which serves as the ideal reference. The NDCG score ranges from 0 to 1, with 1 indicating a perfect ranking. We calculated both median and mean NDCG scores for each method across dataset sizes. To assess speed, we measured the total query time in seconds. This combination of accuracy and efficiency allowed us to comprehensively evaluate the performance of the quadtree-based approaches.

Table **??** shows the experimental results. The standout advantage of our quadtree models was their near-constant retrieval time, regardless of dataset size. QT Distance and QT Window consistently retrieved results in 8-10 seconds for 1000 queries, unlike Pairwise and KNN, whose times increased linearly with dataset size. This resulted in a 24.67x speed improvement over Pairwise 2D and a 147x improvement over Pairwise Full. While the quadtree models traded off some accuracy, they performed reasonably well, with QT Distance achieving NDCG scores between 0.61 and 0.64, and QT Window ranging from 0.56 to 0.58. The accuracy reduction compared to Pairwise 2D was approximately 20.78%.

Size	Model	Median	Mean	Time
30K	Pairwise Full	1	1	528.04
30K	Pairwise 2D	0.80	0.76	89.58
30K	KNN Full	0.92	0.75	22.3
30K	QT Distance	0.64	0.65	8.44
30K	QT Window	0.58	0.56	8.44
50K	Pairwise Full	1	1	900.54
50K	Pairwise 2D	0.76	0.73	149.45
50K	KNN Full	0.92	0.75	36.59
50K	QT Distance	0.61	0.61	8.59
50K	QT Window	0.56	0.57	10.62
70K	Pairwise Full	1	1	1246.77
70K	Pairwise 2D	0.77	0.73	208.67
70K	KNN Full	0.92	0.73	52.43
70K	QT Distance	0.61	0.62	8.46
70K	QT Window	0.56	0.57	10.35

Table 1: Performance Comparison

Conclusion

In this paper, we presented a novel quadtree-based image similarity search algorithm. By leveraging the hierarchical structure of quadtrees, we organized high-dimensional image data for efficient and scalable retrieval. Experimental results demonstrated significant improvements in retrieval speed with reasonable accuracy trade-offs. In future work, we aim to improve the retrieval process while maintaining the speed advantage of our quadtree-based approach. One area of focus is exploring the use of octrees to map images to 3D space, reducing the loss of information caused by dimension reduction.

Acknowledgments

This material is based upon work supported by the National Science Foundation under Grant No. 2246463. We thank Joshua Agar and Yichen Guo for providing the pre-trained embedding model and materials dataset.

References

Ibrahim, S. F.; Wahab, H. B. A.; and Almaaly, A. M. J. 2022. Quad-Tree Image Segmentation Technique with Encryption: A survey. In *2022 Int'l Conf. on Data Sci. and Intel. Comp. (ICDSIC)*. IEEE.

Li, W.; Zhang, Y.; Sun, Y.; Wang, W.; Li, M.; Zhang, W.; and Lin, X. 2019. Approximate nearest neighbor search on high dimensional data—experiments, analyses, and improvement. *IEEE Trans. on Knowledge and Data Engineering*, 32(8): 1475–1488.

[1]https://next-gen.materialsproject.org/