Inverted Index Compression for Scalable Image Matching

David M. Chen[1], Sam S. Tsai[1], Vijay Chandrasekhar[1], Gabriel Takacs[1],
Ramakrishna Vedantham[2], Radek Grzeszczuk[2], Bernd Girod[1]
[1]Department of Electrical Engineering, Stanford University, Stanford, CA 94305
[2]Nokia Research Center, Palo Alto, CA 94304

Recent advances in visual search have been driven by local robust image features such as SIFT, SURF, and CHoG. Equally important for large-scale visual search is a data structure known as a Vocabulary Tree (VT). The VT splits the feature descriptor space by tree-structed vector quantization (TSVQ), where the VQ centroids are the tree nodes. Each image's set of feature descriptors can then be compactly summarized by a tree histogram, which counts how often each tree node is visited. To speed up image search, the VT uses an inverted index. For each tree node, the inverted index stores a list of image IDs indicating which database images have visited that node, as well as the visit frequencies. During a query, features are extracted from the query image and the feature descriptors are classified through the VT. Then, similarity scores are efficiently computed for all database images by traversing only the inverted lists at tree nodes visited by the query feature descriptors.

In this paper, we address a key challenge for scaling image search up to larger databases: the amount of memory consumed by the inverted index. In a VT-based image retrieval system, the most memory-intensive structure is the inverted index. For example, in a database of one million images where each image contains hundreds of features, the inverted index consumes 2.5 GB of RAM. Such large memory usage limits the ability to run other concurrent processes on the same server, such as recognition systems for other databases. A memory-congested server can exhibit swapping between main and virtual memory, which significantly slows down all processes.

We have designed and implemented efficient methods for compressing an inverted index in memory. By exploiting the statistics of image IDs and visit counts contained in the inverted lists, memory usage can be reduced as much as $5\times$ without any loss in accuracy. For a database of one million CD/DVD/book covers, we reduce the inverted index's memory consumption from 2.5 GB to 0.5 GB. Special care is taken to achieve short decoding times, so that the compressed inverted index can be effectively integrated into our real-time mobile image search system. Two fast decoding alternatives to arithmetic coding achieve nearly the same compression ratio but require $6\times$ shorter decoding delay. With fast decoding, reducing the inverted index's memory usage can actually speed up retrieval on a server with limited memory. On a memory-congested server, retrieval from a one-million-image database can take 6 seconds without inverted index compression, but with compression, the retrieval latency drops to 1 second. Our work also includes how to optimally reorder the database image IDs to maximize the compression gain. Cast as a traveling salesman problem, the reordering optimization provides an additional 37 percent savings in memory usage.

Inverted index compression also makes affordable two memory-intensive image matching techniques that boost classification accuracy: soft-binned feature classification and multiview VT's. The main cost of both soft binning and multiview VT's is a several-fold increase in the inverted index's memory usage. Our coding methods effectively reduce the large memory consumption by $5\times$ to a much more manageable amount. For soft binning, we extend our basic coding scheme to include Lloyd-Max quantization of fractional counts. Quantization actually denoises the fractional counts and further improves classification performance.

1068-0314/10 $26.00 © 2010 IEEE
DOI 10.1109/DCC.2010.53

IEEE computer society