Vol.2012-CVIM-181 No.13
2012/3/16

Building a large scale visual index of landmarks and deploying a fast recognition index.

Marco Andreetto,[†1] Hartwig Adam,[†1]
Alessandro Bissacco,[†1] Fernando Brucher,[†1]
Ulrich Buddemeier,[†1] Lijia Jin,[†1] Yuan Li,[†1]
Anand Pillai[†1] and Hartmut Neven [†1]

To create a fast and comprehensive landmark recognition system one has to solve many different challenges. First, information about landmarks has to be extracted from many heterogeneous sources. Second, visual models describing those landmarks need to be built and correctly matched. Finally, it is necessary to create a fast and accurate recognition engine for matching a query image against the landmark visual models. In this talk I will describe how it is possible to address the aforementioned challenges. In particular I will describe how landmark information and visual models can be constructed by mining photo sharing websites and other web resources. Also I will discuss how to efficiently search during query time such a large collection of visual information so that a real time visual recognition service can be deployed.

1. Introduction

Landmarks feature prominently in image collections. These include both natural sites such as the Grand Canyon or Bryce National Park, and buildings such as Saint Peter's Basilica, Big Ben and the Colosseum. Recognizing these interesting locations is therefore very useful for understanding the content of a single image (a picture of the Eiffel Tower) and of a group of images (all the pictures from the same vacation trip) and can lead to important applications such as search by image content, and photo annotation. To create such a recognition system many different problems need to be addressed:

- Mining image collections for visual and text information about landmarks. It's important to observe that there is no comprehensive list of landmarks, and that the concept of landmark itself is poorly defined. While everybody may agree that the Colosseum is an important landmark, the front of a popular store or restaurant may also be of interest to many people and it can appear as background in many photos. Recognizing such locations would be useful, but the amount of information that can be mined in such cases can be limited and very noisy.
- Building visual models describing the mined landmarks. This problem is closely related to the previous one. In building the model, text and visual information are combined. Once a first version of the model is available "noisy" images and text can be pruned, and the model improved.
- Input images must be recognized by matching against the available models. Such a recognition process should be both accurate and fast, ideally on-line. While one can imagine applications that do not require on-line recognition, in general any large scale recognition engine needs fast recognition to handle the ever growing set of uploaded images.

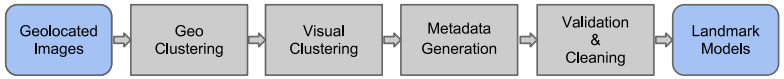

Fig. 1 Landmark mining pipeline.

A possible solution to the first two coupled problems has been presented in[1)] and will be described in Section 2. To address the third problem Buddermeier and Bissacco[2)] developed a parallel version of a k-D tree index. This parallel index can handle a large number of concurrent queries while maintaining an high recognition accuracy. This solution will be described in Section 3. See[3)] for an in-depth discussion.

2. Mining image collections and building visual models.

To recognize landmarks in pictures one needs to construct a visual model describing the appearance the landmark can have in the set of query images. Then additional information describing the landmark has to be attached to each visual model, such as its name, location and possibly a list of useful web links for the landmark (Wikipedia entries). Given an ever growing set of input images and the need to recognize landmarks or locations of moderate or local interest, the land-

†1 Google Inc.

marks models should be constructed with either limited human supervision or no supervision at all. Figure 1 shows the conceptual sequence of steps (pipeline) to construct such landmark visual models (see[1)].

- The input images used to construct the model can be obtained from photo sharing websites like Panoramio and Picasa. Those sources of images have similar visual statistics to the ones we want to recognize. Because we are interested in landmarks, we consider only images associated with latitude and longitude coordinates.
- Images are then grouped in clusters or bins based on their geographic distribution. Since the latitude and longitude of a model can be quite noisy depending on their source, we allow for images that are considerably far apart to be grouped together.
- All the images in the same geo-cluster are then matched against each other using the same method described in Section 3. The final result is a graph for each view cluster whose nodes represent images in the geo-cluster and whose edges describe visual matching among images. Each edge has a score that describes the reliability of the match. A standard graph clustering algorithm can then be used to obtain the view-cluster. In our pipeline we used single linkage clustering.
- Besides its coordinates, each image can have some text labels associated with it. These tags are pooled together in a view-cluster to generate a candidate name. Common tags like "Summer vacation" or tags that are present over a large number of geo clusters (e.g. "Italy") are removed. For each view-cluster a popularity signal is computed by considering its cardinality. Smaller view-clusters can then be pruned as unreliable.
- The previous step for generating metadata can be prone to errors. At least for the most popular view-clusters, which are also the most likely to be recognized, a validation step may be necessary to avoid errors. This step can also provide additional information for a landmark such as its Wikipedia entry. For the less popular landmarks only their position, estimated as the average of the photo positions, can be returned.

All the images contained in the final landmark view clusters can be used to recognize a query image. Once the query image is matched against an image in a view-cluster the landmark metadata for that view-cluster can be returned as the query result.

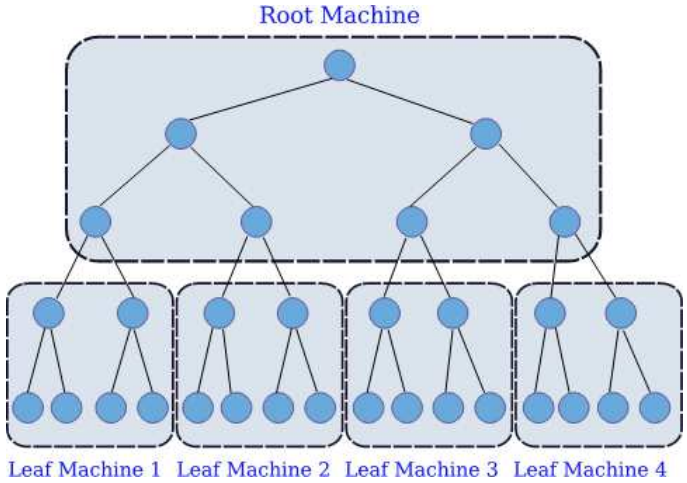

Fig. 2 Distributed k-d tree. A k-d tree is distributed across $N + 1$ machines by dividing the tree into a "root" component (top of the tree) and a set of N subtrees (index shards).

3. Fast landmark recognition with distributed k-d tree.

A widely applied approach to image retrieval and specific object recognition is that developed by David Lowe in[4)]. In this approach a set of local descriptors are extracted from keypoints in the query image. Approximate nearest neighbors of the query features are search in a dataset of descriptors obtained from the images in the index. Candidate matches are then clustered in groups that are coherent with a given geometric transform, typically an affine transform. Since this approach requires the whole index of descriptors to be stored in memory it cannot be scaled to large collection of several millions images. To address this problem Buddermeier and Bissacco proposed the distributed architecture described in Figure. 2. In this solution a conceptually unique k-d tree containing the descriptors of the images to retrieve is divided into $N + 1$ components, each stored and managed by a single machine. The root machine contains the top of the k-d-tree. All the other M machines contain a set of subtrees called shards.

Vol.2012-CVIM-181 No.13
2012/3/16

The root machine receives the query descriptors and traverse the first levels of the k-d tree to find what shard are likely to contains the nearest neighbors. The "root" machine receives requests to locate the (approximate) nearest neighbors of a query feature point. It selects the index shards that most likely contain the nearest neighbors and forwards the query point to them. Each selected shard then operates as a standard k-d tree[5] and returns the results to the root machines. The computational burden of the search - computing feature distances and search backtracking - is carried out by the N machines managing the shards. So by increasing the number of machines it's possible to scale the system to large image collections. This distributed solution has been found to perform very similarly to the single k-d tree in terms of recognition accuracy. See[3] for a comparison.

References

1) Zheng, Y., Zhao, M., Song, Y., Adam, H., Buddemeier, U., Bissacco, A., Brucher, F., Chua, T.S., and Neven, H.: "Tour the World: building a web-scale landmark recognition engine", em International Conference on Computer Vision and Pattern Recognition (CVPR), 2009.

2) Buddemeier U., and Bissacco. A,: "Distributed kd-tree for effcient approximate nearest neighbor search", *Patent Pending*: 2009.

3) Aly, M., Munich, M., and Perona, P.: "Distributed Kd-Trees for Retrieval from Very Large Image Collections. *British Machine Vision Conference (BMVC)*, Dundee, UK, August 2011.

4) Lowe, D.: "Distinctive image features from scale-invariant keypoints". *International Jornaul of Computer Vision*, 2004

5) Beis, J. S. and Lowe, D. G.:"Shape Indexing Using Approximate Nearest-Neighbour Search in High-Dimensional Spaces", em 1997 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'97), 1997