Scalable Logo Recognition in Real-World Images

Stefan Romberg
Multimedia Computing Lab
University of Augsburg
Augsburg, Germany
romberg@informatik.uni-augsburg.de

Lluis Garcia Pueyo
Yahoo! Research
4401 Great America Parkway
Santa Clara, CA, 95054
lluis@yahoo-inc.com

Rainer Lienhart
Multimedia Computing Lab
University of Augsburg
Augsburg, Germany
lienhart@informatik.uni-augsburg.de

Roelof van Zwol
Yahoo! Research
4401 Great America Parkway
Santa Clara, CA, 95054
roelof@yahoo-inc.com

ABSTRACT

In this paper we propose a highly effective and scalable framework for recognizing logos in images. At the core of our approach lays a method for encoding and indexing the relative spatial layout of local features detected in the logo images. Based on the analysis of the local features and the composition of basic spatial structures, such as edges and triangles, we can derive a quantized representation of the regions in the logos and minimize the false positive detections. Furthermore, we propose a cascaded index for scalable multi-class recognition of logos.

For the evaluation of our system, we have constructed and released a logo recognition benchmark which consists of manually labeled logo images, complemented with non-logo images, all posted on Flickr. The dataset consists of a training, validation, and test set with 32 logo-classes. We thoroughly evaluate our system with this benchmark and show that our approach effectively recognizes different logo classes with high precision.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval; I.5.4 [**Pattern Recognition**]: Computer Vision

General Terms

Algorithms, Experimentation

Keywords

indexing local features, logo recognition, object recognition

ICMR '11, April 17-20, Trento, Italy
https://doi.org/10.1145/1991996.1992021

Figure 1: Point triples (red lines) detected on the Fedex logo. Green crosses denote local features.

1. INTRODUCTION

One long-standing goal of computer vision is certainly object recognition. Much research has been dedicated to object recognition in general and also various subproblems have been explored, nevertheless the task remains challenging. In this paper we focus on logo recognition. We consider logo recognition a subset of object recognition, as most logos can be considered objects with a planar surface. In addition logos are designed to catch someone's attention. However, a specific logo class can have relatively large intra-class variance and the recognition of logos in natural images has to deal with perspective tilt which is the main difference to near-duplicate retrieval approaches.

There are many powerful object recognition schemes, such as for instance proposed by Felzenszwalb et al [4], but these methods usually require to learn and apply one or more models per class. In contrast, our long term goal is to perform logo recognition on thousands of images per minute, whereby the number of different logo classes is huge. In that case the usage of one-vs-all classifiers can be expensive or even infeasible. In general a logo recognition system should be able to determine quickly if an unknown image contains a logo of a certain class. While it is certainly doable to test an incoming test image for *every* logo class present in the database this is not an option due to speed issues, the expected high false alarm rate and especially because of the likely disagreement among the multiple classifiers.

To illustrate this, consider the following real-world scenario. On Flickr[1], a large scale social media sharing site,

[1]Flickr: http://www.flickr.com

more than 4000 images are uploaded every minute. Typically, a logo database contains multiple samples of thousands of brands. While the commercial interest to detect logos in images is huge, before any recognition system can be deployed in practice it needs to address the scalability constraints in terms of images processed per minute and number of logo-classes supported, while maintaining a very high rate of recognition accuracy. In this work we go one step towards this long term goal and propose a system that is heavily tuned towards efficient multi-class object recognition with high precision at the cost of recall.

At the basis of our system we use local features, which have been proven to be efficient for image object retrieval in general. Following the visual bag of words paradigm [11], local features can be quantized using a visual dictionary and the image be treated as document containing visual words. Existing retrieval models can then be applied to retrieve similar candidate images. However, for image object retrieval systems to obtain acceptable retrieval performance an expensive post retrieval step is needed, to verify the existence of the object in the candidate image.

Therefore, we propose to index the relative spatial layout of local features on logo regions by means of a cascaded index. The primary index contains quantized pairs of points, each point representing a patch in the model of the logo-class. When a pair of points from a test image correspond with an entry in the primary index, the secondary index is queried, which consists of the triples of points forming the model of each logo class. Figure 1 depicts an example of detected point triples (in red) for the Fedex logo. The introduction of the cascaded index drastically reduces the number of false positive classifications to the point that we no longer need to perform the expansive post-retrieval step on each logo-class to validate the presence of that logo in the test image.

The remainder of this paper is organized as follows. Section 2 describes related work relevant in the context of this paper. The motivation for the cascaded index is given in Section 3. In Section 4 it is discussed how the proposed representation of spatial layout can be used to derive a model for each logo class. The usage of the cascaded index for efficient logo recognition is discussed in Section 5. We present our dataset for evaluation in Section 6. The results of our experiments are then presented in Section 7, followed by the conclusions in Section 8.

2. RELATED WORK

The related work on image retrieval and object recognition is vast. In this section we highlight the related work on image object retrieval and logo recognition that is relevant in the context of our approach.

There are several publications that address the retrieval of printed logos, e.g. for efficient search in logo databases used for petty patents. However, logo recognition in photos has not gained as much attention. Bagdanov et al. [2] retrieve logos from sports video databases by directly matching feature descriptors in the video to these of training images. Joly et al. [5] propose a new visual query expansion strategy for querying a database with SIFT descriptors followed by a geometric consistency check. Both approaches do not scale well with an increasing database size.

A turning point in scalable image object retrieval, which also provides the basis for our approach is the introduction of the bag-of-words approach [11]. At the core of the visual bag-of-words approach is the quantization of local features for efficient indexing instead of matching the raw descriptors directly. The descriptors are quantized to discrete visual words that form a vocabulary derived by k-means clustering. However, for this approach to be succesful, one or more post-retrieval steps are needed to filter the high number of false-positives. By embeding the spatial information in the index, we can significantly reduce the number of false positives, as we will show in this paper. In addition, and unlike recent work on near-duplicate retrieval [13, 3, 9] where large vocabularies with several thousands up to millions visual words are used, we have to deal with intra-class variance of logos themselves and perspective tilt. To minimize quantization errors we therefore use a relatively small vocabulary of 2000 visual words. In this work we use SIFT [7] descriptors derived from hessian-affine interest points [8] to describe images as these are more robust to image tilt and perspective transformations than other features. However, our approach can be used with other local features as well. Kleban et al. [6] also do logo recognition by performing frequent item-set mining to discover association rules in spatial pyramids of visual words.

Essentially different to other existing approaches is the encoding of spatial structure with a hash function. In Section 4 we describe in detail how this can be achieved. In short, similar regions in two images can be determined by indexing geometric structures of local features in hash tables. Our work in this area is similar to geometric hashing [12], however we do not have a single model image but create a model for each logo-class out of several training images. The result of the training is a set of triples of interest point, which consistently appear across the different training images. Poullot et al. [10] use bucketing techniques to build a signature of local feature triples. In contrast to our approach they build a signature for the whole image and their approach is more suited to near-duplicate detection. In a recent publication, Avrithis et al. [1] incorporate global geometry in the index by means of feature map hashing. Rather than building a global geometric representation, we index spatial structures that describe a small region of the image, making our approach more suitable for the detection of logo objects.

3. THE CASCADED INDEX

Experience shows that an index that holds single features - e.g. stored within an inverted file - can be easily used for image retrieval, but the features or the bag-of-word model lack distinctiveness which usually makes a post-retrieval verification step necessary. In previous qualitative experiments we experienced that even pairs of local features have not been as discriminative as desired for logo recognition. Because of that and moreover because three points are the minimum number of points to describe a 2-D plane we chose to describe the spatial structure of logos with feature triples.

The main idea is that if two images contain similar regions we can expect similar spatial layout of features. In other words within these regions the relative position of the features to each other should be roughly the same for both images. This holds disregarding the actual orientation and scale of such a region.

We therefore propose a new index type that holds the spatial structure of feature triples and still can be queried

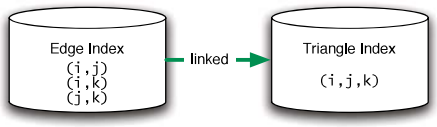

Figure 2: Schema of the cascaded index. i, j and k denote the local features that have been indexed.

efficiently. The structure of the index itself is motivated by the way test images are scanned for logos. For an unknown test image we do not have any prior knowledge where to explore promising point configurations that should be matched against the index. Therefore we propose to use a *cascaded index*. The cascaded index holds both lower-dimensional and higher-dimensional feature representations. These are linked such that each lower-dimensional feature representation is part of a higher-dimensional representation. This structure allows to scan the sparsely populated high-dimensional feature space by determining a lower-dimensional subspace which is part of it first.

Here, we chose to index local feature pairs within an *edge index* as the lower dimensional description and feature triples including their relative layout in a *triangle index*. The two indexes are linked, such that the edge index contains only edges that are part of the triangle index.

This dependency allows us to perform efficient queries in two steps: (1) First pairs of points are used to query the edge index, which contains the edges corresponding to pairs of points in the logos. When pairs of random points are matched against this index only a small number of edges will be matched successfully. (2) These edges give a hint from where random triangles could be constructed. Therefore, the two points of an edge plus an additional point are taken for constructing a query to the triangle index.

To be precise we want to define some terms used throughout the paper. A pair of points enriched with their relative orientations and point triples enriched with their relative layout are called *edges* and *triangles*. We refer to the k-means quantization of local feature descriptors to visual words as *labels* and declare points as matching if their labels are equal. Edges and triangles are considered to *match* if they consist of points that match and also have similar spatial layout. The training aims to gather these matching edges and triangles from training images.

4. TRAINING

As we eventually want to describe the spatial layout of visual features of each class by using an index of edges and triangles we need to build a model for every logo class. That is, given a pair of training images, the training procedure should find correspondences of triangles (i, j, k) such that these are present in both images and within a region that can be judged by humans as visually similar.

We would like to stress the fact that the training we employ can be replaced with other approaches more suited for finding correspondences across two images. One example of an alternative training would be to estimate the homography between training images with RANSAC to derive feature triples. However, to illustrate the indexing of triangles we demonstrate how this indexing technique can be used to determine correspondences across training images by indexing the local features of a single image.

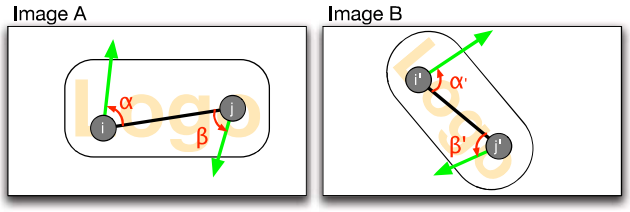

Figure 3: Relative position of feature pairs in two images. Green arrows indicate the orientation of the corresponding SIFT feature.

The triangle index of each class is created by simply aggregating the triangles resulting from matching all pairs of training images. Then for each triangle all of its edges are stored in the edge index. Thus the edges in this index are always part of an indexed triangle. Once the training is done test images can then be matched against the whole index containing the triangle representations of all classes.

4.1 Learning of logo classes

To represent a certain logo class n training images are selected and matching triangles are computed for any combination of these. That is, if there are n images per class $\frac{n(n-1)}{2}$ image pairs are matched. Note that we often observed that the matching procedure was not able to find matching triangles for a certain training image pair. Primarily this is caused by the inability of the features to be robust against large tilts. In addition many logos are on reflecting background which is known to distort descriptors. However, the advantage of this exhaustive matching procedure is that images that cannot be matched with some images can still be matched to other images. The result of the matching procedure can be considered a graph where only some images are connected. So even if the matching fails in some cases, e.g. due to very challenging image pairs and the imperfectness of the features, the connected subgraphs yield matches across different images. For instance, matching fails for logos where the colours are inverted. All the matching triangles then form the *model* of a logo class.

For indexing we quantize the feature descriptors to discrete visual word IDs called labels. We further compute the Euclidean distance between descriptors if the visual word labels are equal. This allows to sort the visual word matches between two descriptors in ascending order by their Euclidean distance. For each descriptor i in image I_A only the top 300 matches to descriptors in image I_b are kept and vice versa. Each feature is described by its x- and y-position, scale, orientation, label and its descriptor.

4.2 Similarity of relative positions and angles

We aim to find initial edge candidates by finding pairs of points (i, j) that have the same visual word labels and their features have also similar *relative* orientations α, β to each other (see Figure 3). Let F_a be the set of features in image I_A and F_b the set of features in image I_B. Then $F_{a\times b}$ denotes the set of all combinations of local features of one image with features with identical visual word labels of the other image. The subsequent steps determine the matching pairs (i, j) from all potential pairs $F_{a\times b}$ and discard all others.

For any two point pairs as in Figure 3 we can compute the

relative orientation of these two points *across* both images by computing the difference of the relative angle:

$$\Delta\alpha = \alpha - \alpha' \text{ and } \Delta\beta = \beta - \beta' \quad (1)$$

Then we compute an angle similarity score $s(\Delta\alpha)$ for all feature pairs (i, j) in $F_{a\times b}$. The similarity score $s(\Delta\alpha)$ is based on the difference of the angles *across* images A and B between the orientation of i and i' and the relative position of j and j'. We compute a normalized score for this difference of the two angles, i.e.

$$s(\Delta\alpha) = \eta\, e^{-\frac{(\Delta\alpha)^2}{2\sigma^2}}. \quad (2)$$

This score yields a value that indicates the similarity of the angles. We empirically adjusted $\sigma = 8$ and normalize the score by η such that the score curve has its maximum of 1.0 at 0° difference. With increasing $\Delta\alpha$ the score quickly and smoothly drops to zero for differences higher than 25°.

While $s(\Delta\alpha)$ only considers a score for the relative position of j seen from point i we can derive a symmetric score: The score $s(\Delta\beta)$ describes the similarity from the view of point j relative to position of i. The final symmetric score for an edge (i, j) is then defined as:

$$sim_{edge}(i, j) = sim_{edge}(j, i) = s(\Delta\alpha)s(\Delta\beta) \quad (3)$$

Note that $sim_{edge}(i, j)$ will quickly drop to 0 if one of the two angles is not consistent across the two images. While the comparison of such a difference of angles could yield a binary result the use of a continuous score function allows to sort the matching edges by their match quality.

All edges (i, j) that have a score $sim_{edge}(i, j)$ above a threshold T_{sim} form the initial edge set E_{match}. Many potential combinations of features (i, j) of the two images are therefore excluded from further computations. The remaining edges are known to match across the two images and serve as a starting point for further processing of spatial configurations.

4.3 Matching a pair of images

The encoding of geometric structure as described in Section 4.2 greatly reduces the number of false positive edge matches. Virtually we have increased the quantization space for edge signatures using $sim_{edge}(i, j)$ as a highly discriminative function to filter non-matching edges. In this section we expand the spatial awareness of our matching technique, to further reduce the noise.

Creating and analysing triangles. We incorporate spatial layout by indexing triangles, rather than edges. In particular, we index the spatial structure of all point triplets of image I_A and then match the point triplets of image I_B against this index. Let E_{match} be the set of detected edges (Section 4.2) and V_{match} the set of points within E_{match}. A triangle (i, j, k) is derived by randomly selecting an edge (i, j) from E_{match} and a third arbitrary point k from V_{match}.

Figure 4 depicts the information extracted from each triangle to create a signature. Each triangle is described by an 8-tuple, of which the quantized elements form a signature:

- The quantized labels for each of the three points (i, j, k).
- The quantized angle δ_1 between edges (i, j) and (i, k).
- The quantized angle δ_2 between edges (j, k) and (j, i).
- The relative orientations α, β and γ of the three points.

The combination of the two angles δ_1 and δ_2 captures the shape of the triangle. By observation we then found that taking the orientations of the features themselves into account leads to better performance on letter-like logos. In this case the orientations of the features are also important to describe the layout. Therefore we include the *relative orientations* α, β and γ (see Figure 4) of the three points in the signature. These orientations depend on the orientations of the SIFT features but are relative to each other. Therefore in-plane rotation and scale invariance of the triangle representation is maintained. Out-of-plane rotations are captured by the width of the quantization bins for the angles describing the triangle's shape.

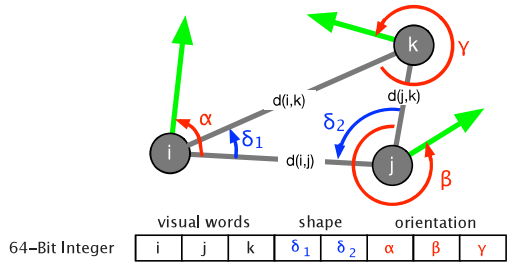

Figure 4: Representation of a triangle

Creating triangle signatures. Quantizing the angles and proportions in bins using hard boundaries introduces errors and potential loss of matches. Therefore, when constructing the signature for each triangle multiple signature variants are generated. We have extensively experimented with the parameter settings to minimize the quantization error, while maintaining the specificity of the triangle signatures. We observed optimal performance when multiple signatures for each triangle were stored. Therefore, the four different angles and the proportion of the distances are quantized to both the best bin and the second best bin. As a result we store 32 ($= 2^5$) different signatures for each triangle. The optimal quantization for all three parameters is evaluated in Section 7.3. For efficiency each triangle signature is packed into a 64-bit integer value (see Figure 4) before indexed.

To avoid degenerated triangles to be included in the index we impose some additional constraints on the points (i, j, k). If these are not satisfied, the triangle is not stored in the index. We discard degenerated triangles that carry little spatial information and are not descriptive. One such constraint is that each of the three points was quantized to a different visual word label. This constraint comes from the fact that many logos have some kind of border. Along that border many interest points may have similar descriptors and have the same label. Point triples consisting of such points do not carry sufficient discriminative information. In addition the spatial distance between all of the points (i, j, k) has to be above 5 pixel and the minimum angle of the triangle has to be 15°. Both constraints discard triangles where at least two points are located very close to each other. In that case the triangle does not describe spatial structure. Finally we constrain the eccentricities $\frac{d(i,j)}{d(i,k)}$ and $\frac{d(j,k)}{d(i,j)}$ such that these are in $[\frac{1}{3}, 3]$. This constraint also discards triangles which carry little information of spatial structure but more important it improves the locality of the detection. Even if two features on the actual logo were detected we often observed

Figure 5: Two triangles that match across images. Green lines show the orientations of the features. The numbers denote the visual word label.

a bad detected triangle caused by the detection of a random third point outside the actual logo.

Detecting triangles.. When matching two images, we store the signature of each triangle in the first image I_A in a hash table. To find a correspondence to triangles of the second image I_B, we then test all the triangles constructed from the top edges in E_{match} and an additional point therein if a similar signature exist in the hash table. Note that for every triangle in I_B we only generate a single discrete representation and only query the hashtable of I_A once. Figure 5 depicts a found matching triangle between two training images of class "Ford".

5. RECOGNITION

5.1 Querying the edge index

Given an unknown test image we have no prior knowledge regarding which logo it contains (if any at all) and at what locations, scales and sizes. Therefore, a Monte Carlo method is employed: The edge index is queried with randomly selected pairs of points out of the set of visual features V of the input image. For most queries no match will be found in the index. Ideally only matches are found for queries that were extracted from a region where a logo is present. Compared to the number of possible combinations of points the randomly chosen pairs of points and the number of returned matches is relatively small. Each edge detection implies that a part of a triangle within the triangle index has been detected. The triangles are then drawn by randomly sampling an edge from the set of detected edges and sampling an additional point belonging to one of these.[2]

Note that querying a hash table is extremely fast and therefore a huge number of random samples can be tested. Unfortunately the feature space of our triangle description is extremely huge, such that it is not feasible to scan this sparsely populated space efficiently in practice. However, the given edge detections within a lower dimensional feature space (the space of edge representations) allow to scan the semantically linked yet higher dimensional feature space of triangle representations in an efficient manner as only a subspace has to be scanned therein. One can observe that with this cascaded sampling Logos covering a big area in the image are very likely to be discovered quickly.

However, logos that are small may need many queries from a small area to be discovered successfully. Therefore in addition to the Monte Carlo sampling close pairs of points are sampled by selecting neighbors within a distance of $3px$ to $30px$ of a given point. These samples will likely cover even very small logos. As the queries are not selected randomly a relatively small number of queries are enough to cover the whole image thoroughly.

[2]Randomly drawn degenerated triangles (see Section 4.3) can be discarded without actually querying the index. Duplicate drawn samples are discarded as well.

5.2 Querying the triangle index

Given the set of edges E_{hit} that are contained in the edge index we then construct triangle queries using (i, j) and a third point k from E_{hit} and query the triangle index. Once we find matching triangles then the votes for each class are accumulated and yield the frequency of detections per class.

The detection counts are then thresholded with a decision threshold T_{class} for *each class separately* to determine whether a logo is present or not. Section 7.2 describes how these class-specific thresholds are obtained in an adaptive manner. In general, once a class has more than T_{class} triangle detections the logo is considered as discovered. Otherwise the image is considered to show no logo.

6. DATASET

For a realistic evaluation of our proposed approach a large collection of photos in a real world environment is required. At the moment we are only aware of one dataset for evaluating logo recognition on photos, the BelgaLogos dataset [5]. However, we feel that this dataset is not ideal to evaluate an approach based on local features as it contains many images containing very small logos. The overall performance on this dataset therefore depends not only on the recognition system but also largely on the capabilities of the visual features. We feel that the dataset is not adequate for evaluating the recognition system rather than the visual features themselves. In addition the BelgaLogos dataset was originally used for logo retrieval rather than for classification and defines only a small number of query images.

Therefore we built and publish the new dataset "FlickrLogos" containing photos depicting logos.[3] We collected logos of 32 different classes by downloading them from Flickr. The specific classes were determined by trial-and-error and mostly chosen by the number of logos we could retrieve by querying the Web service with appropriate queries. In addition we only included logos that have an approximately planar surface.

There are 32 logo classes: Adidas, Aldi, Apple, Becks, BMW, Carlsberg, Chimay, Coca-Cola, Corona, DHL, Esso Erdinger, Fedex, Ferrari, Ford, Foster's, Google, Guiness, Heineken, HP, Milka, Nvidia, Paulaner, Pepsi, Ritter Sport, Shell, Singha, Starbucks, Stella Artois, Texaco, Tsingtao and UPS.

The retrieved images were inspected manually to ensure that the specific logo is actually shown. The whole dataset is split into three disjunct subsets P_1, P_2, and P_3, each containing images of all 32 classes. The first partition P_1 consists of 10 images that were hand-picked such that these consistently show a single logo under various views with as little background clutter as possible. The other two partitions P_2 and P_3 contain 30 images per class. Unlike P_1 these images possibly contain more than one instance of a logo. However, we tried to avoid this where possible. In total both P_2 and P_3 have 960 images showing logos. The subset P_3 for four

[3]The dataset and supplementary material is available at http://www.multimedia-computing.de/flickrlogos

Figure 6: Example detection of the Esso logo: Original image (left), all detected edges of any class (middle left), edges belonging to real class (middle right) and the final detected triangles (right)

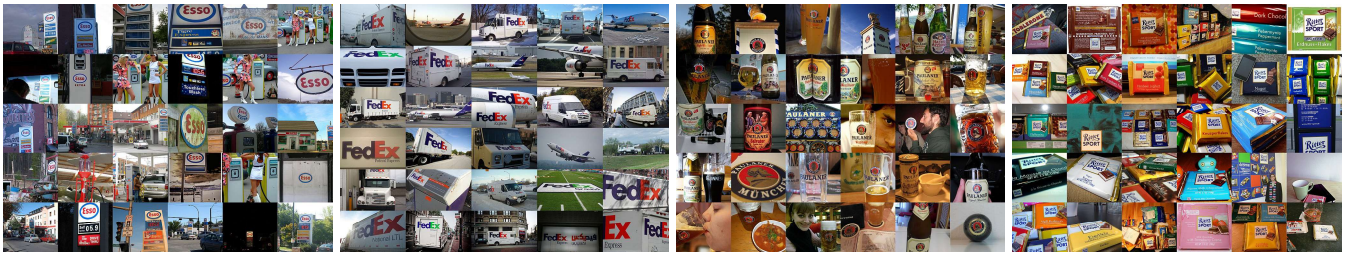

Figure 7: Examples of 4 of the 32 logo classes. Classes from left to right: Esso, Fedex, Paulaner, Ritter Sport

of these classes is shown in Figure 7.

Subset	*Description*	*Images*	*Sum*
P_1	Hand-picked images, single logo, clean background	10 per class	320
P_2	Images showing at least a single logo under various views Non-logo images	30 per class 3000	3960
P_3	Images showing at least a single logo under various views Non-logo images	30 per class 3000	3960

Table 1: Disjunct subsets of our dataset

To facilitate the development of high-precision classifiers the evaluation of their sensitivity on non-logo images is very important. Therefore both partitions P_2 and P_3 include another 3000 images downloaded from Flickr with the queries "building", "nature", "people" and "friends". These images are unlikely to contain logos and complete our dataset. A brief summary of the data subsets is shown in Table 1.

7. EVALUATION

In the following we describe the setup of several experiments and their results.

7.1 Setup

For all training images we performed the matching of image pairs as described in Section 4 to derive the triangles for each image pair. Then we index all triangles in our cascaded index and let our system perform the logo detection on the test set. We tune the parameters using a validation set and report the performance obtained on the test set. In our experiments we choose the subset P_2 of our dataset as validation set and P_3 as test set.

7.2 Parameter sweep

Our proposed system has several parameters that can be tuned for performance. Many of them directly change the number of detections of point pairs and triples. As these numbers may change significantly we need to adaptively select our threshold T_{class} for the decision whether a logo of a given *class* is present or not. Moreover an adaptive threshold further improves the robustness to class imbalance caused by different logo designs and their common placements. Thus we first run the detection on a validation set and refine all class-specific thresholds. Once we obtained the raw detection counts on the validation set we perform a parameter sweep over the threshold T_{class} and recompute precision and recall for each class separately. Precision here means the ratio of recognizing the true logo class once a logo is detected. We determine the optimal threshold for each class by fixing the precision to 0.95 and selecting the corresponding threshold. In case one class does not reach 0.95 precision we select the threshold for the best precision obtained. If in severe cases no single detection was made for a certain class we set the threshold empirically to 20. The final performance is then computed on the test set using those class-specific thresholds that have been refined on the validation set. In other words, in our experiments we choose our desired precision first and then optimize all other parameters for recall. We also report precision on the test set as we fixed the precision only for the validation set.
Note that while we tune parameters defining the quantization of items in the cascaded index we implicitly tune the constraints of our indexing scheme to be restrictive or tolerant. If these constraints get less restrictive false positive detections increase. However, the precision is kept constantly high by the adaptive increase of the decision threshold if more noise is present. Thus, more noise on the validation set implicitly leads to decreasing recall on the test set.

7.3 Parameter optimization

As initial parameters we chose parameter values that have been determined empirically. In each experiment we optimize one parameter at a time and keep all others fixed.

Monte Carlo sampling density. As the recall of our approach is based on Monte Carlo sampling of points, one crucial parameter is the number of random samples, that is the number of queries to the index. To increase recall, we can increase the density of the Monte Carlo sampling at query time. Increasing this ratio will directly affect the number of detected edges and triangles. The number of queries that are issued to the index directly implies the probability of the detection of a certain combination of points. We therefore evaluate our proposed approach by varying the number of queries made to the cascaded index. As the number of queries instantly changes the number of detections we perform the parameter sweep as described in Section 7.2 on the validation set and compute the final performance on the test set. This assures that we compare the recall only for equivalently well performing systems (in terms of precision).

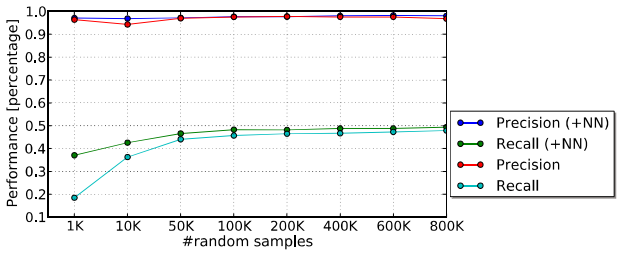

Figure 8: Performance for varying numbers of random samples.

In Figure 8 the results of two type of system are shown. The first system only tests random edges, the second additionally tests the spatial nearest neighbors as described in Section 5.1 (marked as +NN).

One can see from Figure 8 that recall improves with increasing number of random samples. There is a major improvement when e.g. 100K samples are used instead of a few thousands but only little improvement since then. Thus, for further experiments we always perform the nearest neighbour scanning and test 100,000 random edges.

Here we used a visual vocabulary of 2000 visual words derived from gray-scale images. The quantization is done with empirically determined bin sizes of 24° for the three feature orientations and 10° for δ_1 and δ_2.

Visual Vocabularies. The most important part for the discriminativeness of triangle representations are the visual word labels. Therefore we compare different vocabulary sizes.

We compare vocabularies of 1000 to 4096 visual words, up to the maximum number of distinct labels we can pack into our 64-bit integer code. In addition to hessian-affine SIFT features extracted from grayscale versions of the images we also compare results of the usage of 192-dimensional color-SIFT features. As the clustering and the quantization of the descriptors introduce hard boundaries in feature space a smaller vocabulary should be more robust to small changes of descriptor appearance but also possibly yield more false positives during detection. Note that an increasing amount of false positive detections on the validation set leads to smaller recall on the test set (see Section 7.2).

One can see from Figure 9 that including color in descriptors improves performance and larger vocabularies perform better due to fewer false positive detections. Therefore the

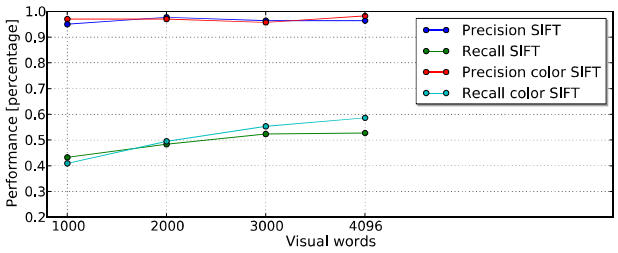

Figure 9: Performance for different sizes of visual vocabularies for both grayscale and color SIFT.

following experiments are performed with the largest color-SIFT vocabulary.

Quantization of triangle shape and feature orientation. First we evaluate how the quantization of the triangle shape affects recall. The shape of a triangle is quantized into two discrete angles δ_1 and δ_2. From Figure 10 (left) we can see we obtain the best recall when the bin size is 15°.

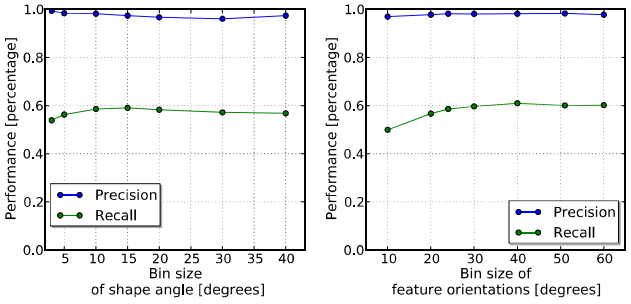

Figure 10: Performance for different quantization bin sizes for angle and feature orientations of triangle shapes.

We further evaluate how the performance is affected if the quantization of the feature orientations changes. While designing our index we observed that including relative orientations of the features significantly lowers the false positive detections on mismatching regions. However, the orientations of features can change quite dramatically depending on the view angle of the logo. Figure 10 (right) also shows the recall for different quantizations bin sizes of feature orientations.

Larger and smaller quantization bin sizes of either shape or feature orientations should lead to overly specific or coarse triangle signatures that then degrade recall. However, both graphs in Figure 10 show rather insignificant differences. We assume this might be caused by the extremely sparse triangle representation and the fact that its discriminativeness is bound by several components instead of a single component. that this is due our extremely sparse triangle representation and the effect Note that we re-train and re-build our index for each shape and orientation parameter value to ensure that the evaluation is not biased towards the matching results derived with a certain parameter configuration.

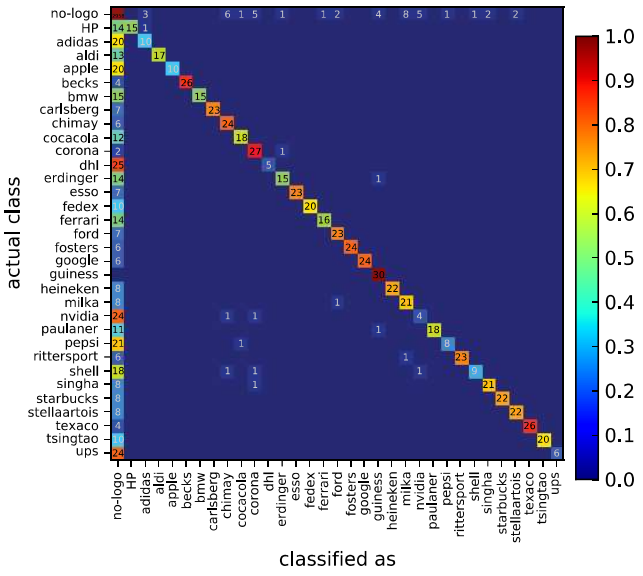

Figure 11: Confusion Matrix

7.4 Summary

After tuning the parameters of the system, the best system has a precision of 0.982 and recall of 0.61. Its edge index holds about 613K keys and its triangle index holds about 11M keys. Figure 11 shows the confusion matrix for all 32 classes of this system which further underlines the high precision of the logo recognition system. The obtained results clearly show that our system produces an extremely low number of false positives, resulting in a high precision rate at the cost of recall. However, one can also observe that some logo classes are easier detected than others. One explanation is that the number of triangles per logo can vary. With fewer triangles, and therefore edges, in the index the chances of detecting a logo declines. Another cause is the varying difficulty of the structure of the logos themselves. Finally, we can observe that even without sophisticated post-processing the detection accuracy based on adaptively thresholded detection counts is very high.

8. CONCLUSIONS

In this paper we propose a highly effective and scalable framework for recognizing logos in an image. At the core of our approach lays a method for encoding and indexing spatial structure that is derived from local features detected in the logo images. We use an automatic method for constructing a model for each logo-class out of multiple training images, which significantly extends our ability to detect logos under varying conditions.

Our logo recognition system is inspired by the bag of visual words approach, but through embedding spatial knowledge into the cascaded index, we have successfully demonstrated that we can get rid of the expansive post-retrieval processing step during which existing image object retrieval systems have to validate the existence of the query object.

For the evaluation of our system, we have constructed and released the FlickrLogo recognition benchmark, which consists of manually labelled logo images, complemented with non-logo images. The benchmark consists of a training, development, and test set with 32 logo-classes. We have tuned and tested our system against the benchmark, and found that we can effectively recognize the different logo classes with a high precision, while maintaining a good recall.

9. ACKNOWLEDGMENTS

This project was funded by Deutsche Forschungsgemeinschaft (DFG).

10. REFERENCES

[1] Y. Avrithis, G. Tolias, and Y. Kalantidis. Feature map hashing: sub-linear indexing of appearance and global geometry. In *Proceedings of the International Conference on Multimedia*, 2010.

[2] A. Bagdanov, L. Ballan, M. Bertini, and A. Del Bimbo. Trademark matching and retrieval in sports video databases. In *Proceedings of the International Workshop on Multimedia Information Retrieval*, ACM, 2007.

[3] O. Chum, M. Perdoch, and J. Matas. Geometric min-Hashing: Finding a (thick) needle in a haystack. *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 17–24, June 2009.

[4] P. Felzenszwalb, R. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part-based models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(9), Sept. 2010.

[5] A. Joly and O. Buisson. Logo retrieval with a contrario visual query expansion. In *Proceedings of the seventeen ACM international conference on Multimedia*, ACM, 2009.

[6] J. Kleban, X. Xie, and W. Ma. Spatial pyramid mining for logo detection in natural scenes. In *IEEE International Conference on Multimedia and Expo*, IEEE, 2008.

[7] D. Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004.

[8] K. Mikolajczyk, T. Tuytelaars, C. Schmid, A. Zisserman, J. Matas, F. Schaffalitzky, T. Kadir, and L. V. Gool. A Comparison of Affine Region Detectors. *International Journal of Computer Vision*, 65(1-2), 2005.

[9] D. Nistér and H. Stewénius. Scalable Recognition with a Vocabulary Tree. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2006.

[10] S. Poullot, M. Crucianu, and S. Satoh. Indexing local configurations of features for scalable content-based video copy detection. In *Proceedings of the First ACM workshop on Large-scale Multimedia Retrieval and Mining*, 2009.

[11] J. Sivic and A. Zisserman. Video Google: a text retrieval approach to object matching in videos. *International Conference on Computer Vision*, 2003.

[12] H. Wolfson and I. Rigoutsos. Geometric hashing: An overview. *IEEE Computational Science & Engineering*, 4(4), 1997.

[13] Z. Wu, Q. Ke, M. Isard, and J. Sun. Bundling features for large scale partial-duplicate web image search. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2009.