Content-based Image Retrieval

Tutorial Image Retrieval

Thomas Deselaers, Henning Müller
Outline

• What is CBIR
• Approaches
  – Continuous approach
  – Discrete approach
• Features for content-based image retrieval
  – Global
  – Local
• Relationships between different approaches
• Available resources
Content-based Image Retrieval

• Two main questions
  – How are the images represented? ➔ features
  – How are the image descriptors compared ➔ similarity measures

• Two main views on the concept of a `feature’:
  – Features are numerical values computed from each image
    • View connected to image classification
    • Ideas and methods from classification and machine learning
  – Features are image properties that are present or absent
    • View connected to textual information retrieval
    • Ideas and methods from text retrieval
Continuous Approach

- Inspired by nearest-neighbour classification
- Each image is represented by a feature vector
- Feature vectors are compared using distance functions
- Images similar to a query image are assumed to be relevant
- Normally: query-by-visual example(s)
Nearest Neighbour

• Voronoi Diagram in a 2D space
• A set of points and the corresponding Voronoi cells
• A Voronoi cell is the area where a point’s nearest neighbour is the seed of the cell

• Standard nearest neighbour search
  
  $X[1] \ldots X[N]$: data set
  $Q$: query
  $\text{mindist}=\infty$
  $\text{bestN}=-1$

  For $n=1:N$:
    
    $d=\text{dist}(q,X[n])$
    if $d<\text{mindist}$:
      $\text{mindist}=d$
      $\text{bestN}=n$
    
  return $\text{bestN}$
**R*-Tree**

- Data structure for multi-dimensional data
- Save data in a tree structure
- Directory nodes (blue)
- Data nodes (red)
- Save minimum bounding rectangles in directory nodes
Index Structures: Complexity

• Linear search
  – Complexity $O(n/C)$, very small overhead

• Bad situation for index structures
  – Large range
  – Strongly overlapping regions
  – Few regions are not accessed
  – Common with high dimensionalities
  – Complexity $O(n/C)$, high overhead

• Good situation
  – Small range
  – Small overlaps
  – Many regions do not have to be read
  – Complexity $O(\log_C(n))$
Nearest Neighbour Search with Index

Init: resultdist=∞
Function SimpleNNQuery(Point q, Address: page):
    page.load()
    if isDataPage(page):
        for x in page.points:
            d=dist(q,x)
            if d<resultdist:
                resultdist=d
                result=x
    else:
        for p in page.childPages:
            if MINDIST(q,p)<resultdist:
                SimpleNNQuery(q,p);

• First path finds arbitrary point
• Search space only slowly reduced
• Many pages unnecessarily read
Example Query

• If the search had started with $p_2$ no page from $p_1$ would have been read
• Clearly non-optimal
Nearest Neighbour Search with Index

Best-First Search

• Avoid recursion
• Instead use priority queue APL (active page list)
  – List which contains directory pages to be processed sorted by priority

• Definition: a page p is active, if and only if
  – p not yet processed
  – Parent of p processed
  – Minimum distance between p and q < current best distance
• Initialisation: APL = [ root ]
• In each step, process best page from APL
  – Data pages: as before
  – Directory pages: check minimum distance to query
Best First Nearest Neighbour Search with Index

Init: apl = [(0.0,root)] // sorted by dist
resultdist=∞
While apl.notEmpty() and apl[0].dist<resultdist:
    page=apl[0].load()
    delete(apl[0])
    if isDataPage(page):
        for x in page.points:
            d=dist(q,x)
            if d<resultdist:
                resultdist=d
                result=x
    else:
        for p in page.childPages:
            h=MINDIST(q,p)
            if h<resultdist:
                apl.insert((h,p))
Example Query with Best First Search

Example of a query with Best First Search, visualized with a tree structure. The root node is at the top, with branches leading to child nodes labeled p1, p2, p21, p22, p23, p11, p12, p13. The search process involves expanding nodes with the lowest cost (APL) and updating the result distance. For instance, the process might involve expanding nodes from root to p1, then p11, p12, p13, and so on, updating the result distance at each step. The final result might be a path from root to a leaf node with the minimum distance.
Best First Search is optimal
Here: a draft of the proof

1. Completeness: It will find the correct NN of a query
   – Every correct algorithm has to access all pages that intersect with the NN sphere of q
   – These pages have MINDIST < resultdist

2. It accesses pages in ascending order from the query
   – The APL is sorted by MINDIST and the algorithm will terminate once it is impossible to find any point closer to q than the current result

3. It will not access a single page with MINDIST larger than the true NN distance
   – Child pages cannot have a MINDIST smaller than its parents
Continuous Approach

• Simple Case:
• Database: \( \{x_1, \ldots x_n, \ldots, x_N\} \)
• Query: \( q \)
• Distance between and \( q, x_n \) \( d(q, x_n) \)
• Sort images \( (x_{n_1} \ldots, x_{n_i} \ldots, x_{n_N}) \)
• Such that \( d(q, x_{n_i}) \leq d(q, x_{n_i+1}) \)
• Holds for all \( x_n \) and \( x_{n_i+1} \)
Continuous Approach

• How to use multiple features
• Each image $x_i$ is represented by $M$ descriptors $x_{nm}$
• Calculate score instead of distance to allow for different weights

$$S_q(x_n) = \exp \left( -\sum_{m=1\ldots M} \omega_m \frac{d_m(x_{nm}, q_m)}{\frac{1}{N} \sum_{n=1\ldots N} d_m(x_{nm}, q_m)} \right)$$
Discrete Approach

• Inspired by textual information retrieval
• Each image is represented by a set of binary features (features may be present (possibly multiple times) or absent)
• Feature is either present or absent
  – Similar to words being absent or present in a document
• Images containing the same (informative) features are assumed to be relevant to a query
• Example: GIFT – GNU Image Finding Tool
Discrete Approach

• GIFT uses TF-IDF (text frequency/inverse document frequency) ranking
  – Reduce the impact of features which occur frequently in the data (comparable to “the” in texts)

• TF: frequency a feature $i$ has in a document $d_j$

$$tf(i, d_j) = \frac{n(i, d_j)}{\sum_k n(k, d_j)}$$

• IDF: measures importance of a term

$$idf(i) = \log \left( \frac{|B|}{|\{d_j : n(i, j) > 0\}|} \right)$$
Discrete Approach

• \( tf \) captures how often a feature occurs in a document
  – Features that occur often in a document describe this document well

• \( idf \) captures how relevant a feature is
  – Features that occur rarely in the full database are important

• Important are those features which are often in one image, but seldom over the full data set
  – Images which share seldom features are relevant with respect to each other
Discrete Approach in GIFT

• In GIFT, 4 different feature sets are considered
  – Global colour
  – Local colour
  – Global texture
  – Local texture

• For all local features a tf-idf-like score is calculated and these are fused as a weighted sum
  – Global features are compared with a histogram intersection

• For many cases, the discrete and the continuous approach can be simulated in the respective other

• In GIFT
  – Images have about 1,500 to 2,000 features
  – Similar images share about 400-500
Discrete Approach: Inverted Files

• Store a mapping from content to location
  – E.g. for each feature a list of images that contain this feature

• Allow for efficient searches even for huge amounts of images
  – $idf$ for each feature can be pre-calculated
  – $tf$ for each feature is stored in the files
  – Allows for searching without accessing the images

• In the continuous approach searching for neighbours is linear to the amount of images, here it is at most linear to the number of features in an image
  – In practice, the features with high impact are processed first and the other features have less influence (Zipf’s Law). Therefore, in practice, this approach is very fast
The inverted file

Feature 1 → Image 5 → Image 7 → Image 1 → Image 25

Feature 2 → Image 1 → Image 17 → Image 3 → ...

Feature n-1 → Image 25 → Image 17 → Image 1 → Image 4

Feature n → Image 4 → Image 5 → Image 6 → ...

Image 2 → Image 17 → Image 12 → Image 3
Inverted Files

• Access feature by feature instead of image by image
• Extremely **fast** access for rare features
• Efficient for **sparsely populated** spaces
Visual Properties of Images

- Colour
- Texture
- Shapes
- Image parts
- Complete image
- Meta data
- Textual labels/captions/annotations
Features for Content-based Image Retrieval

• Global Descriptors
  – Colour histograms
  – Texture Features
  – Shape Features

• Local Descriptors
  – Direct approach
  – Patch-histograms / bag-of-visual words
  – SIFT features
Global Descriptors

- Capture a property of the image with few values
- Describing the image in its entirety
- E.g.
  - Which colours occur in the image?
  - Is the image of high contrast?
  - Is the image bright/dark?
Colour histograms

• Describe the distribution of colours in an image
• Discard spatial information

1. Quantise colour space
2. Count which colour occurs how often
Colour Histograms: Example

RGB colour space

HSV colour space

Colour Histograms: Example

RGB colour space

HSV colour space

Texture Features

“Texture refers to the properties held and sensations caused by the external surface of objects received through the sense of touch.”

Texture in image processing:
• Different definitions
• Different representations
Tamura Features

- Proposed by Tamura [1978]
  - Features corresponding to human perception
  - Examined 6 different features, found 3 to correspond strongly to human perception
  - **Coarseness** – coarse vs. fine
  - **Contrast** – high vs. low
  - **Directionality** – directional vs. non-directional
  - Line-likeness – line-like vs. non-line-like
  - Regularity – regular vs. irregular
  - Roughness – rough vs. smooth
Tamura Features

• Originally one value per feature per image was determined

• Create a descriptor by
  1. Calculate coarseness, contrast, and directionality in a local neighbourhood for each pixel
  2. Create a joint histogram over these values
Gabor Features

- Obtain several values per pixel denoting spatial frequencies and directions
Gabor Features

- Windows Fourier transform with Gaussian window function:
  \[ g_\alpha(t) = \frac{1}{2\sqrt{\pi\alpha}} e^{-\frac{t^2}{4\alpha}} \]

\[
G_f(\omega, \tau) = \int_{-\infty}^{+\infty} f(t) g_\alpha(t - \tau) e^{i\omega t} dt
\]

\[
= e^{-i\omega \tau} (f(\tau) \ast h(\tau))
\]

- Gabor features:
  - Yield one value per pixel (for grey value images)
  - For colour images use a transformed HSV space

- Compute global (histogram) or local features from Gabor responses
Gray-Level Co-Occurrence Matrices

- Statistical descriptor for texture properties of an image by comparing neighbouring pixels
  - Direction and distance
  - Extract features from matrix
    - Entropy
    - Contrast
    - Correlation
    - ...
Histogram Comparison

• $L_p$ - distances

$$d(h, h') = \left( \sum_i (h_i - h'_i)^p \right)^{\frac{1}{p}}$$

• Jensen Shannon Divergence

$$d_{JSD}(h, h') = \sum_i h_i \log \frac{2h_i}{h_i + h'_i} + h'_i \log \frac{2h'_i}{h'_i + h_i}$$
Pixel Values as Features

• Most straightforward
• Scale all images to a common size
• Compare pixels using e.g. Euclidean distance

• Multi-scale Representations:
Direct Comparison of Images

- Pixel-wise:

\[ d(A, B) = \sqrt{\sum_{i=1}^{I} \sum_{j=1}^{J} (A(i, j) - B(i, j))^2} \]
Image Distortion Model

- Allow for small local displacements
- Computationally efficient
Shape: GIST descriptor

- Describe the shapes occurring in an image with one descriptor
  - Subdivide image in 4×4 sub images
  - Calculate Gabor responses in each of these
  - Create histograms of Gabor responses in each sub image

- Has been shown to be helpful to distinguish
  - Naturalness
  - Openness
  - Roughness
  - Expansion
  - Ruggedness
Shape: GIST descriptor

GIST descriptor
Oliva and Torralba, IJCV 2001

Slide by James Hays and Alexei Efros
Local Descriptors

• Definition:
  • Features extracted from local regions from the image
  • E.g. patches, SIFT features, local colour histograms, ...
  • Extraction position determined by interest points
  • Known to achieve good results in many tasks

• Active field of research in object recognition, detection, scene classification, image annotation, and more recently: image retrieval
Local Descriptors: Interest Points
Local Descriptors: Patches

• Extract patches from the image
• Apply a PCA transformation to reduce dimensionality

• Can easily handle colour and gray value images
• All methods from invariant image comparison can be applied at patch level
Local Descriptors: SIFT

- Store a histogram of gradients in local areas
- SIFT = Scale Invariant Feature Transform

- Leading to 128-dimensional feature vectors
- Have been shown to perform well in many tasks

Figure by T. Weyand
Local Descriptors: Direct Retrieval

Database images → Extraction of local features → KD tree → Ranking the database images by number of retrieved nearest neighbor features → Querying the KD tree with each patch of the query image
Histograms of Local Descriptors

Figure by T. Weyand
Histograms of Local Descriptors
Features for CBIR: Performance Evaluation
Correlation Between Features

1: colour histogram
2: MPEG7: colour layout
3: LFSIFThistogram
4: LFSIFTsignature
5: LFSIFTglobal search
6: MPEG7: edgehistogram,
7: Gaborvector
8: Gaborhistograms
9: grayvaluehistogram
10: global texturefeature,
11: inv. Feat histo color
12: Lfpatchesglobal
13: LFpatcheshistogram
14: LFpatchesssignature,
15: inv. feat histo rel
16: MPEG7: scalablecolor,
17: Tamura
18: 32x32 image
19: Xx32image.
Correlation Between Features
Combining Features

- Manually tuned
  - Have an ‘expert’ find a proper set of parameters
- Heuristic to capture different image properties
- Combination to reflect human perception
- Combination to obtain optimal performance (given a set of training queries)
Combining Features to Capture Different Image Properties

- Given the result from the correlation analysis, first choose a simple feature.
- Then add features which have low correlation.

Color Histogram: 50.5% MAP
+ Global Texture Features: 49.5% MAP
+ Tamura Texture Histogram: 51.2% MAP
+ Image Thumbnails: 53.9% MAP
+ Patch Histograms: 55.7% MAP
Combining Features
Reflecting Human Perception

• Comparison of Human perception of image similarity and low-level image descriptors:

<table>
<thead>
<tr>
<th>Feature</th>
<th>Study 1</th>
<th></th>
<th>Study 2</th>
<th></th>
<th>Study 3 (AB)</th>
<th></th>
<th>Study 3 (BA)</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1 32×32 image</td>
<td>0.230*</td>
<td>0.0009</td>
<td>0.162*</td>
<td>0.0009</td>
<td>0.103</td>
<td>0.066</td>
<td>0.070</td>
<td>0.148</td>
</tr>
<tr>
<td>2 Colour histogram</td>
<td>0.269*</td>
<td>0.0009</td>
<td>0.160*</td>
<td>0.0009</td>
<td>0.062</td>
<td>0.168</td>
<td>0.076</td>
<td>0.118</td>
</tr>
<tr>
<td>3 GTF</td>
<td>0.015</td>
<td>0.3027</td>
<td>0.047</td>
<td>0.0569</td>
<td>0.055</td>
<td>0.199</td>
<td>0.079</td>
<td>0.128</td>
</tr>
<tr>
<td>4 Monomial IFH</td>
<td>0.232*</td>
<td>0.0009</td>
<td>0.162*</td>
<td>0.0009</td>
<td>0.043</td>
<td>0.2478</td>
<td>0.055</td>
<td>0.1888</td>
</tr>
<tr>
<td>5 Relational IFH</td>
<td>0.185*</td>
<td>0.0009</td>
<td>0.090**</td>
<td>0.0029</td>
<td>-0.037</td>
<td>0.315</td>
<td>-0.047</td>
<td>0.271</td>
</tr>
<tr>
<td>6 Tamura histogram</td>
<td>0.184*</td>
<td>0.0009</td>
<td>0.107*</td>
<td>0.0009</td>
<td>0.021</td>
<td>0.343</td>
<td>0.042</td>
<td>0.262</td>
</tr>
<tr>
<td>7 4,096 bin patch histo</td>
<td>0.295*</td>
<td>0.0009</td>
<td>0.237*</td>
<td>0.0009</td>
<td>0.214*</td>
<td>0.0009</td>
<td>0.267*</td>
<td>0.0009</td>
</tr>
<tr>
<td>8 65,536 bin patch histo</td>
<td>0.281*</td>
<td>0.0009</td>
<td>0.245*</td>
<td>0.0009</td>
<td>0.121</td>
<td>0.015</td>
<td>0.143</td>
<td>0.008</td>
</tr>
</tbody>
</table>
Combining Features
Reflecting Human Perception

- Combinations of image descriptors to achieve best compliance with human perception

<table>
<thead>
<tr>
<th>Weights from study</th>
<th>Study 1</th>
<th>Study 2</th>
<th>Study 3 (AB)</th>
<th>Study 3 (BA)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.393648</td>
<td>0.244613</td>
<td>0.201997</td>
<td>0.207113</td>
</tr>
<tr>
<td>2</td>
<td>0.257257</td>
<td>0.269415</td>
<td>0.265245</td>
<td>0.294700</td>
</tr>
<tr>
<td>3 (AB)</td>
<td>0.256781</td>
<td>0.167512</td>
<td>0.298475</td>
<td>0.350045</td>
</tr>
<tr>
<td>3 (BA)</td>
<td>0.245353</td>
<td>0.163617</td>
<td>0.290779</td>
<td>0.355902</td>
</tr>
</tbody>
</table>

$r$ values ($p = 0.000999$)
Combining Features
Reflecting Human Perception
Combining Features to Obtain Optimal Performance

- Consider retrieval to be a two-class problem
- Classify images as `relevant’ or `irrelevant’
- Use trained classifiers
- **Requires training data**
  - Can be obtained from experts
  - Or relevance feedback (more on this later)
Combining Features: Maximum Entropy/Log-Linear Models

• Learn a model to predict probabilities for an image to be relevant

\[ p(\oplus|Q, X) = \frac{\exp \left[ \sum_i \lambda_\oplus f_i(Q, X) \right]}{\sum_{k\in\{\oplus, \ominus\}} \exp \left[ \sum_i \lambda_k f_i(Q, X) \right]} \]

• \( f_i(Q, X) \) is a function describing some similarity/dissimilarity of the query Q and image X

• \( \lambda_{\oplus i} \) are the parameters to be trained
Trained Lambdas for medical retrieval

1. En
2. Fr
3. Ge
4. Colour
5. Gray
6. GTF
7. IFH
8. Tamura
9. 32x32
10. PatchHisto
Combining Features: Support Vector Machines

• Train a two-class SVM, +1= relevant, -1= irrelevant

\[ X \mapsto \hat{c}(X) = \text{sgn} \left\{ \sum_{v_i \in S} \alpha_i K(X, v_i) + \beta \right\} \]

• Since SVMs sometimes have problems with non-uniformly distributed classes, subsample training samples to have approximately the same amount of positive and negative samples
Available Resources

- **Image Retrieval Systems:**
  - GIFT – GNU Image Finding Tool
    - Full image retrieval system
    - Following the discrete approach
  - FIRE – Flexible Image Retrieval Engine
    - [http://www-i6.informatik.rwth-aachen.de/~deselaers/fire/](http://www-i6.informatik.rwth-aachen.de/~deselaers/fire/)
    - Research image retrieval system
    - Developed to allow for easy extension
    - Following the Continuous approach
  - openCV – computer vision library
    - [http://sourceforge.net/projects/opencvlibrary/](http://sourceforge.net/projects/opencvlibrary/)
    - Implements may image processing operations
    - E.g. face detection and recognition, feature extraction
Available Resources

Data sets

• IAPR TC 12 Dataset
  – Used in ImageCLEF since 2006
  – 20,000 images with text in English, German, Spanish
  – Available from www.imageclef.org/photodata

• ImageCLEFmed Datasets
  – Nearly 70,000 images
  – From 50,000 medical cases
  – Images and text
Available Resources

Data sets

• Corel dataset
  – Widely used but not freely available
  – Questionable for evaluation
  – Different sets are in use in the CBIR community
    • Between 1,000 and 200,000 images

• MSRC dataset
  – Provided by Microsoft Research, Cambridge, UK
  – 4320 images from 45 classes

• Flickr
  – Difficult to use for evaluation although large number of images
References

• Overview

• Features:
  – SIFT Descriptors
  – Patch Histograms
  – GIST
References

• Continuous/Discrete
  – FIRE: see features

• Datasets
  – www.imageclef.org

• Learning Feature Combinations

• Efficient NN search