Stefan Rüger

Multimedia Information Retrieval

Material for RuSSIR 2010, Voronezh, Russia

Chapters 1 and 2 are an *excerpt* from lecture notes

S Rüger: *Multimedia information retrieval.* Lecture notes in the series Synthesis Lectures on Information Concepts, Retrieval, and Services. Morgan and Claypool Publishers, 171 pages, 74 figures, 4 videos, 1 soundfile, ISBN paper 978-1-60845-097-8, ISBN ebook 978-1-60845-098-5, 2010

If you find this resource useful, please consider buying the full lecture notes. They sell for around 20 USD as e-book from the publisher (and not much more in print from booksellers).

Chapter 3 is a preprint of

S Little, A Llorente and S Rüger: *An overview of evaluation campaigns in multimedia retrieval.* In H Müller, P Clough, T Deselaers and B Caputo (Eds): *ImageCLEF — Experimental Evaluation of Visual Information Retrieval,* Springer, in preparation, 2010

Milton Keynes, 2 August 2010

Table of contents

Chapter 1

Basic Multimedia Search Technologies

The current best practice to index multimedia collections is via the generation of a library card, ie, a dedicated database entry of metadata such as author, title, publication year and keywords. Depending on the concrete implementation, these can be found with SQL database queries, text-search engines or XML queries, but all these search modes are based on text descriptions of some form and are agnostic to the structure of the actual objects they refer to, be it books, CDs, videos, newspaper articles, paintings, sculptures, web pages, consumer products etc. The first section of this chapter is about the traditional metadata driven retrieval.

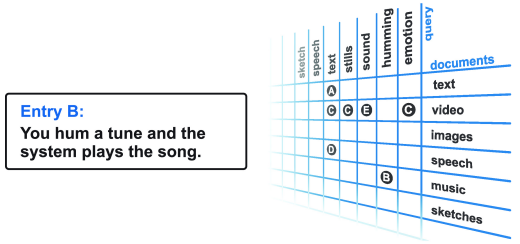


Figure 1.1: New search engine types

The text column of the matrix of Figure 1.1 is underpinned by text search technology and requires the textual representation of the multimedia objects, an approach that I like to call *piggy-back text retrieval*. Other approaches are based on an automatic classification of multimedia objects and on assigning words from a fixed vocabulary. This can be a certain camera motion that can be detected in a video (zoom, pan etc); a genre for music pieces such as jazz or classics; a generic scene description in images such as inside/outside, people, vegetation, landscape, grass, city-view etc or specific object detection like faces and cars etc. These approaches are known as *feature classification* or *automated annotation*.

The type of search that is most commonly associated with multimedia is *content-based*: the basic idea is that still images, music extracts, video clips themselves can be used as queries and that the retrieval system is expected to return 'similar' database entries. This technology differs

most radically from the thousands-year-old library card paradigm in that there is no necessity for metadata at all. In certain searches, there is the desire to match not only the general type of scene or music that the the query represents but instead one and only one exact multimedia object. For example, you take a picture of a painting in a gallery and submit this as a query in the hope of receiving the gallery's (or otherwise) record about this particular painting. In this case, you use an image of the real world to obtain a link into the electronic world and not to see a variant or otherwise similar exhibit. The underlying technology is sometimes called *fingerprinting* or *known-item search*.

1.1 Metadata Driven Retrieval

Metadata are pieces of information about a multimedia object that are not strictly necessary for working with it, but that are useful to

- *describe* resources so they can be indexed, classified, located, browsed and found
- *store technical* information, such as data formats and compression schemes
- *manage* resources such as their rights or where they are currently located
- *record preservation* actions
- *create usage* trails, eg, which section of a video has been watched how many times

All of these aspects are relevant for multimedia information retrieval. Undoubtedly, the first type of so-named *descriptive metadata* has been deployed since thousands of years to keep track of documents and objects: the old Sumerians used incipits to form surrogate summaries, which they could browse, while later document titles were invented and are still the most important form of metadata. Library cards such as the one in Figure 1.2 have for centuries recorded metadata such as title, author, year of publication, classification tag, location in library, publisher with their address, and so on.

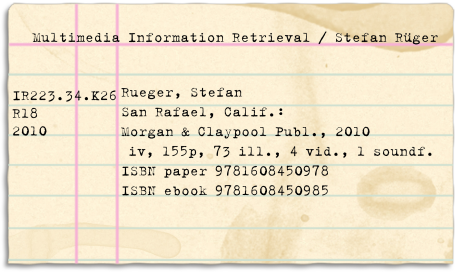


Figure 1.2: What library cards looked like

The printed out library card has become obsolete, and the most important technique for multimedia retrieval based on metadata has become structured-document retrieval. Document structure is increasingly often expressed in the XML schema language[1] of the World Wide Web Consortium[2]. Lalmas (2009) covers XML Retrieval in depth.

[1] http://www.w3.org/XML/Schema
[2] http://www.w3.org

When using metadata, it becomes apparent how vital it is that they can be exchanged, especially so in digital libraries. One of the oldest and most widespread standards for bibliographic metadata is MARC, which stands for machine-readable cataloguing. It is an elaborate standard with several hundred entries that are subject to the Anglo-American Cataloguing Rules in its current version AARC2R. Owing to the complexity of these rules, only trained specialists are able to create a MARC record. However, once created, a record can then be shared by libraries all over the world. It is the use of standards such as MARC that have made it possible to create and search the Worldcat union catalogue at OCLC of currently more than 150 million entries in 470 languages[3]. MARCXML is the corresponding XML language that expresses a MARC record in XML.

On the other side of the spectrum of complexity resides the Dublin Core standard named after the city in Ohio, US, where the first meeting took place in 1995. Its almost trivial structure — only 15 elements such as title, creator, subject, description and date each of which is optional and may be repeated — make it very easy for everyone to instantly create metadata for their multimedia objects. The Dublin Core metadata initiative have produced a comprehensive and comprehensible guide Using Dublin Core[4]. Figure 1.3 shows an XML representation of simple Dublin Core metadata for this book.

```
<?xml version="1.0"?>
<metadata
    xmlns="http://w.x.y/z/"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://w.x.y/z/ http://w.x.y/z/schema.xsd"
    xmlns:dc="http://purl.org/dc/elements/1.1/">

    <dc:title> Multimedia information retrieval </dc:title>
    <dc:description>
        This book covers the process of finding multimedia documents, how to
        build multimedia search engines, and guides through related research.
    </dc:description>
    <dc:creator> Stefan Rueger </dc:creator>
    <dc:date> 2010 </dc:date>
    <dc:publisher> San Rafael, Ca: Morgan & Claypool Publishers </dc:publisher>
    <dc:identifier>
        http://dx.doi.org/10.2200/S00244ED1V01Y200912ICR010
    </dc:identifier>
    <dc:identifier> ISBN paper 978-1-60845-097-8 </dc:identifier>
    <dc:identifier> ISBN ebook 978-1-60845-098-5 </dc:identifier>
</metadata>
```

Figure 1.3: Dublin core record for this book

Metadata are not always stored outside the documents themselves. Many multimedia documents contain provisions for storing metadata, especially technical metadata. For example, the *tagged image file format* TIFF contains an internal directory structure to hold many images in one file; each image can have its own description in terms of size, bit sampling information, compression and so on. Although all images are rectangular, the format allows for a vector-based cropping path for outlines or image frames. TIFF was initially conceived for scanners but is now a good format

[3] http://www.oclc.org/worldcat/statistics/charts/languagecloud.htm as of Nov 2009
[4] http://dublincore.org/documents/usageguide

for any digital image no matter what the source, be it a screenshot, a digital camera, a grey-level scan, a photo-editing programme or a medical imaging device. The TIFF standard is extensible and some companies have created proprietary image file formats on the back of TIFF describing using custom compression schemes or additional tags. Nikon's NEF format is one such example. Most digital libraries chose to store images in TIFF with lossless compression, which is also the best format to process images in photo-editors — a lossy file format such as JPEG would otherwise lose some of its quality in each store-reload step.

Most camera manufacturers have agreed on a way of incorporating metadata into the files that the camera produces; this is the so-named *exchangeable image file format* EXIF that records mostly technical (as opposed to descriptive) metadata, including a thumbnail of the image, see Figure 1.4 for an example for the kind of information that can be stored in it. EXIF metadata can be stored in TIFF.

Thumbnail

Model	Nikon D200
Exposure	Auto exposure, aperture priority, pattern metering
Exposure	1/640, f8.0, +0.78eV, 100 ISO, 15mm focal length
Flash	did not fire
Orientation	top_left
Date	22.07.2007 12.45 BST
GPS data	52 ° 03'27.6"N, 0 ° 43'30.0"W, 71m
Media	3872x2592 image/jpeg Baseline Fine (4:1)
Bits/channel	8 8 8

Figure 1.4: Some EXIF metadata for a JPEG photograph

Other file formats associate different metadata standards with them. Adobe Systems, for example, pushes its *extensible metadata platform* XMP that is very versatile and includes, amongst other things, rights management. Originally developed for Adobe's Portable Document Format, its structure can be applied to images, audio and video files alike. In 2008 the International Press Telecommunications Council released a photo metadata standard based on XMP. The Library of Congress' Network Development and MARC Standards Office promotes a different XML schema for a set of technical data elements required to manage digital image collections. The schema, also known as *Metadata for Images in XML* (MIX), provides an alternative format for interchange and/or storage.

The most salient metadata standard for multimedia is MPEG-7, which was developed by MPEG (Moving Picture Experts Group)[5] and is a format for description and search of audiovisual resources. MPEG-7 also contains low-level descriptors that can be used with content-based queries. As such, MPEG-7 is the ideal choice of meta-data for audiovisual search engines. MPEG-7 also proposes a set of features that can be used for this purpose, but the standard does not suggest nor prescribe how content-based queries should be carried out. Section 1.3 and Chapter 2 will go into more detail how content-based queries can be processed.

MPEG-7 descriptions cater for still images, graphics, 3d models, audio, speech, video, and information about how these elements are composed in a multimedia presentation. They care about the *content* of the multimedia object on various levels, from low-level machine-extractable features, to high-level human annotations, but they do *not* engage with the way the content is

[5] Other MPEG standards include MPEG-1 and MPEG-2, which enabled video on CD-ROM, MP3, Digital Audio Broadcasting and Digital Television through compression; MPEG-4, which is the multimedia standard to support animation and interactivity; MPEG 21, which is a metadata standard for content delivery and rights management

represented: physical world objects such as a drawing on paper can have an MPEG-7 description in the very same way as a compressed digital TIFF image.

As with other metadata standards, there is not a single one "right" MPEG-7 file for a particular multimedia object. MPEG-7 allows, and encourages, different levels of granularity in the description depending on the application type. Although MPEG-7 puts a great emphasis on content description, more traditional metadata such as media type, rights information, price and parental ratings can also be included. The three main elements of MPEG-7 are:

- *Descriptors* to define the syntax and the semantics of each feature, and *description schemes* to specify the relationships between their components, which in turn may be descriptors and description schemes
- *Description definition language* to define the syntax of the MPEG-7 description tools and to allow the creation of new description schemes and descriptors
- *System tools and reference implementations* to support binary coded representation for efficient storage and transmission, multiplexing of descriptions, synchronization of descriptions with content, management and protection of rights

Figure 1.5 shows an MPEG-7 encoding of the results of an algorithm that predicts the presence of *tree*, *field* and *horses* with various levels of confidence. Can you spot those words? More information about MPEG-7 can be found at the MPEG[6] website and the MPEG-7 Consortium website[7].

Witten et al (2010, Chapter 6) give a deeper insight into metadata in general and their use within digital libraries, but see also (Hillmann and Westbrooks, 2004; Gilliland-Swetland, 1998; Intner et al, 2006; Lagoze and Payette, 2000; Zeng and Qin, 2008; Messing et al, 2001).

1.2 Piggy-back Text Retrieval

Amongst all media types, TV video streams arguably have the biggest scope for automatically extracting text strings in a number of ways: directly from closed-captions, teletext or subtitles; automated speech recognition on the audio and optical character recognition for text embedded in the frames of a video. Full text search of these strings is the way in which most video retrieval systems operate, including Google's latest TV search engine[8] or Blinkx-TV[9]. This technology existed in some research labs much earlier: for example, Físchlár-TV[10] was an experimental web-based video recorder system, developed and maintained 1999–2004 by Dublin City University's Centre for Digital Video Processing. A final-year student project at Imperial College London that indexed videos through teletext received a national prize in the year 2000[11].

In contrast to television, for which legislation normally requires subtitles to assist the hearing impaired, videos stored on DVD don't usually have textual subtitles. They have *subpicture* channels for different languages instead, which are overlaid on the video stream. This requires the extra step of optical character recognition, which can be done with a relatively low error rate owing to good quality fonts and clear background/foreground separation in the subpictures. In general, teletext has a much lower word error rate than automated speech recognition. In practice, it turns out that

[6] http://www.chiariglione.org/mpeg
[7] http://mpeg7.nist.gov
[8] http://video.google.com
[9] http://www.blinkx.tv
[10] http://www.cdvp.dcu.ie/aboutfischlar.html
[11] http://www.setawards.org/previous_winners.vc

```
<?xml version="1.0" encoding="UTF-8"?>
<Mpeg7 xsi:schemaLocation="urn:mpeg:mpeg7:schema:2004 ./davp-2005.xsd"
xmlns="urn:mpeg:mpeg7:schema:2004"
xmlns:mpeg7="urn:mpeg:mpeg7:schema:2004"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <Description xsi:type="ContentEntityType">
    <MultimediaContent xsi:type="AudioVisualType">
      <AudioVisual>
        <StructuralUnit href="urn:x-mpeg-7-pharos:cs:AudioVisualSegmentationCS:root"/>
        <MediaSourceDecomposition criteria="kmi image annotation segment">
          <StillRegion>
            <MediaLocator>
            <MediaUri>http://server/location/Zion_National_Park_392099.jpg</MediaUri>
            </MediaLocator>
            <StructuralUnit href="urn:x-mpeg-7-pharos:cs:SegmentationCS:image"/>
            <TextAnnotation type="urn:x-mpeg-7-pharos:cs:TextAnnotationCS:
                image:keyword:kmi:annotation_1" confidence="0.87">
              <FreeTextAnnotation>tree</FreeTextAnnotation>
            </TextAnnotation>
            <TextAnnotation type="urn:x-mpeg-7-pharos:cs:TextAnnotationCS:
                image:keyword:kmi:annotation_2" confidence="0.72">
              <FreeTextAnnotation>field</FreeTextAnnotation>
            </TextAnnotation>
            <TextAnnotation type="urn:x-mpeg-7-pharos:cs:TextAnnotationCS:
                image:keyword:kmi:annotation_3" confidence="0.63">
             <FreeTextAnnotation>horses</FreeTextAnnotation>
            </TextAnnotation>
          </StillRegion>
        </MediaSourceDecomposition>
      </AudioVisual>
    </MultimediaContent>
  </Description>
</Mpeg7>
```

Figure 1.5: MPEG-7 example for automated text annotation of an image

this does not matter too much as query words often occur repeatedly in the audio - the retrieval performance degrades gracefully with increased word error rates.

Web pages afford some context information that can be used for indexing multimedia objects. For example, words in the anchor text of a link to an image, a video clip or a music track, the file name of the object itself, metadata stored within the files and other context information such as captions. A subset of these sources for text snippets are normally used in web image search engines.

Some symbolic music representations allow the conversion of music into text, such as MIDI files which contain a music representation in terms of pitch, onset times and duration of notes. By representing differences of successive pitches as characters one can, for example, map monophonic music to one-dimensional strings as demonstrated in Figure 1.6. The numbers are midi representations of the pitch of the score. We just record the differences of successive notes (as only a few gifted ones have the power of absolute pitch) and convert the difference to a letter. Zero is Z, 1 is upper case *A*, -1 is lowercase *a*, and so on. The process in Figure 1.6 glides a window over the

music piece and records "musical words" of a certain length. These words then act as surrogate text for music representation; query by humming can thus be treated as a text retrieval problem. Downie and Nelson (2000) were the first to map music to text in this way. Later Doraisamy (2005) deployed this principle and extended it to both polyphonic music, where more than one note is present, and rhythm, ie, music retrieval by tapping. She built a query by humming system that is based on the reduction of music to text followed by text search: you can watch a demonstration video that lays open the individual steps of her algorithm by clicking on Figure 1.6 or by going directly to http://people.kmi.open.ac.uk/stefan/mir-book/movie0008-audio.wmv.

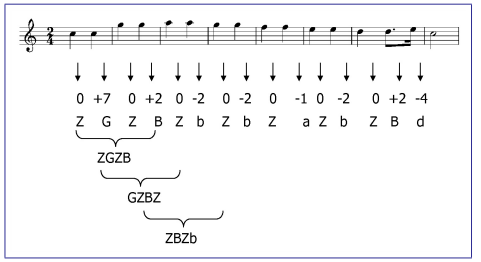


Figure 1.6: Music Retrieval by Humming (click frame to play demo video)

A large range of different text matching techniques can be deployed, for example, the edit distance of database strings with a string representation of a query. The edit distance between two strings computes the smallest number of deletions, insertions or character replacements that is necessary to transform one string into the other. In the case of query-by-humming, where a pitch tracker can convert the hummed query into a MIDI-sequence (Birmingham et al, 2006), the edit distance is also able to deal gracefully with humming errors.

1.3 Content-based Retrieval

Content-based retrieval uses characteristics of the multimedia objects themselves, ie, their content to search and find multimedia. Its main application is to find multimedia by examples, ie, when the query consists not of words but of a similar example instance.

One of the difficulties of matching multimedia is that the parts the media are made from are not necessarily semantic units. Another difficulty comes about by the sheer amount of data with little apparent structure. Look at the black and white photograph of Figure 1.7, for example. It literally consists of millions of pixels, and each of the pixels encodes an intensity (one number between 0=black and 255=white) or a colour (three numbers for the red, green and blue colour channel, say). One of the prime tasks in multimedia retrieval is to make sense out of this sea of numbers.

The key here is to condense the sheer amount of numbers into meaningful pieces of information, which we call *features*. One trivial example is to compute an intensity histogram, ie, count which proportion of the pixels falls into which intensity ranges. In Figure 1.7 I have chosen 8 ranges, and the histogram of 8 numbers conveys a rough distribution of brightness in the image.

Figure 1.8 shows the main principle of *query-by-example*; in this case, the query is the image of an ice-bear on the left. This query image will have a representation as a certain point (o) in feature space. In the same way, every single image in the database has its own representation (x) in the

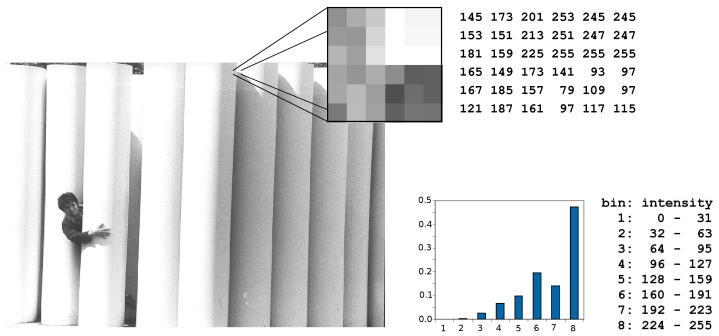


Figure 1.7: Millions of pixels with intensity values and the corresponding intensity histogram

same space. The images, whose representations are closest to the representation of the query are ranked top by this process. The two key elements really are features and distances. Our choice of feature space and how to compute distances has a vital impact on how well visual search by example works.

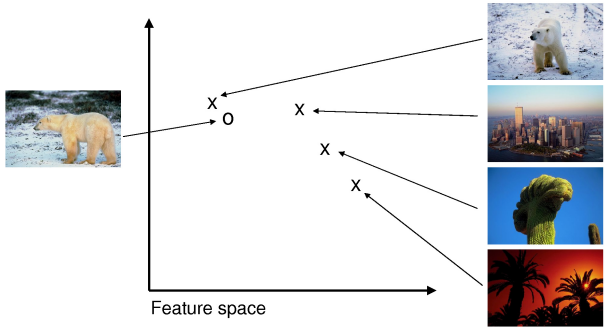


Figure 1.8: Features and distances

Features and distances are a vital part of content-based retrieval and so is the ability to efficiently find nearest neighbours in high-dimensional spaces. This is the content of Chapter 2 that treats content-based retrieval in depth. Lew et al (2006) and Datta et al (2008) have published overview articles on content-based retrieval.

1.4 Automated Image Annotation

Two of the factors limiting the uptake of digital libraries for multimedia are the scarcity and the expense of metadata for digital media. Flickr[12], a popular photo sharing site, lets users upload, organise and annotate their own photographs with tags. In order to search images in Flickr, little more than user tags are available with the effect that many photographs are difficult or impossible to find. The same is true for the video sharing site YouTube[13]. At the other end of the spectrum are commercial sites such as the digital multimedia store iTunes[14], which sells music, movies, TV shows, audio-books, podcasts and games. They tend to have sufficiently many annotations as the commercial nature of iTunes makes it viable to supply metadata to the required level of granularity. While personal photographs and videos do not come with much metadata except for the data that the camera provides (time-stamp and technical data such as aperture, exposure, sensitivity and focal length), a whole class of surveillance data carries even less incentive to create metadata manually: CCTV recordings, satellite images, audio recordings in the sea and other sensor data. The absence of labels and metadata is a real barrier for complex and high-level queries such as "what did the person with a red jumper look like who exited the car park during the last 6 hours in a black Volvo at high speed".

One way to generate useful tags and metadata for multimedia objects is to involve a community of people who do the tagging collaboratively. This process is also called folksonomy, social indexing or social tagging. Del.icio.us[15] is a social bookmarking system and a good example for folksonomies. Similarly, the ability of Flickr to annotate images of other people falls also into this category. Von Ahn and Dabbish (2004) have invented a computer game that provides an incentive (competition and points) for people to label randomly selected images. All these approaches tap into "human computing power" for a good cause: the structuring and labelling of multimedia objects. Research in this area is still in the beginning, and it is by no way clear how to best harness the social power of collaborative tagging to improve metadata for, and access to, digital museums and libraries.

Another way to bridge the semantic gap is to try to assign simple words automatically to images solely based on their pixels. Methods attempting this task include dedicated machine vision models for particular words such as "people" or "aeroplane". These individual models for each of the words can quickly become very detailed and elaborate: Thomas Huang of the University of Illinois at Urbana Champaign once joked during his keynote speech at CIVR 2002 that in order to enable a system to annotate 1,000 words automatically, it was merely a case of supervising 1,000 corresponding PhD projects!

Automated annotation can be formulated in more general terms of machine translation as seen in Figure 1.9. The basic idea is to first dissect images into blobs of similar colour and then use these blobs as "words" of a visual vocabulary. Given a training set of annotated images a correlation between certain words and certain blobs can then be established in a similar way to correlations between corresponding words of two different languages using a parallel corpus (for example, the official records of the Canadian Parliament in French and English). Duygulu et al (2002) created the first successful automated annotation mechanisms based on this idea.

However, the most popular and successful *generic* approaches are based on classification techniques. This normally requires a large training set of images that have annotations from which one

[12] http://flickr.com
[13] http://www.youtube.com
[14] http://www.apple.com/itunes
[15] http://del.icio.us

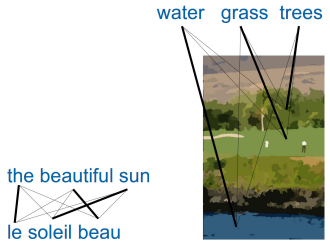


Figure 1.9: Automated annotation as machine translation problem

can extract features and correlate these with the existing annotations of the training set. For example, images with tigers will have orange-black stripes and often green patches from surrounding vegetation, and their existence in an unseen image can in turn bring about the annotation "tiger". As with any machine learning method, it is important to work with a large set of training examples. Figure 1.10 shows randomly selected, royalty free images from the Corel's Gallery 380,000 product that were annotated with *sunset* (top) and *city* (bottom). Each of these images can have multiple annotations: there are pictures that are annotated with *both* sunset and city, and possibly other terms.

Automated algorithms build a model for the commonalities in the features of images, which can later be used for retrieval. One of the simplest machine learning algorithms is the Naïve Bayes formula,

$$P(w|i) = \frac{P(w,i)}{P(i)} = \frac{\sum_j P(w,i|j)P(j)}{\sum_j P(i|j)P(j)}$$

$$= \frac{\sum_j P(i|w,j)P(w|j)P(j)}{\sum_j \sum_w P(i|w,j)P(w|j)P(j)},$$

where j are training images, w are word annotations and $P(w|i)$ is the probability of a word w given an (unseen) image i. The probability $P(w,j)$ that word w is used to annotate image j can be estimated from an empirical distribution of annotations in the training data.

Figure 1.11 shows an unseen image i for which the five words with the highest probabilities $p(w|i)$ according to above Naïve Bayes classification are all sensible and useful.

Yavlinsky et al (2005) built models based on a similar idea for which the model for keywords appearance is derived from non-parametric density estimators with specialised kernels that utilise the Earth mover's distance. The assumption is that these kernels reflect the nature of the underlying features well. Yavlinsky built a corresponding search engine behold[16], where one could search for Flickr images using these detected terms. These algorithms all make errors as one can expect from fully automated systems. Figure 1.12 shows screenshots from an early version of behold. Clearly, not all words are predicted correctly, and the ugly examples from this figure might motivate to study methods that use external knowledge, for example, that stairs and icebergs normally do not go together.

[16] http://www.behold.cc

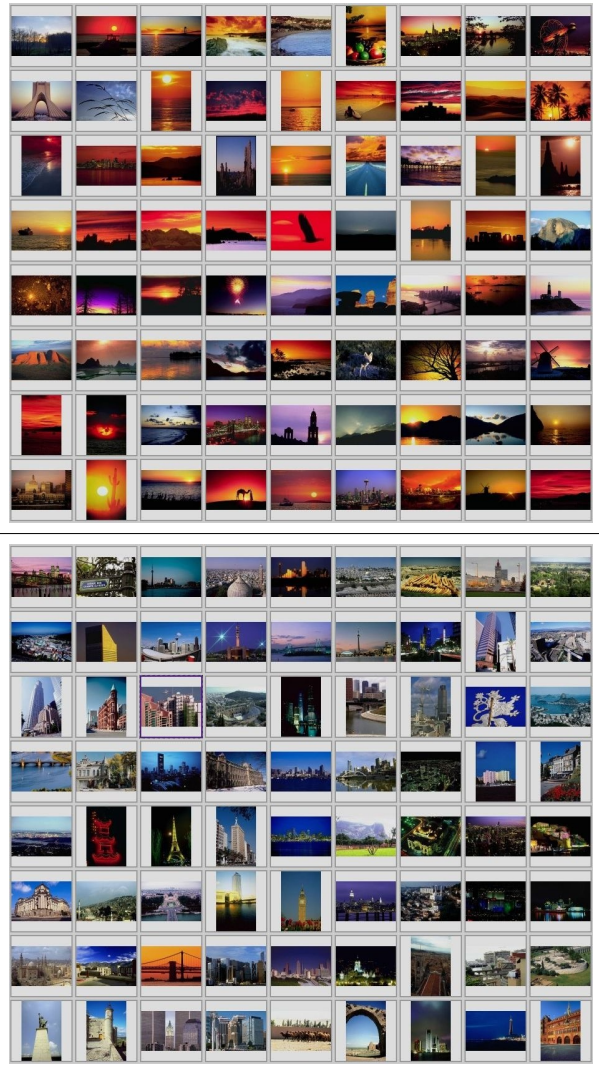


Figure 1.10: Machine learning training samples for *sunset* (top) and *city* images (bottom)

Figure 1.11: Automated annotation results in *water*, *buildings*, *city*, *sunset* and *aerial*

Today, Makadia et al's recent (2008) work on the nearest neighbour label transfer provide a baseline for automatic image annotation using global low-level features and a straightforward label transfer from the 5 nearest neighbours. This approach is likely to work very well if enough images are available in a labelled set that are very close to the unlabelled application set. This may be the case, for example, in museums where images of groups of objects are taken in a batch fashion with the same lighting and background and only some of the objects in the group have received manual labels. Liu et al (2009a) also use label transfer, albeit in a slightly different setting since they aim to segment and recognise scenes rather than assign global classification labels.

Automated annotation from pixels faces criticism not only owing to its current inability to model a large and useful vocabulary with high accuracy. Enser and Sandom (2002, 2003) argue that some of the vital information for significance and content of images *has* to come from metadata: it is virtually impossible to, eg, compute the date or location of an image from its pixels. A real-world image query such as "Stirling Moss winning Kentish 100 Trophy at Brands Hatch, 30 August 1968" cannot be answered without metadata. They argue that pixel-based algorithms will never be able to compute *significance* of images such as "first public engagement of Prince Charles as a boy" or "the first ordination of a woman as bishop". Their UK-funded arts and humanities research project "Bridging the Semantic Gap in Visual Information Retrieval" (Hare et al, 2006; Enser and Sandom, 2003) brought a new understanding about the role of the semantic gap in visual image retrieval.

Owing to these observations and also owing to their relatively large error rates, automated annotation methods seem to be more suitable in the context of browsing or in conjunction with other search methods. For example, if you want to "find shots of the front of the White House in the daytime with the fountain running"[17], then a query-by-example search in a large database may be solved quicker and better by emphasising those shots that were classified as "vegetation", "outside", "building" etc — even though the individual classification may be wrong in a significant proportion of cases.

There is a host of research that supports the bridging of the semantic gap via automated annotation. Hare and Lewis (2004) use salient interest points and the concept of scale to the selection of salient regions in an image to describe the image characteristics in that region; they then extended this work (2005) to model visual terms from a training set that can then be used to annotate unseen images. Magalhães and Rüger (2006) developed a clustering method that is more computationally efficient than the currently very effective method of non-parametric density estimation, which they later (2007) integrated into a unique multimedia indexing model for heterogeneous data. Torralba

[17] Topic 124 of TRECVid 2003, see http://www-nlpir.nist.gov/projects/tv2003

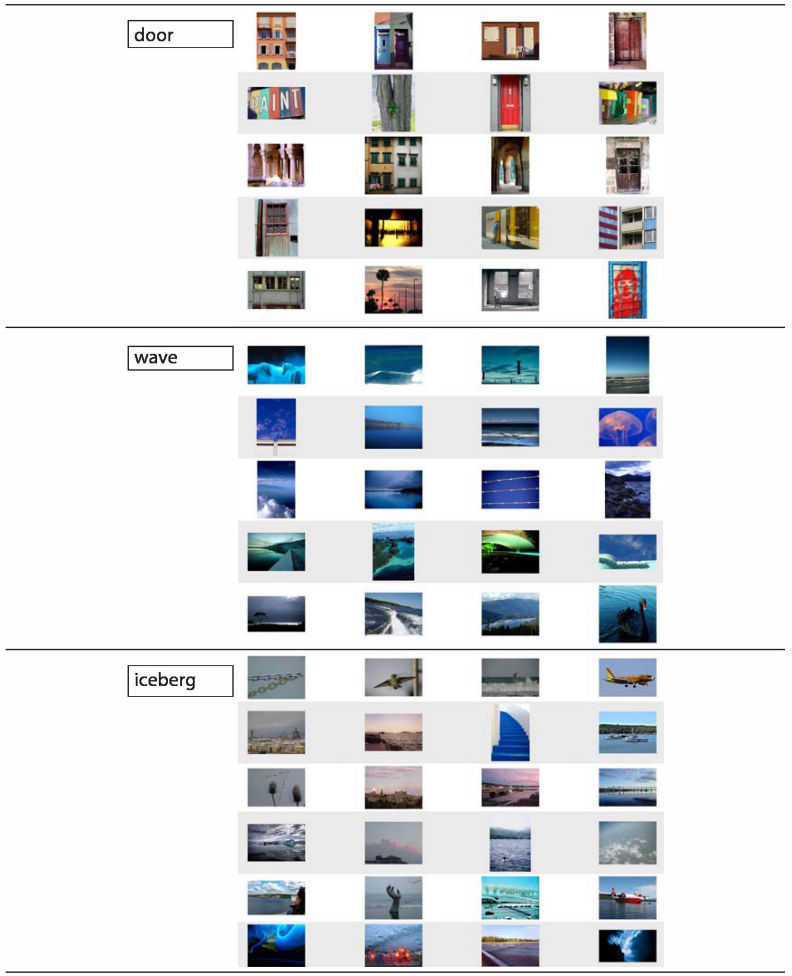


Figure 1.12: The good, the bad and the ugly: three examples for automated annotation

and Oliva (2003) obtained relatively good results with simple scene-level statistics, while others deploy more complex models: Jeon et al (2003) and Lavrenko et al (2003) studied cross-lingual information retrieval models, while Metzler and Manmatha (2004) set up inference networks that connect image segments with words. Blei and Jordan (2003) carry out probabilistic modelling with latent Dirichlet allocation, while Feng et al (2004) use Bernoulli distributions.

Machine learning methods for classification and annotation are not limited to images at all. For example, one can extract motion vectors from MPEG-encoded videos and use these to classify a video shot independently into categories such as object motion from left to right, zoom in, tilt, roll, dolly in and out, truck left and right, pedestal up and down, crane boom, swing boom etc. In contrast to the above classification tasks, the extracted motion vector features are much more closely correlated to the ensuing motion label than image features are to text labels, and the corresponding learning task should be much simpler a consequence.

The application area for classification can be rather diverse: Baillie and Jose (2004) use audio analysis of the crowd response in a football game to detect important events in the match; Cavallaro and Ebrahimi (2004) propose an interaction mechanism between the semantic and the region partitions, which allows to detect multiple simultaneous objects in videos.

On a higher level, Salway and Graham (2003) developed a method to extract information about emotions of characters in films and suggested that this information can help describe higher levels of multimedia semantics relating to narrative structures. Salway et al (2005) contributed to the analysis and description of semantic video content by investigating what actions are important in films.

Musical genre classification can be carried out on extracted audio-features that represent a performance by its statistics of pitch content, rhythmic structure and timbre texture (Tzanetakis and Cook, 2002): timbre texture features are normally computed using short-time Fourier transform and Mel-frequency cepstral coefficients that also play a vital role in speech recognition; the rhythmic structure of music can be explored using discrete wavelet transforms that have a different time resolution for different frequencies; pitch detection, especially in polyphonic music, is more intricate and requires more elaborate algorithms. For details, see the work of Tolonen and Karjalainen (2000). Tzanetakis and Cook (2002) report correct classification rates of between 40% (rock) and 75% (jazz) in their experiments with 10 different genres.

1.5 Fingerprinting

Multimedia fingerprints are a means to uniquely identify multimedia objects in a database, given a possibly different representation of it. Fingerprints are computed from the contents of the multimedia objects. They are small, allow the fast, reliable and *unique* location of the database record and, most importantly, are robust against degradation or deliberate change as long as the *human perception* is the same.

The idea is to be able to identify a specific multimedia object based solely on its content. For example, you listen to a song in a restaurant and would like to know more about it. You could record a piece of the song and use this representation to query a database of performances. Your recording will be degraded by background noise, and it will be only a part of the original performance. Its fingerprint should be sufficient to uniquely determine the entry in your large database of original recordings, say, CD tracks as published by the music industry. This does not mean that the fingerprint of the query has to be identical to the fingerprint of the original multimedia object — the former needs only contain enough evidence to identify the original fingerprint beyond

reasonable doubt. This also means that we would expect to distinguish even between different performances of the same song by the same artist at different occasions. Sinitsyn (2006) explains how audio fingerprinting algorithms can be integrated into data management middleware to perform background self-cleaning from duplicates.

Fingerprinting is just as useful in the visual world: for example, consider videos uploaded to YouTube, a video sharing site. The uploaded video may already exist in YouTube, but have been transcoded for a different bandwidth into a different format, edited to conform to a different aspect ratio, may have logos or advertisement inserted, but its fingerprint should still identify it as a copy of another one already in the database.

As we want to identify objects based on perception, simple hashes such as Message Digest algorithm 5 (MD5) or the Cyclic Redundancy Check (CRC) of the multimedia contents are not sufficient: these would already change when a single bit of the file is changed, let alone when the whole representation and encoding of the multimedia object radically changed.

1.5.1 Audio Fingerprinting

For audio, many fingerprinting algorithms are based on the spectrogram of the song. A spectrogram is a threeway graph telling us which frequencies contain how much energy at which time of the audio piece. Salient points of a spectrogram are time-frequency points (τ, f) that contain a relatively high amount of energy. In order to compute the energy as a function of the frequency at a given point τ in time, the original pressure-wave sound signal $\tau \mapsto s(\tau)$ is subjected to a short window around τ in time. Rather than using rectangular windows that would harshly cut off the signal and artificially introduce high frequencies near the cut-off points, one uses smooth infinite windows such as a Gaussian or, more popularly, a finite Hann window $t \mapsto w(t) = (1 - \cos(2\pi(t-\tau)/T))/2$ with width T on the interval $[\tau - T/2, \tau + T/2]$. This short-term signal is then subjected to a Fourier transform, the magnitude of which is known as the energy distribution in frequency space:

$$\text{spectrogram}(f, \tau) = \left| \int_{-\infty}^{\infty} s(t) w(t-\tau) e^{\text{æ} f t} dt \right|^2$$

In practice, the spectrogram is computed as the squared modulus of the discrete short-term Fourier transform using discrete time and frequency variables and sums instead of integrals.

Other transformations are in use, such as the Modulated complex lapped transform, the Walsh-Hadamard transform or variants of the Fourier transform, from all of which robust features are extracted (Cano et al, 2005). In the following few paragraphs, we will look at two detailed mechanisms to find known pieces of music in a database from spectrograms: Shazam's constellation maps (Wang, 2003) and Philips Research's fingerprint blocks (Haitsma and Kalker, 2003). Shazam[18] actually provides a service that allows you to capture snippets of music via your mobile phone and returns one of the 8 million tracks in their database[19].

Figure 1.13 is a screenshot of audacity, a free digital audio editor, displaying a spectrogram of a 5 second light sabre sound from Star Wars. The bright points in the colour spectrogram correspond to the peaks in the energy. The important point to remember is that the location of these peaks would remain invariant under varying compression, audio encoding, background noise etc. The light sabre sound has a characteristic near continuous band of high energy in the low frequencies in addition to columns of energy spread over the whole frequency spectrum. Once quantised with a

[18] http://www.shazam.com

[19] http://www.shazam.com/music/web/newsdetail.html?nid=NEWS103 (last accessed Apr 2009)

suitable algorithm, we get distinct salient points in the spectrum, see Figure 1.14. In music pieces, they look a bit like star constellations, which is why Wang (2003) has called the scatter plot of these points *constellation maps*.

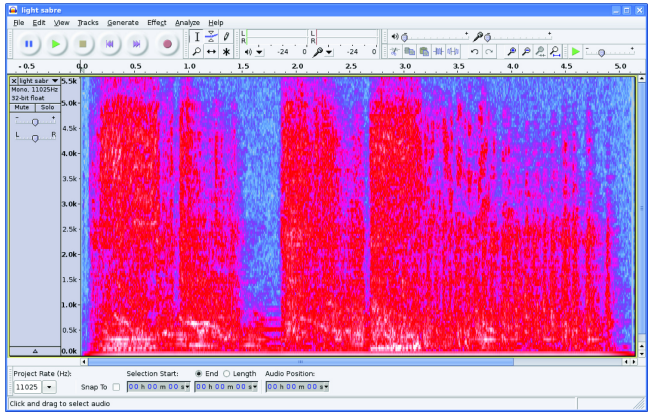


Figure 1.13: Audacity screenshot of the spectrogram of a light sabre sound

The quantised coordinate list for the constellation maps recording peak energy is sparse and discrete. The problem of detecting the song in the database is reduced to a registration problem of the constellation maps: Imagine the constellation maps of all database songs on a long strip of paper and the query's constellation map on a small piece of transparency foil. You find the matching music piece by gliding the transparency over the paper strip and noting the best match.

Wang (2003) used certain *pairs* of constellation points to search for matches: for each constellation point (also called anchor point) he considers a fixed rectangular target zone to the right of the anchor point and uses all pairs that consist of the anchor point and a point in target zone. Each of these constellation pairs is encoded as its pair of frequencies and the time difference making the representation time invariant. With a suitable discretisation of the time and frequency axis, say 10 bits each, these three values can be packed into a 32-bit hash value that associates the song id and time offset of the anchor point within the song to it, see Figure 1.14. The use of constellation pairs instead of points increases the specificity of a single match of a constellation pair (as opposed to a point) by a factor of $2^{20} \approx 1,000,000$ (30 bits vs 10 bits). Let us assume that the dimension of the target zone is constructed in a way that the number of constellation points in it is limited to, on average, n points, say $n = 10$. Then there are n times more pairs to store than there are points and there are n times more query pairs to match than there are anchor points. This reduces the speed-up factor of one match to approximately $1,000,000/n^2 = 10,000$, assuming $n = 10$.

The actual matching process is straightforward: the audio query, which is a captured part of a song, is processed and hash values of certain constellation pairs (as described above) are extracted. Each of the hashes is looked up in the database of constellation pairs hashes of all songs. Each match results in three pieces of information: the time t_1^q of the anchor point in the query hash, the time t_1^d of the matching anchor point in the database and the corresponding song id. For a true match of the music sample with a particular song, the matched pairs need to align consistently in

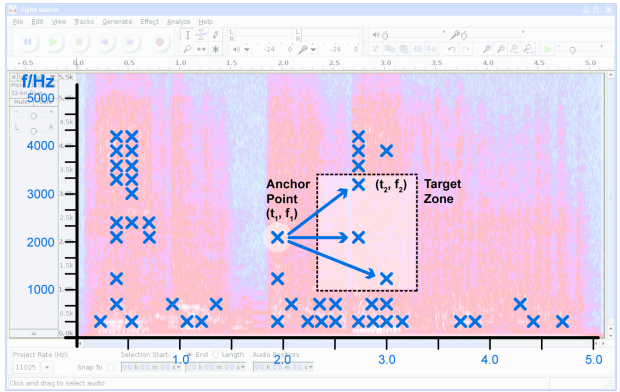


Figure 1.14: Salient points in the spectrum of a light sabre sound. Certain pairs of spectral points are used for retrieval and encoded as as the hash function $(f_1, f_2, t_2 - t_1) \mapsto (t_1, id)$. The visualised salient points here are not to scale: see Exercise 1.6.10.

time. As the query only captures a part of the song, which can start anywhere in the song, the times of the anchor points of query and matching song will not be the same, but their difference $t_1^d - t_1^q$ is expected to be a constant for true matches, and this constant should be the starting time of the captured query in the full song. Separating true matches from spurious ones can thus be done with the following method: for each song, in which a match of a constellation pair occurred, keep a small histogram of the observed differences $t_1^d - t_1^q$. One of these histograms is expected to exhibit a substantial peak in a particular time difference, and the corresponding song will be the right one from the database.

Haitsma and Kalker (2003) from Philips Research suggest to use the spectrogram for their fingerprints, too. They extract a 32-bit sub-fingerprint every 11.6 ms (the granularity of the time axis) and collect 256 adjacent sub-fingerprints into a block covering around 3 s of music. They divide the frequency scale between 300 Hz and 2,000 Hz into 33 logarithmically spaced frequency bands, so that the distance between two neighbouring frequency bands becomes roughly 1/12 of an octave, or a semitone. Let $E(m, n)$ denote the energy of the mth frequency band at the nth time frame of 11.6 ms. Then the mth bit of the nth frame is set to the sign of the following neighbouring energy differences ($0 \le m \le 31$ and $0 \le n \le 255$):

$$[E(m, n) - E(m + 1, n)] - [E(m, n + 1) - E(m + 1, n + 1)] \tag{1.1}$$

They argue that the sign of this energy difference is relatively robust under different encoding schemes. A partial fingerprint block for the light sabre sound is depicted in Fig 1.15. In theory this is now a similar registration and matching problem as in Shazam's representation, but it is approached differently here. The reason for this is presumably owing to the fact that perceptually same music pieces can exhibit a bit error rate of 10-30% between matching fingerprint blocks. Indeed, Haitsma and Kalker (2003) set a threshold of 35%: if the bit error rate between two fingerprint blocks falls below 35% then they declare these two blocks as coming from the same song.

Even though a high bit error rate of $b = 0.3$ causes the probability $p(4)$ that no more than 4 bits

were flipped to drop under 2%, it is the case that when you look at 256 sub-fingerprints, at least one of them will have no more than 4 bit errors with more than 99% probability (see Exercise 1.6.8).

The basic search idea is to go through the 256 sub-fingerprints in a query block and try to match each sub-fingerprint against a database sub-fingerprint. If there is a match of two sub-fingerprints, then the corresponding blocks are compared; if, furthermore, the bit error rate is below the threshold, then a match is declared on the corresponding song. If no matching song was found when checking all 256 sub-fingerprints in the query, then it is assumed that none of the 256 sub-fingerprints of the query has survived the transformation from the original song unscathed. The next best thing is to assume that at least one of the query sub-fingerprints has only one bit flipped. This gives rise to 32 different sub-fingerprints with one changed bit. If the next round of checks with $256 \cdot 32$ modified sub-fingerprints from the query does not identify a matching song then 2 bit-errors are assumed to have happened in the query. There would be $\binom{32}{2} = 32 \cdot 31/2 = 496$ different combinations that can be produced from each sub-fingerprint. As this is getting more and more computationally expensive, Haitsma and Kalker (2003) suggest a heuristics: they assume that bits in the query sub-fingerprints are more susceptible to bit changes if they arise from energy differences (1.1) closer to zero. They would only create modified query sub-fingerprints for the most susceptible bits. This makes it feasible to explore variations where more than 2 bits are changed. If after all reasonable attempts still no matching block could be identified, it is assumed that the query song was either too distorted or is not in the database.

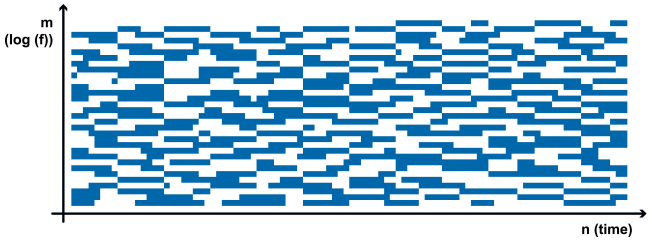


Figure 1.15: Partial fingerprint block of initial light sabre sound

Audio fingerprints of music tracks are expected to distinguish even between different performances of the same song by the same artist at different occasions.

Interesting applications include services that allow broadcast monitoring companies to identify what was played, so that royalties are fairly distributed or programmes and advertisements verified. Other applications uncover copyright violation or, for example, provide a service that allows you to locate the metadata such as title, artist and date of performance from snippets recorded on a (noisy) mobile phone.

1.5.2 Image Fingerprinting

The requirements for image fingerprints are the same as for audio fingerprints: they should be small, and allow the fast, reliable and *unique* location of the database record under degradation or small deliberate change. The main difference between Images and Audio is that images are static two-dimensional colour distributions, while sound is a one-dimensional air pressure function of time. We will look at different ways to record image features, so that similar images can be recognised

"beyond a reasonable doubt". The underlying basic idea is to create a number L of *independent* representations, each of which maps near duplicates to the same quantised value with a reasonable, but not necessarily very high probability; this quantised value corresponds to the constellation pair hash for audio fingerprinting. The next step is to require a match of a certain number m out of the L independent representations between two images for them to be declared near identical. In the following, we will look at one method, locality sensitive hashing (LSH), for near-duplicate detection for dense features and one, min hash, for sparse features.[20]

Locality Sensitive Hashing

LSH (Datar et al, 2004) consists of independent random projections of the original feature space to integers that are combined into a hash value. The basic procedure maps a feature vector $v \in \mathbb{R}^d$ to an integer

$$h^i(v) = \left\lfloor \frac{a^i v + b^i}{w} \right\rfloor, \tag{1.2}$$

where $a^i \in \mathbb{R}^d$ is a random vector with independent normal distributed components, $w \in \mathbb{R}^+$ is a constant specifying the granularity of the results and $b^i \in [0, w)$ is a random uniformly distributed number. A k-tuple of these integers defines a composite hash-value

$$h(v) = (h^1(v), h(^2(v), \ldots, h^k(v)).$$

The preimage in feature space of a particular hash-value is an area that is bordered by pairs of hyperplanes that are perpendicular to one of the random vectors a^i and that have a distance of w. Normally we have $k < d$, which means that the preimage is not bounded. Figure 1.16 visualises how such a hash function works for $k = 2$. Consequently, we have two random vectors $a^1, a^2 \in \mathbb{R}^d$ that determine the orientation of the $d-1$-dimensional hyperplanes in feature space where the value of the hash function changes. In the figure, each random vector a^i and the corresponding perpendicular hyperplanes are shown in the same colour. The random offsets $b^1, b^2 \in [0, w)$ effectively shift the point of origin in feature space by less than the width w of a hash bin, and they are not visualised here. From the illustration it is clear that nearby points in feature space are likely to end up with the same hash value, though they could have neighbouring hash values owing to boundary effects. It is also apparent that points far away may share the same hash bin. Hence, a series of L independent hash functions is computed for the query with the condition that at least m of these L composite hash values coincide with the corresponding one of a multimedia object before the latter is assumed to be close to the query beyond reasonable doubt.

In practice, only a finite number of bits will be used to store a particular hash integer computed from (1.2). In the example of Figure 1.16, we have 3 bit per axis yielding 6 finite bins and two infinite bins per axis. This setup is only reasonable if we can assume that the points in feature are concentrated on a finite domain (ie, the features are essentially normalised). The composite hash value $h(v)$ only has 6 bit in our example. Note that a reasonable number of bits in a large repository would be around 20 bit, so that a single hash lookup has the potential to reduce the number of points in feature space by a factor of roughly one million (ie, 2^{20}). Based on this the approximate speedup factor should be $L/1,000,000$, as executing a single query means to compute

[20] Dense features are ones that are likely to have non-zero components in a fair number of them for a typical image. This would be the case for colour histograms of images. In contrast to this, sparse feature vectors are those for which the number of non-zero components is small, eg, a 'bag of word' representation of text documents or an artificial visual vocabulary for images.

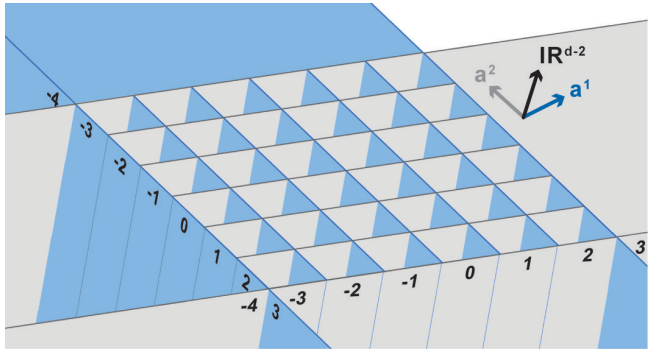


Figure 1.16: Preimage of an LSH function $v \mapsto h(v) = (h^1(v), h^2(v))$ leaving $d-2$ dimensions unbounded in the d-dimensional feature space

L composite hash values and looking up the contents of corresponding, possibly unbounded, hash bins. Those database elements that appear in at least m of the L bins are candidates for nearest neighbours. If precision is important, then the true distance to the query in feature space can be checked for each candidate, but in some applications one might even consider not looking at the feature values of the database elements at all during query time.

Note that w, k and L have to be chosen at index time (unlike m that can be chosen at query time), and their choice should reflect the properties of the repository data in feature space: the product mk ought to exceed d in order for LSH to have a chance of bounding a finite area around the query, while w and k should be chosen so that the typical bin has the desired occupancy.

It looks as if each query can be done in constant time, but — like all hashing techniques — LSH has a query time that is proportional to the number of database entries as the average bucket occupancy increases with the database size. A careful redesign of the hash parameters k and w, however, can reduce the average bucket occupancy once the response is deemed too slow. Although each composite hash value may use only a small disk space, the fact that L hash tables need to be stored makes LSH space-hungry. Ideally, the hash tables are kept in main memory, which limits the number of multimedia entries that can be managed by one single server. In order to alleviate that effectively, kL individual hash values are kept for each feature vector of a multimedia object, one can reuse hash functions: rather than deploying kL random vectors a^i and offsets b^i, one can generate a smaller number n ($2k < n < kL$) of these and randomly chose k out of n vectors and offsets for each of the L composite hash functions. The effect of this reuse is that the composite hash values are not completely independent of each other, but this ultimately saves space.

The significance of using random vectors a^i with components drawn independently from a normal (Gaussian) distribution lies in the fact that their random projections approximately preserve Euclidean distances. It has been argued that Manhattan distances often work better for feature comparisons in content-based image retrieval (Howarth and Rüger, 2005a); random vectors whose components are independently drawn from a particular distribution of family of Cauchy distributions $x \mapsto \frac{c}{\pi}\left(c^2 + x^2\right)^{-1}$ with parameter $c > 0$, approximate Manhattan distances.

Min Hash

Some of the features developed for describing images are sparse, for example, quantised SIFT features from salient image regions. These quantised features share some of the properties of words in language and, hence, are called *visual words*. Let V be the vocabulary of visual words and $A_i \subset V$ be a set of words that describes a particular image i. The similarity between two images i and j can then be expressed as the ratio of the size of the intersection and the union of the representing sets,

$$\mathrm{sim}(A_i, A_j) = \frac{|A_i \cap A_j|}{|A_i \cup A_j|}, \tag{1.3}$$

which is a number between 0 meaning no overlap and 1 meaning identical representing sets. Broder (1997) called this similarity the *resemblance* of two documents and published a method of estimating $\mathrm{sim}(A_i, A_j)$ using random permutations. I will exemplify Broder's algorithm in a simplified form using the following four small text documents:

1. Humpty Dumpty sat on a wall,
2. Humpty Dumpty had a great fall.
3. All the King's horses, And all the King's men
4. Couldn't put Humpty together again!

Removing stop words and putting the words into sets leaves the following four sets:
$A_1 = \{\text{humpty, dumpty, sat, wall}\}$
$A_2 = \{\text{humpty, dumpty, great, fall}\}$
$A_3 = \{\text{all, king, horse, men}\}$
$A_4 = \{\text{put, humpty, together, again}\}$
Equivalently, a document-word matrix of zeros and ones records membership of words in documents. This results in long document vectors, the columns of this matrix:

	A_1	A_2	A_3	A_4
humpty	1	1	0	1
dumpty	1	1	0	0
sat	1	0	0	0
wall	1	0	0	0
great	0	1	0	0
fall	0	1	0	0
all	0	0	1	0
king	0	0	1	0
horse	0	0	1	0
men	0	0	1	0
put	0	0	0	1
together	0	0	0	1
again	0	0	0	1

Using the full document-word matrix for a large collection to compute the inter-document similarities (1.3) is very slow. Owing to the sparsity of the document vectors, the naïve approach of randomly sampling rows of this matrix would result in many uninformative rows for particular documents, ie samples that contain only zeros. Note that for any pair of documents there are only four different rows in the corresponding matrix: (0,0), (0,1), (1,0) and (1,1). It turns out, indeed,

that above definition (1.3) of $\text{sim}(A_i, A_j)$ does not depend on how many words occur in neither document: let c_{xy} count the number of (x, y) rows. The key observation is then that

$$\text{sim}(A_i, A_j) = \frac{c_{11}}{c_{11} + c_{10} + c_{01}}, \tag{1.4}$$

ie, $\text{sim}(A_i, A_j)$ is independent of c_{00}. The trick of the min hash is to use a random permutation of the vocabulary, and then, for each document, *only* record its first word under this permutation ignoring all the words that do not occur in the document. This word is called a *min hash*. Consider the following 4 permutations:

π_1 = (dumpty, men, again, put, great, humpty, wall, horse, king, sat, fall, together, all)

π_2 = (fall, put, all, again, dumpty, sat, men, great, wall, king, horse, humpty, together)

π_3 = (horse, dumpty, wall, humpty, great, again, sat, all, men, together, put, king, fall)

π_4 = (king, humpty, men, together, great, fall, horse, all, dumpty, wall, sat, again, put)

They give rise to the following min hashes:

	A_1	A_2	A_3	A_4
π_1	dumpty	dumpty	men	again
π_2	dumpty	fall	all	put
π_3	dumpty	dumpty	horse	humpty
π_4	humpty	humpty	king	humpty

The surprising fact is that the probability that the respective min hashes of two documents coincide is equal to $\text{sim}(A_i, A_j)$; this means that $\text{sim}(A_i, A_j)$ can be estimated with a simple counting exercise, namely how often their min hashes coincide under different permutations. In order to understand why this is is the case, let w be the min hash of $A_i \cup A_j$ under a random permutation π, and without restriction of generality, assume $w \in A_i$ implying that w is the min hash of A_i under π. Then w coincides with the min hash of A_j under π if and only if $w \in A_i \cap A_j$. As π is a random permutation, w is chosen from $A_i \cup A_j$ with uniform probability. Consequently, the probability of min hash of A_i coinciding with A_j is exactly the proportion $|A_i \cap A_j|/|A_i \cup A_j|$, ie, $\text{sim}(A_i, A_j)$.

Hence, the proportion of cases where the min hashes of A_i and A_j coincide while π varies is a good approximation of $\text{sim}(A_i, A_j)$ provided the permutations are chosen independently and uniformly. In above example, $\text{sim}(A_1, A_2)$ is estimated to be $3/4$ (the true value being $1/2$), while $\text{sim}(A_1, A_4)$ is correctly estimated to be $1/4$.

Broder (1997) uses the concept of shingles, ie, word sequences in documents, making the representation even more sparse. This is not necessary for image matching as there is no natural order of the visual words in an image. The other generalisation in the original paper is that each min hash records the first s words of a document in the permutation vector.

For the purposes of near duplicate detection k independent min hashes of a document are subsumed into what is called a *sketch*. The probability that two documents have an identical sketch is $\text{sim}(A_i, A_j)^k$. Requiring coinciding sketches reduces the probability of a false positive considerably and will only look at candidates that are highly likely to be very similar. As with LSH, L independent sketches, which act as hash values, are stored for every document. At query time, the sketches of a query document are used to retrieve L sets of documents with the same sketch and the requirement for candidates of near duplicates is to coincide at least m times out of the L sketches.

1.6 Exercises

1.6.1 Memex

One of the early true visionaries of digital libraries was Vannevar Bush (1890–1974): he predicted the internet and modern digital libraries long ahead of their time — and did not live to experience them! In July 1945 he published an essay "As we may think" in The Atlantic Monthly, in which he described the idea of a memory extender, short *memex*. This fictitious machine (see Figure 1.17 for its design) would be integrated into a desk that contains a glass plate with a camera to take pictures of pages, microfilm storage, projection systems for two screens, and a keyboard with levers. Its purpose was to provide scientists with the capability to exchange information and to have access to all recorded information in a private library that contained all your books, correspondence and own work. It would function as a rapid information retrieval system and extend the power of human memory.

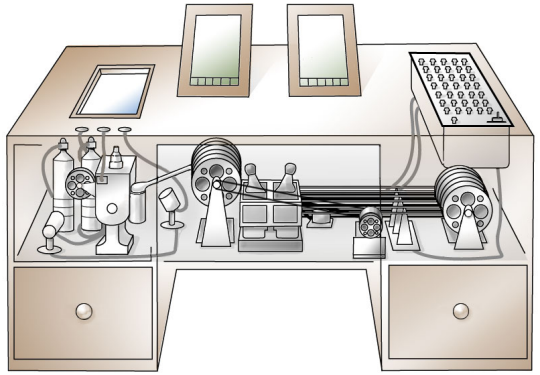


Figure 1.17: Memex design

More importantly, his design included the concept of associating resources and adding comments to them. The most remarkable fact of the memex design is that its (analogue) links are very similar to the links on web-pages. For that reason, Bush is now seen as the grandfather of the world-wide web. It took nearly two decades after the invention of the internet for the element of memex that allows adding your own work to be widely and easily used: wikipedia[21], an online encyclopedia, was born in 2001, where everyone can link to resources, add comments and corrections and write own articles.

• Discuss whether the microfilms in the memex constitute monomedia, multimedia, hypertext or hypermedia in their strict literal sense. What about wikipedia articles? Or this book used online? What about the same as a printed book? Which of the above are multimedia as defined in the tutorial?

[21] http://wikipedia.org

1.6.2 Loops and Interaction

Information retrieval is more than just search: it is browsing, searching, selecting, assessing and evaluating, ie, ultimately accessing information. Figure 1.18 gives a breakdown of different stages of this process from information need to information use.

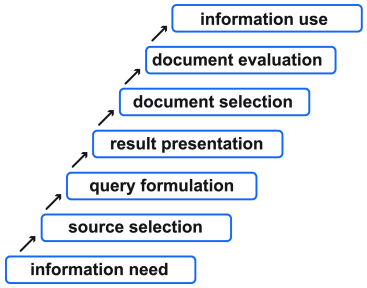


Figure 1.18: Stages of Information Retrieval

• Illustrate these steps using examples of an information retrieval quest for (a) a book, say an early "Asterix" graphic novel, in your local library; (b) for a government tax form on the web that allows you to declare the income-equivalent-benefits of your employer providing free tea; (c) for a figure in an electronic image library that demonstrates how a prism disperses light into rainbow colors.
• Identify loops in this process, for example, after document evaluation you may want to go back to the document selection step. Add all loops to Figure 1.18 that you think useful for the information retrieval process.

1.6.3 Automated vs Manual

Not all manual intervention has been abandoned in web search engines. For example, the Yahoo directory is an edited classification scheme of submitted web sites that are put into a browsable directory structure akin to library classification schemes.
• Find and discuss other examples of manual intervention or manually delivered services that are associated to web search engines.

1.6.4 Compound Text Queries

Text search engines normally go through the following indexing steps: collect the documents to be indexed; extract a list of terms from each document by ignoring punctuation, identifying word-boundaries and normalising terms with respect to accents, diacritics spelling variants and folding to lower case; and maintain one list of documents for every term sorted by decreasing relevance. Such a list is called postings list for that particular term. The set of postings lists for the whole vocabulary is called *inverted-file index*, simply *index* or, misleadingly, *inverted index*.
• Assuming you wanted a two-term query *white house* to return all documents that contain the terms *white* and *house*, what would be a fast way of processing the intersection between the two postings lists for *white* and *house*? How would you efficiently compute the results for the query *white* −*house* targeting documents that contain the term *white* but not the term *house*?

• How could you change the index of a collection in order to be able to search for phrases such as *white house* requiring the two terms to appear successively in the document?
• Implement a simple text search engine that indexes e-mail folders using above ideas: relevance is defined as recency of the e-mail (the older, the less relevant).

1.6.5 Search Types

Revisit the matrix of search engine types in Figure 1.1. Give five more search and retrieval scenarios for elements of this query-retrieval matrix. Give two more examples for search and retrieval scenarios that, like Entry C, span multiple query modes or multiple document types.

1.6.6 Search Types Continued

Coming back to Exercise 1.6.5 and Figure 1.1, what would be the most appropriate search technology (piggy-back text search, feature classification, content-based, fingerprint) for each scenario?

1.6.7 Intensity Histograms

Revisit Figure 1.7: why is it a good idea to record the *proportion* rather than the absolute value of the number of pixels that fall into the intensity ranges? Explain why these histograms are normalised with respect to the L_1 norm.

1.6.8 Fingerprint Block Probabilities

Given a fingerprint block of 256 sub-fingerprints each of which has 32 bits assume the whole block is subjected to an independent identically distributed bit errors at the rate of b. Show that the probability $p(k, b)$ of having no more than k bit errors in one sub-fingerprint is

$$p(k,b) = \sum_{i=0}^{k} \binom{32}{i} (1-b)^{32-i} b^i.$$

Show that the probability that among 256 sub-fingerprints at least one survives with no more than k bit errors is given by

$$1 - (1 - p(k,b))^{256}.$$

Verify, using above formulas, the following claim on page 21: Even though a high bit error rate of $b = 0.3$ causes the probability $p(4)$ that no more than 4 bits were flipped to drop under 2%, it is the case that when you look at 256 sub-fingerprints, at least one of them will have no more than 4 bit errors with more than 99% probability.

1.6.9 Fingerprint Block False Positives

Assuming that the fingerprint block extraction process yields random, independent and identically distributed bits, what is the probability that a randomly modified fingerprint block matches a *different* random block in the database that consists of, say, 10^{11} overlapping fingerprint blocks (4 million songs with around 5 minutes each)? The bit error rate for the random modification is assumed to be 35%.

1.6.10 Shazam's Constellation Pairs

Assume that the typical survival probability of each 30-bit constellation pair after deformations that we still want to recognise is p, and that this process is independent per pair. Which encoding density, ie, the number of constellation pairs per second, would you need on average so that a typical captured query of 10 seconds exhibits at least 10 matches in the right song with a probability of at least 99.99%? Under these assumptions, further assuming that the constellation pair extraction looks like a random independent and identically distributed number, what is the false positive rate for a database of 4 million songs each of which is 5 minutes long on average?

1.6.11 One Pass Algorithm for Min Hash

Rather than actually permuting the rows of the document-word matrix, one would create kL permutation functions $\pi_l: \{1, \ldots, n\} \to \{1, \ldots, n\}, l \in \{1, \ldots, kL\}$, where n is the size of the visual vocabulary (ie, the number or words), k is the min hash sketch size, and L is the number of sketches that are deployed.

a) Show that the following algorithm computes min hash values: let m be the number of images in the repository. Initialise all elements of a $kL \times m$ matrix h to ∞. The document-word matrix is scanned once in an arbitrary order: for each nonzero (j, i) element (meaning that word j appears in image i) and for each l with $1 \leq l \leq kL$ set $h[l, i]$ to $\pi_l(j)$ if $\pi_l(j) < h[l, i]$.

b) Assume π is an array of n integers that is initialised so that $\pi[l] = l$. Show that by assigning a random number to each array element and then sorting the array elements by the assigned random number you create a random permutation with uniform probability.

c) Initialise π as above. Show that by scanning π from $l = n$ down to $l = 1$ and, at each step, computing a random index i drawn uniformly and independently between 1 and l and then swapping the contents of $\pi[i]$ with $\pi[l]$ you create a random permutation with uniform probability.

Chapter 2

Content-based Retrieval in Depth

As features and distances are the main two ingredients for content-based retrieval, we will devote a whole section for each (2.2 and 2.3, respectively) and on tricks of the trade on how to standardise features and distances (2.4). Before delving into these, let us look at the global architecture (2.1) of how to use features and distances in principle. It will become apparent that the crux of the indexing problem is to compute nearest neighbours efficiently (2.5). With all this in hand, we can then look into the question of how to fuse evidence from the similarity with respect to different features that multimedia objects exhibit and how to merge search results from multiple query examples (2.6).

2.1 Content-based Retrieval Architecture

The query-by-example paradigm extracts features from the query, which can be anything from a still image, a video clip, humming etc, and compares these with corresponding features in the database. The important bit is that both the query example and the database multimedia objects have undergone the same feature extraction mechanism. Figure 2.1 shows a typical architecture of such a system. The database indexing and similarity ranking tasks in Figure 2.1 identify the nearest neighbours in feature space with respect to a chosen distance function. It is sufficient to sort the distances of the query object to all database objects and only present as results the nearest neighbours, ie, the ones with the smallest distances to the objects in feature space.

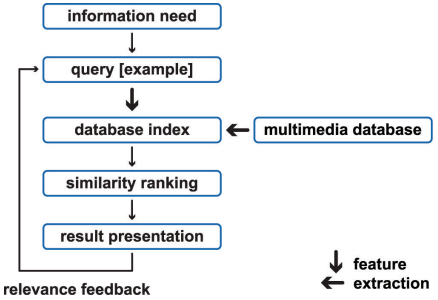


Figure 2.1: Content-based multimedia retrieval: generic architecture

The complexity of computing all distances and sorting them is $O(N \log(N))$ in time with N being the number of database objects. This can be brought down to $O(N)$ if one is only interested in the nearest, say, 100 neighbors. As such, this process is scalable in the sense that an increase of the database size requires a corresponding increase in the resources[1]. This naïve approach is very resource intense, though, as a single query requires a system to touch all feature vectors. This is neither practical, nor desirable for large databases. Section 2.5 looks at the indexing problem to compute nearest neighbours efficiently.

Both Figure 1.8 and the architecture sketch in Figure 2.1 suggest that there is only one monolithic feature space. However, it makes more sense to compute different features independently and in a modular fashion. Rather than one feature vector for a multimedia object m, one would have a number r of low-level features $f_1(m)$, $f_2(m)$, ..., $f_r(m)$, each of which would typically be a vector or numbers representing aspects like colour distribution, texture usage, shape encodings, musical timbre, pitch envelopes etc. Of course, one can always concatenate these individual feature vectors to obtain a large monolithic one, but this is inelegant as different applications might want to focus on different features of the multimedia objects, and different individual feature vectors might call for different distance functions. Section 2.6 will look at ways to "fuse" query results from multiple feature spaces and at how to fuse comparisons over multiple query examples.

The architecture presented here is a typical, albeit basic, one; there are many variations and some radically different approaches that have been published in the past. A whole research field has gathered around the area of video and image retrieval as exemplified by the ACM Multimedia conference (ACM MM), the Conference on Image and Video Retrieval (CIVR), and Multimedia Information Retrieval (MIR), which used to be a workshop at ACM MM, has later developed into an ACM conference and is going to be merged with CIVR to form the ACM International Conference on Multimedia Retrieval (ICMR) from 2011. The TREC video evaluation workshop TRECVid has supported video retrieval research through independent, metric-based evaluation (see Subsection 3.2), while ImageCLEF has had this role for image retrieval research (see Chapter 3). There is another research field around music retrieval, as evidenced by the annual International Society for Music Information Retrieval Conference (ISMIR[2]), which has grown out from a symposium to a notable conference in 2003.

2.2 Features

One common way of indexing multimedia is by creating summary statistics, which represent colour usage, texture composition, shape and structure, localisation, motion and audio properties. In this section, I discuss some of the widely used features and methods in more depth. Deselaers et al (2008) compare a large number of different image features for content-based image retrieval and give an overview of a large variety of image features. Features are not only relevant for retrieval tasks: Little and Rüger (2009) demonstrate how important the choice of the right features is for the automated image annotation task.

2.2.1 Colour Histograms

Colour is a phenomenon of human perception. From a purely physical point of view, light emitted from surfaces follows a distribution of frequencies as seen in Figure 2.2. Each pure spectral frequency

[1] the technical definition of scalable is that the problem is $O(N)$ or better in time and memory
[2] http://www.ismir.net

corresponds to a hue, all of which create the rainbow spectrum. The human eye has three different colour receptors that react to three different overlapping ranges of frequencies; their sensitivity peaks fall into the red, green and blue areas of the rainbow spectrum. Hence, human perception of colour is three-dimensional, and modelling colour as a mixture of red, green and blue is common. Virtually all colour spaces are three-dimensional (except for ones that utilise a fourth component for black), and so are colour histograms.

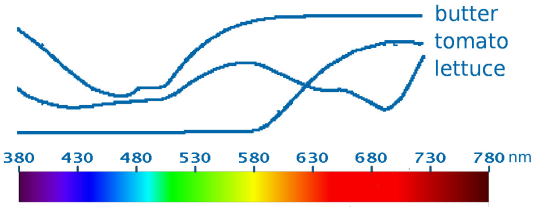


Figure 2.2: Spectral power of light emitted by different substances (imagined — not measured)

An example of a 3-dimensional colour histogram is depicted in Figure 2.3, which shows a crude summary of the colour usage in the original image. Here each of the red, green and blue colour axes in the so-called RGB space is subdivided into intervals yielding $4 \times 4 \times 4 = 64$ 3d colour bins; the proportion of pixels that are in each bin is represented by the size of a circle, which is positioned at the centre of a bin and coloured in correspondingly.

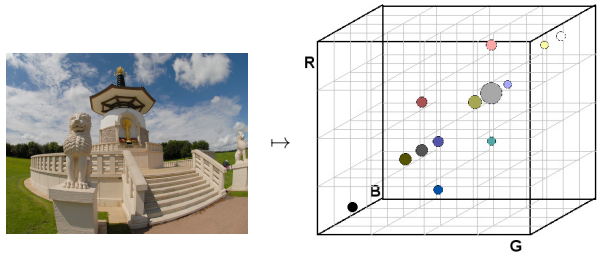


Figure 2.3: 3d colour histogram

In general, the algorithm for computing histograms involves four steps as follows: (a) the underlying space is partitioned into cells — these can be 3d cells as in Figure 2.3 or 1d intensity intervals as in Figure 1.7; (b) each cell is associated with a unique integer, known as the histogram bin number or as the index of the histogram; (c) the number of occurrences in each cell is recorded using an integer counter each, eg, by sweeping over all pixels in the image, computing into which cell the pixel falls, and incrementing the corresponding bin counter; (d) the histogram is then normalised and, optionally, quantised: normalisation just requires to divide each bin counter h_i by the number n of image pixels ($n = wh$, where w is the width and h the height of the image). Quantisation is often required for space efficiency and typically done by assigning a k-bit integer

to each histogram bin i through

$$i \mapsto \left\lfloor 2^k \frac{h_i}{n+1} \right\rfloor .$$

We divide by $n+1$ as opposed to n ensuring that $h_i/(n+1)$ can never reach 1 (in which case i were to be mapped to 2^k, which is just outside the range of a k-bit integer).

2.2.2 Statistical Moments

Statistical moments are other ways to summarise distributions. If you wanted to express a quality of an object, say the intensities $p(i,j)$ of pixels at position (i,j), through one number alone you would most likely chose its average

$$\mu = \frac{1}{wh} \sum_{i=1}^{w} \sum_{j=1}^{h} p(i,j),$$

where w is the width and h the height of the image. The values

$$\overline{p}_k = \frac{1}{wh} \sum_{i=1}^{w} \sum_{j=1}^{h} (p(i,j) - \mu)^k \tag{2.1}$$

with $k > 1$ are known as the *central moments* of the quantity p, Indeed, the knowledge of all central moments and of μ is sufficient to reconstruct the distribution of p. $\overline{p}_2$ is also known as *variance*, while $\overline{p}_3$ and $\overline{p}_4$, respectively, are used to define (but not equal to) *skewness* and *kurtosis*. Skewness is a measure of the asymmetry of the probability distribution of the variable p: a negative value for a bell-shaped unimodal (ie, one peak) distribution indicates a long left tail where the mean is farther out in the left long tail than is the median; symmetric distribution bring about zero skewness, and the mean coincides with the median; finally, a positive value indicates a long right tail.[3] Kurtosis is a measure of how fat the tails of the distribution are: a high value means that much of the variance is owed to infrequent extreme deviations (a thin long tail), as opposed to frequent modestly-sized deviations.

In practical terms, the $\overline{p}_k$ have a highly different typical range owing to the "to the power of k" element in Equation 2.1. Hence, when using moments as parts of feature vectors, one normally deploys the k-th root in order to make the values comparable in size:

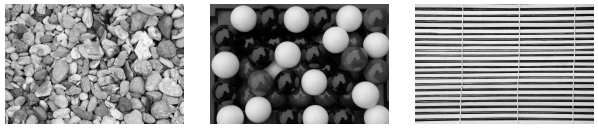
$$m_k = \text{sign}(\overline{p}_k) \sqrt[k]{|\overline{p}_k|} \tag{2.2}$$

The vector $(\mu, m_2, \ldots, m_l)$ of l floating point numbers roughly describes the underlying distribution and can be used as a feature vector. Note that $\overline{p}_k$ can be negative if k is odd, which explains why in Equation 2.2 the k-th root is applied to its absolute value. Nevertheless, m_k receives the same sign as $\overline{p}_k$ in Equation 2.2, and it is noteworthy that the feature vector $(\mu, m_2, \ldots, m_l)$ is neither normalised nor non-negative in contrast to histogram feature vectors.

[3] These rules of thumb can fail for multimodal distributions or those where the shorter tail is correspondingly "fatter" as to compensate the weight of the longer tail.

2.2.3 Texture Histograms

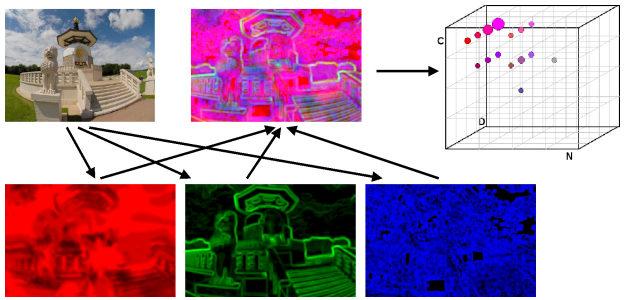
Of course, the features of Figures 1.7 and 2.3 are very simple, and the features that we compute are normally more complex than that. For example, Howarth and Rüger (2005b) studied and devised ways to extract texture descriptions from images. Tamura et al (1978) have found out through psychological studies that we humans respond best to coarseness, contrast, and directionality as visualised in Figure 2.4, and to a lesser degree to line-likeness, regularity and roughness.



coarseness contrast directionality

Figure 2.4: Textures

Unlike colour, which is a property of a pixel, texture is a property of a region of pixels, so we need to look at an area around a pixel before we can assign a texture to that pixel. Figure 2.5 is an example, how we compute for each point in an image (by considering a window around this point) a coarseness (C) value, a contrast value (N) and a directionality value (D). These values can be assembled into a single false-colour image, where the red, blue and green channels of an ordinary image are replaced by C, N and D, respectively. This expresses visually the use and perception of texture in an image. From the false-colour texture image, we can then compute 3d texture histograms exactly in the same way as we do for colour images.



coarseness contrast directionality

Figure 2.5: 3d texture diagram via false-colour images

Textures are normally computed from greyscale images — although there is nothing to prevent one from extracting textures from colour channels or from studying the patterns that colours of similar greyscale introduce.

We will first study in depth how Tamura texture features can be computed.

Tamura Texture Features

Coarseness has a direct relationship to scale and repetition rates and was seen by Tamura et al (1978) as the most fundamental texture feature. An image will contain textures at several scales; coarseness aims to identify the largest size of these scales, even where a smaller regular pattern exists. First, we take a moving average at every point over $2^k \times 2^k$ windows, $k < 6$. This average at the point (x, y) is

$$a_k(i,j) = \sum_{i'=i-2^{k-1}}^{i+2^{k-1}-1} \sum_{j'=j-2^{k-1}}^{j+2^{k-1}-1} \frac{p(i',j')}{2^{2k}},$$

where $p(i, j)$ is the grey level at the image pixel coordinates (i, j). Then one computes the bigger of the horizontal and vertical differences of a_k at the edge of the window:

$$c_k(i,j) = \max(\ |a_k(i-2^{k-1},j) - a_k(i+2^{k-1},j)|, \\ |a_k(i,j-2^{k-1}) - a_k(i,j+2^{k-1})|)$$

This value will differ with k, and the

$$\hat{k}(i,j) = \text{argmax}_k\, c_k(i,j)$$

that maximises c_k indicates the biggest detected scale $2^{\hat{k}(i,j)}$ at the point (i, j). The coarseness value for a whole picture is then averaged as

$$\text{coarseness} = \frac{1}{wh} \sum_{(i,j)} 2^{\hat{k}(i,j)}.$$

When carrying out these calculations one has to be careful not to exceed the area of the original image. The moving average a_k can only be computed in a smaller area, and c_k can only be computed in a smaller area still, see Figure 2.6.

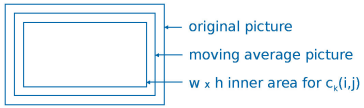


Figure 2.6: Domain over which coarseness can be computed

Contrast aims to capture the dynamic range of grey levels in an image together with the polarisation of the distribution of black and white. The first quantity is measured using the variance

$$\sigma^2 = \frac{1}{wh} \sum_{(i,j)} (p(i,j) - \bar{p}_1)^2$$

of grey levels, while the second quantity is obtained by the

$$\alpha_4 = \bar{p}_4/\sigma^4 = \frac{1}{wh} \sum_{(i,j)} (p(i,j) - \bar{p}_1)^4/\sigma^4.$$

$\bar{p}_1$ stands for the average grey value. The contrast measure is defined as

$$\text{contrast} = \sigma/(\alpha_4)^n$$

Experimentally, Tamura found $n = 1/4$ to give the closest agreement to human measurements.

Directionality is a global property over a region. The feature described does not aim to differentiate between different orientations or patterns, but it measures the total degree of directionality. At each pixel, a gradient

$$g = \begin{pmatrix} \Delta_h \\ \Delta_v \end{pmatrix} \text{ with}$$
$$\Delta_h = \sum_{k\in\{-1,0,1\}} p(i+1,j+k) - p(i-1,j+k) \text{ and}$$
$$\Delta_v = \sum_{k\in\{-1,0,1\}} p(i+k,j+1) - p(i+k,j-1)$$

is computed. The gradient computation corresponds to convolving the image with two simple masks for edge detection,

$$\begin{pmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{pmatrix} \text{ and } \begin{pmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{pmatrix},$$

respectively. The gradient g is then transformed into bi-polar coordinates

$$(|g|, \phi) = \left(\frac{|\Delta_h| + |\Delta_v|}{2}, \tan^{-1}\left(\frac{\Delta_v}{\Delta_h}\right) + \frac{\pi}{2} \right),$$

which reveal size and direction of the gradient. The next step is to compute a histogram over quantised angles for those gradients with a size $|g|$ larger than a certain threshold. The histogram bin $h\phi(k)$ counts the proportion of those pixels above threshold for which

$$\frac{2k-1}{2n} < \phi/\pi \leq \frac{2k+1}{2n} (\text{mod } 1).$$

n determines the granularity of the histogram of edge directions. This histogram reflects the degree of directionality, see Figure 2.7. To extract a measure, the sharpness of the histogram peaks is computed from their second moments.

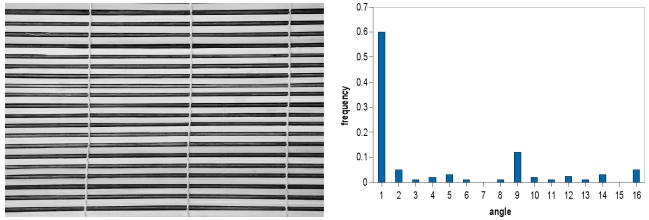


Figure 2.7: Example image and directionality histogram

2.2.4 Shape

Shape is commonly defined as an equivalence class of geometric objects invariant under translations, rotations and scale changes that keep the aspect ratio. Many retrieval applications require global scale invariance, ie, relative sizes are still meaningful. Hence, I will not require scale invariance in the following when looking at shapes. Strict scale invariance would imply that the sizes of objects do not matter at all.

Shape representations can preserve the underlying information, so the shape can be reconstructed or they can simply aim at keeping interesting aspects. The former type is used for compression while the latter may be sufficient for retrieval. There are a number of boundary-based features than can be extracted — these ignore the interior of shapes including holes in it. Other shape features are region-based.

This section assumes that we have a representation of the shape to be analysed at hand. It is normally non-trivial to separate background objects from foreground objects in pixel-images, and this theme is outside the scope of this book. There are some cases, however, where this separation is easily possible because the shape objects already exist in parameterised form, for example, as vector graphics, or in different layers owing to the production method of the multimedia object. Some media, for instance, comic strips or cartoon movies, lend themselves to simple object separation.

Boundary-based Shape Features

The *boundary* of a 2d shape can be described mathematically in terms of a parameterised curve:

$$\begin{aligned} B\colon [0,1] &\to \mathbb{R}^2 \\ t &\mapsto (x(t), y(t)) \end{aligned}$$

Normally, one would expect B to be continuous, and a continuous boundary B is called closed if and only if $B(0) = B(1)$.

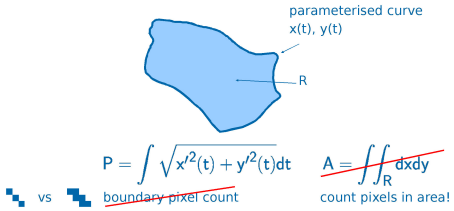


Figure 2.8: Perimeter and area of a shape

The *perimeter of a shape* is the length

$$P = \int \sqrt{x'(t)^2 + y'(t)^2} dt$$

of its boundary $t \mapsto (x(t), y(t))$. Here f' denotes the derivative $\partial f / \partial t$ of a function f with respect to t. If you have a representation of a boundary in form of pixels, it is normally not good enough to count the pixels in this representation, as the digitisation of the mathematical line consumes a

different amount of pixels at different angles and apparent line thicknesses, see Figure 2.8. This is in stark contrast to the best way of determining the *area of a shape* that can easily be approximated by counting the pixels that it fills.

The *convexity of a shape* is one important characteristics. A convex region is one, where the connecting line between any two points of the region lies within it. It is always possible to construct a convex hull from a region, see Figure 2.9. The ratio of the perimeter of the convex hull and the perimeter of the original boundary is called *convexity*. It is 1 for convex shapes and less than 1 for non-convex shapes.

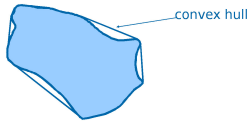


Figure 2.9: Convexity and convex hull

A similar idea is behind the characteristics of the circularity of a shape, which is defined as

$$T = 4\pi \frac{A}{P^2},$$

where A is the area of the shape and P is its perimeter. The circularity of a shape is 1 for a circle and less than one for other shapes.

Corners in a parameterised curve are places with a high curvature

$$c = \frac{x'y'' - y'x''}{(x'^2 + y'^2)^{3/2}}.$$

The number of corners is another possible characteristics for a shape.

An early boundary representation with strings that can be used for recognition purposes is that of *chain codes* proposed by Freeman (1961). They approximate curves with a sequence of vectors lying on a square grid. In its simplest form, each pixel in a curve has eight neighbours, numbered counter-clockwise from 0 to 7 starting with the neighbour to the right, see Figure 2.10. A line can be encoded by following the pixels in the line one by one, each time making a note of the neighbour number. Figure 2.10 illustrates this with the small circle-like contour. We begin encoding it at a pixel in the middle-left, move up one position (2), move diagonally up to the right (1), then to right (0), and so on. For closed curves the resulting chain code depends on where we started and in which direction. One way out of this is to assign the representation with the smallest number, here 007765434321.

Figure 2.10: Freeman chain code

However, this representation is still not rotation invariant. To achieve the latter one can transform a chain code $(f_1, f_2, \ldots, f_n)$ into a *difference chain code* via a sequence of angles

$$a_i = (f_{i+1} - f_i) \mod 8 \tag{2.3}$$

rather than directions. Figure 2.11 illustrates the 8 different angles with their codes from 0 to 7 and that the difference chain code of the rotated figure remains the same.

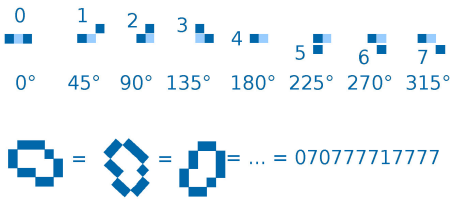


Figure 2.11: Difference chain codes are rotation invariant

Finally, as seen in Figure 2.12, one can then summarise individual curves into histograms.

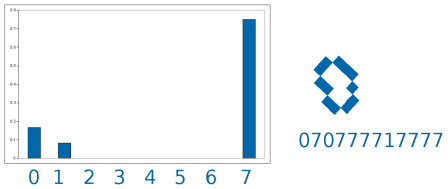


Figure 2.12: Histogram of difference chain code

A well-established technique to describe closed planar curves is the use of Fourier descriptors (Zahn and Roskies, 1972; Persoon and Fu, 1977). Using the Fourier descriptors one can achieve representational invariance with respect to a variety of affine transformations; Wallace and Wintz (1980) have successfully used these descriptors for recognition tasks.

For the computation of Fourier descriptors, the contour pixels at coordinates (x, y) need to be represented as complex numbers $z = x + æy$. For closed contours, we get a periodic function which can be expanded into a convergent Fourier series. Specifically, let Fourier descriptor C_k be defined as the kth discrete Fourier transform coefficient

$$C_k = \sum_{n=0}^{N-1} (z_n e^{\frac{-2\pi æ k n}{N}}),$$

$-N/2 \leq k < N/2$, which we compute from the sequence of complex numbers $z_0, z_1, \ldots, z_{N-1}$ where N is the number of contour points. To characterise contour properties any constant number of these Fourier descriptors can be used. The most interesting descriptors are those of the lower frequencies as these tend to capture the general shape of the object. Translation invariance is achieved by discarding C_0, rotation and starting point invariance by further using only absolute values of the descriptors, and scaling invariance is brought about by dividing the other descriptors by, say, $|C_1|$. The final feature vector has the form

$$\left(\frac{|C_{-L}|}{|C_1|}, \ldots, \frac{|C_{-1}|}{|C_1|}, \frac{|C_2|}{|C_1|}, \ldots, \frac{|C_L|}{|C_1|} \right)^T,$$

where L is an arbitrary constant between 2 and $N/2 - 1$. The thus derived feature vector has a good theoretical foundation and a clear interpretation, which makes it easy to decide on granularity and size of the vector to chose.

Region-based Shape Descriptors

We model a bounded region of a shape as a function

$$\begin{aligned} f: \mathbb{R}^2 &\rightarrow \{0, 1\} \\ (x, y) &\mapsto \begin{cases} 1 & \text{if pixel at } (x, y) \text{ is on} \\ 0 & \text{otherwise.} \end{cases} \end{aligned}$$

It turns out that the knowledge of all its 2d moments

$$M_{ij} = \sum_{(x,y)} x^i y^j f(x, y), \quad i, j \in \mathbb{N},$$

is sufficient to reconstruct f and with it the region. Two useful entities are M_{00}, which is the area of the region, and

$$(\bar{x}, \bar{y}) = \left(\frac{M_{10}}{M_{00}}, \frac{M_{01}}{M_{00}} \right),$$

which is the centre of mass of the region, see Figure 2.13.

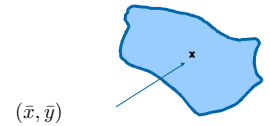


Figure 2.13: Centre of mass expressed with 2d moments

The *central moments*

$$C_{ij} = \sum_{(x,y)} (x - \bar{x})^i (y - \bar{y})^j f(x, y),$$

which are centred around $(\bar{x}, \bar{y})$, are translation invariant by construction. If these moments are then divided by the right power of the region's area then the ensuing *normalised central moments*

$$c_{ij} = C_{ij} / M_{00}^{1+(i+j)/2}$$

are both translation and scaling invariant. It is possible to construct the following 7 descriptors,

which are rotation invariant in addition:

$$
\begin{aligned}
I_1 &= c_{02} + c_{20} \\
I_2 &= (c_{20} - c_{02})^2 + 4c_{11}^2 \\
I_3 &= (c_{30} - 3c_{12})^2 + (3c_{21} - c_{03})^2 \\
I_4 &= (c_{30} + c_{12})^2 + (c_{21} + c_{03})^2 \\
I_5 &= (c_{30} - 3c_{12})(c_{30} + c_{12})((c_{30} + c_{12})^2 - 3(c_{21} + c_{03})^2) + \\
&\quad (3c_{21} - c_{03})(c_{21} + c_{03})(3(c_{30} + c_{12})^2 - (c_{21} + c_{03})^2) \\
I_6 &= (c_{20} - c_{02})((c_{30} + c_{12})^2 - (c_{21} + c_{03})^2) + \\
&\quad 4c_{11}(c_{30} + c_{12})(c_{21} + c_{03}) \\
I_7 &= (3c_{21} - c_{03})(c_{30} + c_{12})((c_{30} + c_{12})^2 - 3(c_{21} + c_{03})^2) + \\
&\quad (3c_{12} - c_{30})(c_{21} + c_{03})(3(c_{30} + c_{12})^2 - (c_{21} + c_{03})^2)
\end{aligned}
$$

These 7 numbers are called *invariant moments*. I_2 is also known as eccentricity. Although they are useful descriptors for shapes, invariant moments are not very expressive, and it is not clear how one could use these to scale up complex shapes. One can compute other useful properties of shapes with normalised central moments, though: the *orientation of a shape*, an angle indicating its main axis, is given by

$$\phi = \frac{1}{2}\tan^{-1}\left(\frac{2c_{11}}{c_{20} - c_{02}}\right),$$

see Figure 2.14.

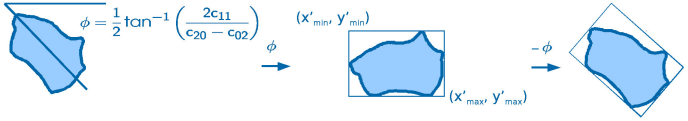


Figure 2.14: Bounding box computation via orientation angle ϕ

Rotating the shape by this angle ϕ

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} \cos(\phi) & \sin(\phi) \\ -\sin(\phi) & \cos(\phi) \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix}$$

makes it easy to determine a minimal bounding box. Let $(x'_{\min}, y'_{\min})$ and $(x'_{\max}, y'_{\max})$ be the extreme coordinates of the rotated bounding box, then you can identify the coordinates of the minimal bounding box by rotating about $-\phi$.

All of the above-mentioned shape representations have in common that they can be stored as a simple vector. More elaborate shape representations have been introduced some of which are the curvature scale space representation or the spline curve approximation which require sophisticated shape matching techniques (del Bimbo and Pala, 1997).

2.2.5 Spatial Information

Tiled Histograms

Histograms as in Figure 1.7 are useful instruments, but they are very crude indicators of similarity. For example an eight-bin histogram of intensity values of an image is a simple approximation of its

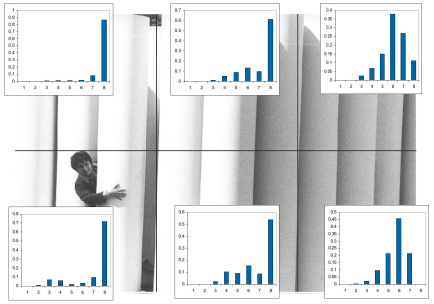


Figure 2.15: The feature vector of tiled images

brightness distribution, and many images do indeed share the same histogram. Figure 2.16 shows a woman in the middle of bright column sculptures, but an image of a skier in snow is likely to have the same intensity histogram. The other disadvantage of global histograms computed over the whole image is that they lose all locality information. One simple solution is to tile an image into $n \times m$ rectangles of the same size, each of which creates a histogram (see Figure 2.15). The full feature vector is then the concatenation of the feature vectors of individual tiles and, in this case, contains $6 \cdot 8$ numbers.

Designing Different Areas of Importance

Different areas in images carry different importance: computing separate histograms for its centre and the border region allows one to focus more on one than on the other. Figure 2.16 is an example of a centre-border intensity histogram, where two histograms are computed: as the centre area is much bigger than the border area the corresponding proportion of pixels that fall into centre area intensity ranges is typically larger than for intensity ranges of the border area. This gives the centre extra weight.

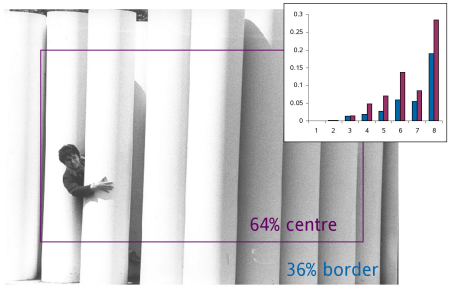


Figure 2.16: Centre-border intensity histogram of an image

The photographic composition principle of the "rule of thirds" implies that lines or objects of interest work best in a photograph if they are off-centre, roughly one third or two thirds into the image. A similar, but slightly more evolved aesthetic principle, suggests composing interesting objects at points that divide the height or width of an image at the golden ratio of $(\sqrt{5}-1)/2 \approx 0.618$ as opposed to $2/3$. It is a surprising fact that the 25% of the central area of an image still captures all these points of interest with a generous margin. Figure 2.17 contrasts global histograms with three other schemes that aim at recovering crude localisation information. *Focal* histograms only consider the 25% interior of any image, while *central* histograms put much weight on the main histogram of the central 50% area but retain some information about the background in four sub-histograms of less weight. Finally, *local* histograms create five sub-histograms of equal weight, four of which are around the possible four quadrants of interest, while the fifth caters for the background.

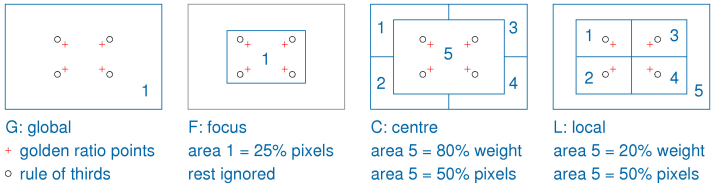


Figure 2.17: Different strategies to capture essential areas in photographs

Structure Histograms

The MPEG-7 *colour structure histogram* folds the frequency of a colour with information how concentrated or scattered it occurs in an image. In the same way as in global colour histograms, each pixel is assigned a corresponding bin (either as 1d intensity or 3d colour bin). However, here a gliding 8×8 window moves over the image, and each histogram bin counts the *number of overlapping windows* that contain at least one pixel of the corresponding bin. An image that has a concentrated occurrence of blue in the top will have a smaller count for the blue cell than another image, where the same number of blue pixels are peppered across the whole image. Messing et al (2001) claim that, for this reason, the colour structure descriptor outperforms other colour descriptors. It should be noted that the structuring element of the colour structure descriptor is applicable to all qualities of spatial arrangements, eg, the false-colour texture representation above.

Normalisation is through dividing by the number of different window positions; each component is thus between 0 and 1, but the sum can, in rare cases, be as large as 64, the number of pixels in the gliding window.

Localised Features

One quite successful idea to exploit local structure is to compute points of interest. These are salient points in images that give rise to encoding features in limited area. Figure 2.18 illustrates the idea, but algorithms that compute points of interests normally compute many more regions.

Lowe (1999, 2004) has developed a popular *scale-invariant feature transform* (SIFT) that detects and encodes local features in images. The first step is to detect candidate points of interest in an image by convolving a 2d Gaussian function of different scale σ with the image (basically a

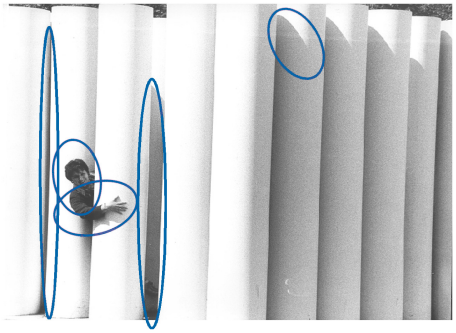


Figure 2.18: Points and regions of interest

smoothing or blurring of the image with different radius) and taking differences of the resulting function (blurred image) at each point with respect to slightly different scales σ and $k\sigma$. The extrema of this function, called difference of Gaussians, indicate candidate points of interest. From these key-points are localised and orientations assigned, which serve as a local reference coordinate system. The final features that are extracted from this area are computed relative to this local reference, so that they are encoded in a scale, rotation and location invariant manner. A typical image exhibits in the order of 2000 key-points.

Using points of interests with localised features allows one to quantise the latter into so called *visual words*. Being quantised one can deploy the same type of retrieval techniques that are so successful for text retrieval: inverted-file indices. Localised features and quantisation is a very powerful combination for duplicate detection.

2.2.6 Other Feature Types

There is a sheer abundance of different feature types. Above were a mere selection of features for colour, texture and shape features that are useful for visual still image retrieval. They demonstrate the type of processing from counting to Fourier transforms that is typical for low-level processing.

It should be noted that MPEG-7 (see Section 1.1 on page 8) has defined the following set of features for low-level audiovisual content description:

- colour: colour space, colour quantization, dominant colour(s), scalable colour, colour layout, colour-structure descriptor, GoF/GoP colour
- texture: homogeneous texture, texture browsing, edge histogram
- shape: region shape, contour shape, shape 3d
- motion: camera motion, motion trajectory, parametric motion, motion activity

I have described a single one of them, the colour-structure descriptor, just above.

A deep treatment of music and audio features is beyond the scope of this book. I refer to Liu et al (1998), who reviewed a good range of audio features.

2.3 Distances

Most of above features create real-valued vectors of a fixed dimension n, where distance computation is straightforward. We will first introduce standard distance measures that work component-wise such as the Minkowski distance, then distances that include cross-component correlations such as the Mahalanobis distance between vectors, followed by statistical and probabilistic distance measures including the popular mover's distance between probability distributions, and those that work on a string level.

2.3.1 Geometric Component-wise Distances

Let $v, w \in \mathbb{R}^n$. The so-called *Minkowski norm*

$$\begin{aligned} \mathrm{L}_p : \mathbb{R}^n &\to [0, \infty) \\ w &\mapsto |w|_p = \left(\sum_{i=1}^{n} |w_i|^p \right)^{1/p} \end{aligned}$$

induces a distance

$$d_p(v, w) = \mathrm{L}_p(w - v).$$

L_p is a norm for all real values $p \in [1, \infty]$ meaning $\mathrm{L}_p(v) = 0$ if and only if $v = 0$; $\mathrm{L}_p(av) = |a| \, \mathrm{L}_p(v)$ for all $a \in \mathbb{R}$ and $v \in \mathbb{R}^n$, and $\mathrm{L}_p(v+w) \leq \mathrm{L}_p(v) + \mathrm{L}_p(w)$ for all $v, w \in \mathbb{R}^n$. The latter is also known as triangle inequality. The induced distance fulfills corresponding axioms.

One special case for $p = 2$ is also known as the *Euclidean norm*, which is the length of a line between points in space. Another special case, $p = 1$, yields the *Manhattan norm*, which corresponds to the total length of a sequence of lines parallel to the coordinate axes from point v to point w. $d_1(v, w)$ is called Manhattan distance, because cars in Manhattan can only go along the grid system of perpendicular streets and avenues (ignoring the existence of the diagonal Broadway). As p increases beyond 2, the sum in $d_p(v, w)$ is more and more dominated by the largest difference $|v_i - w_i|$ of components, and as p approaches infinity $d_p(v, w)$ will be identical to the maximum of all values $|v_i - w_i|$ over i. Hence L_∞ is also called *maximum norm* or *Chebyshev norm*. However, this norm is not particularly useful for multimedia retrieval as it implies that the distance between two media representations v and w is solely determined by the biggest non-matching component, which may as well be an outlier, and hence irrelevant to our perception of similarity. It turns out that a p at the lower end of the spectrum normally give better retrieval results owing to a larger emphasis on components that actually match.

This has led to exploring values of p in the area of $(0, 1)$ for nearest neighbour search (Aggarwal et al, 2001). Technically, L_p is no longer a norm for $p < 1$ as the triangle inequality is violated, but it is still possible to order the induced dissimilarity[4] values accordingly. Howarth and Rüger (2005a) have found best retrieval results for many feature vectors types with p values between 0.5 and 0.75.

A special dissimilarity measure for histograms, called partial histogram intersection, is given by

$$d_{\mathrm{phi}}(v, w) = 1 - \frac{\sum_i \min(v_i, w_i)}{\max\left(\mathrm{L}_1(v), \mathrm{L}_1(w)\right)}.$$

[4] we prefer the term dissimilarity when not all mathematical axioms for distances are valid

In this context, v and w are not necessarily normalised histograms, but they ought to be in the same range to make sense. The components of v and w are expected to be non-negative. This dissimilarity measure is very popular for normalised histogram features, and for those equivalent to the Manhattan distance (see Exercise 2.7.6).

There are other geometrically motivated distances between vectors, for example, the *Canberra distance*

$$d_{\mathrm{Can}}(v, w) = \sum_{i=1}^{n} \frac{|v_i - w_i|}{|v_i| + |w_i|},$$

which is very sensitive for components near zero. Note that the term $|v_i - w_i|/(|v_i| + |w_i|)$ needs to be replaced by zero if both v_i and w_i are zero.

A common dissimilarity measure in Information Retrieval is the *cosine dissimilarity* defined as

$$d_{\cos}(v, w) = 1 - \frac{v \cdot w}{\mathrm{L}_2(v) \, \mathrm{L}_2(w)}$$

named after the fact that the normalised scalar product $v \cdot w / \mathrm{L}_2(v) \, \mathrm{L}_2(w)$ between v and w is the cosine of the angle between the vectors v and w. $d_{\cos}$ is not a distance, as one cannot conclude from $d_{\cos}(v, w) = 0$ that $v = w$. This is owing to the fact that the length of v and w is irrelevant and only their direction is used for the computation.

The *Bray-Curtis dissimilarity* is derived from the Manhattan distance as

$$d_{\mathrm{BC}}(v, w) = \frac{d_1(v, w)}{d_1(v, -w)} = \frac{\sum_i |v_i - w_i|}{\sum_i |v_i + w_i|}$$

and approaches infinity as v approaches $-w$.

A less usual measure is given by the *squared chord dissimilarity*,

$$d_{\mathrm{sc}}(v, w) = \sum_{i=1}^{n} (\sqrt{v_i} - \sqrt{w_i})^2,$$

which seems to have been used in paleontological studies and in pollen data analysis, both with little theoretical justification. In comparative evaluation, the squared chord measure does remarkably well though (Hu et al, 2008). Please note that the squared chord dissimilarity cannot work with negative components; it should be replaced with a modified version

$$d_{\mathrm{msc}}(v, w) = \sum_{i=1}^{n} \left(\mathrm{sign}(v_i) \sqrt{|v_i|} - \mathrm{sign}(w_i) \sqrt{|w_i|} \right)^2$$

instead.

2.3.2 Geometric Quadratic Distances

If two images only differ by lighting, then their respective colour histograms will be shifted. All pixels that would end up in the white bin in one image might end up in the light-yellow bin in the other image. Component-wise distance measures will not recognise that the light-yellow bin is not so different from the white bin: they only see a mismatch, and the white pixels might as well have ended up in the dark-blue bin.

One way to recognise the closeness of the feature components that the individual bins represent is through a quadratic matrix A that maps two bins to a number that represents how similar their underlying features are. In the case of colour histograms, A_{ij} might be set to a similarity of the colour triplets that represent the bins in, say, RGB colour space $[0,1]^3 \ni (r,g,b)$:

$$A_{ij} = 1 - |(r_i, g_i, b_i) - (r_j, g_j, b_j)|_\infty = 1 - \max(|r_i - r_j|, |g_i - g_j|, |b_i - b_j|)$$

is one such example of how the matrix A could be created. The *feature quadratic distance* is then defined as

$$d_{\text{fq}}(v,w) = \sqrt{(v-w)^t A (v-w)}.$$

This distance measure is identical in form to the *Mahalanobis distance* between two random vectors v and w of the same distribution with the covariance matrix S, only that A is replaced by S^{-1}. If S (or A) is the unit matrix then d_{fq} collapses to the Euclidean distance d_2. If the co-variance matrix S is diagonal (or A is), then d_{fq} is the so-called normalised Euclidean distance

$$d_2^\sigma(v,w) = \sqrt{\sum_i \frac{(v_i - w_i)^2}{\sigma_i^2}}.$$

Since the histogram quadratic distance computes the cross similarity between features, it is computationally more expensive than component-wise distance measures.

2.3.3 Statistical Distances

Histograms approximate distributions of the underlying quantity, and normalised histograms can be interpreted as probability distributions themselves. There is a number of dissimilarity measures that are derived from a statistical motivation. For example, the χ^2 statistics

$$d_{\chi^2}(v,w) = \sum_i \frac{(v_i - m_i)^2}{m_i}$$

with $m = (v+w)/2$ measures how different two frequency distributions are.

A whole raft of distances can be defined by converting normalised histograms v into cumulative histograms $\hat{v}$ via

$$\hat{v}_i = \sum_{j \le i} v_j.$$

Plugging cumulative histograms into L_p defines cumulative distances

$$\hat{d}_p(v,w) = d_p(\hat{v}, \hat{w}) = \mathrm{L}_p(\hat{v} - \hat{w}).$$

$\hat{d}_1$ is also known as *match distance*, $\hat{d}_2$ is known as *Cramér-von-Mises-type distance*, while $\hat{d}_\infty$ is also known as *Kolmogorov-Smirnov distance* (Puzicha et al, 1997). Note, however, that cumulative histograms — and hence cumulative distances — are only defined for histograms over a one-dimensional space, i.e., histograms with bins that can be linearly ordered. 3d colour histograms, for example, do not have a cumulative version, as it is not immediately clear how the two colours j and i could be intrinsically ordered such that $j \le i$.

Another measure, derived from the Pearson correlation coefficient, is defined as

$$d_{\text{pcc}}(v,w) = 1 - |r|,$$

where the correlation coefficient

$$r = \frac{n \sum_{i=1}^n v_i w_i - \left(\sum_{i=1}^n v_i\right)\left(\sum_{i=1}^n w_i\right)}{\sqrt{\left[n \sum_{i=1}^n v_i^2 - \left(\sum_{i=1}^n v_i\right)^2\right]\left[n \sum_{i=1}^n w_i^2 - \left(\sum_{i=1}^n w_i\right)^2\right]}}.$$

is a number between -1 and 1, where -1 corresponds to a strong negative correlation (small v_i correspond to large w_i and vice versa), $r = 0$ corresponds to uncorrelated distributions r and w, while positive values for r indicate a positive correlation. Note that both strong positive and strong negative correlation of v and w yield small distances.

2.3.4 Probabilistic Distance Measures

The *Kullback-Leibler divergence* expresses the degree of discrepancy between two probability distributions v and w, and measures the extra information needed to express a sample from the "true" distribution v when one encodes them with samples from an approximation distribution w. For discrete random vectors v and w, their Kullback-Leibler divergence is defined as

$$d_{\text{KL}}(v,w) = \sum_i v_i \log \frac{v_i}{w_i}.$$

d_{KL} is not a distance: it is not even symmetric and grows arbitrarily big for any component of w approaching 0, which makes the Kullback-Leibler divergence unsuitable for many feature types that tend to have vanishing components. This measure can be made symmetric through the definition of the *Jensen-Shannon divergence*

$$d_{\text{JS}}(v,w) = (d_{\text{KL}}(v,m) + d_{\text{KL}}(w,m))/2,$$

where $m = (v+w)/2$. The Jensen-Shannon divergence has the added benefit of being finite. It needs to be re-emphasised that both the Kullback-Leibler and the Jensen-Shannon divergence are only defined for probability vectors, ie, with non-negative components that sum to one. These measures would not be suitable for a number of feature types, for example, those that describe distributions with their central moments, as they can be negative.

Another intuitive distance measure between two probability distributions over a space D is the *earth mover's distance*: here, one distribution v is defined as an amount of earth piled up in regions of D, while the other distribution w is defined as an amount of holes distributed in regions of D. The earth mover's distance between v and w is defined as the minimum cost of moving the piles of earth into the holes, cost being defined as the amount of earth moved times the distance it is moved.

It is important to realise that the distributions are defined over some space D with a distance called *ground distance*. For instance, v and w might be colour distributions over a colour space, say, RGB, where the ground distance is defined as some distance between colours. In this sense, the earth mover's distance is similarly expressive as the feature quadratic distance, but it has the advantage of not needing a fixed partition of the space D that histograms would give. Instead a distribution v can be defined as a list of n^v cluster centres $c_i^v \in D$ and corresponding masses $m_i^v \in [0,1]$ (eg, proportion of pixels that fall into this cluster):

$$v = ((c_1^v, m_1^v), (c_2^v, m_2^v), \ldots, (c_{n^v}^v, m_{n^v}^v))$$

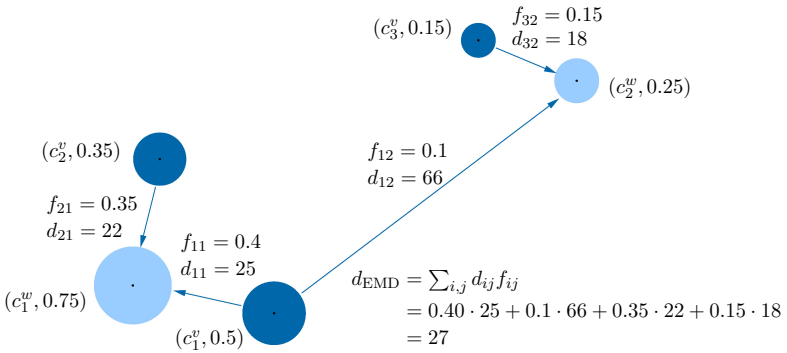


Figure 2.19: Earth mover's distance between two signatures

These lists are also known as *signatures*; note that they have variable length and generalise the notion of histograms. Figure 2.19 illustrates two signatures v, w over $D = \mathbb{R}^2$ and shows the optimal way of morphing v into w.

Although the earth mover's distance can be defined where both signatures have a different total mass, we restrict ourselves to signatures with a total mass of 1 each. A formal way of describing the earth mover's distance between any two such signatures v and w involves the definition of a ground distance

$$d_{ij} = d(c_i^v, c_j^w)$$

and a flow f_{ij} of mass between the i-th cluster centre of v and the j-th cluster centre of w. This flow is subject to the constraints

$$f_{ij} \geq 0 \tag{2.4}$$

$$\sum_j f_{ij} = m_i^v \tag{2.5}$$

$$\sum_i f_{ij} = m_j^w \tag{2.6}$$

for all $1 \leq i \leq n^v$ and all $1 \leq j \leq n^w$. (2.4) stipulates that mass can only flow in the direction from v to w, ie, from the piles of earth into the holes, (2.5) that the total flow from a pile must be equal its size, while (2.6) means that the total flow into a hole must be equal to the capacity of the hole.

The earth mover's distance is now defined as the minimum cost to shift all mass from v to w with respect to all possible flows f that fulfill constraints (2.4)–(2.6):

$$d_{\text{EMD}}(v, w) = \min_f \sum_{i,j} f_{ij} d_{ij}$$

Figure 2.19 illustrates one such cost computation. The problem of finding the flow f with the minimal cost is also known as transportation problem for which efficient algorithms exist. Yossi

Rubner distributes a C programme[5] based on an algorithm from Hillier and Lieberman's textbook (1990).

Rubner et al (2000) showed that d_{EMD} applied to mass 1 signatures is a distance, ie, fulfills the triangle inequality if the ground distance is a distance; they also examined the use of the earth mover's distance for image retrieval where it outperforms other distances. While the earth mover's distance is made for signatures it can, in theory, be applied to histograms as well. However, large histogram bins bring about quantisation errors while small bins make the transportation algorithm run slowly. It appears to be advantageous to apply the earth mover's distance to signatures rather than fixed-bin histograms.

2.3.5 Ordinal and Nominal Distances

When feature values are non-numeric, ie, general strings, none of the above distance measures can be used. An exception to this are ordinal labels, ie, those which can be ranked or have a numeric rating scale attached. These labels can be mapped to numbers: for example, *small*, *normal*, *large* and *extra large* can be mapped to their ranks 4, 3, 2, and 1, respectively.

In all other cases, the string attributes are a known as nominal labels and a matching coefficient

$$m(v, w) = \sum_i \delta(v_i, w_i) \quad \text{with} \quad \delta(v_i, w_i) = \begin{cases} 1 & \text{if } v_i = w_i \\ 0 & \text{otherwise} \end{cases}$$

can be transformed into a distance measure

$$d_{\text{mc}}(v, w) = 1 - \frac{m(v, w)}{n}.$$

The matching coefficient simply $m(v, w)$ counts the number of coordinates, whose strings match.

The same matching coefficient can be used with binary feature components. There is an important case, though, where the binary data are *asymmetric*, ie, the outcome one expresses a rare and important case while zero encodes a normal and frequent situation. This is the case in sparse vector representations of "bags of words", where a text document is represented as a vector telling which words out of the whole repository vocabulary are present and which are not. In this representation, d_{mc} between two documents would decrease for each word that both do *not* contain. For example, the fact that two documents *did not* contain the word "antidisestablishmentarianism" would make them less distant in the same way as if the word "electroencephalogram" *did* occur in both. This is clearly not desirable! It is best to ignore matching frequent cases in those asymmetric encodings as is done in the Jaccard distance

$$d_{\text{Jaccard}}(v, w) = 1 - \frac{m^1(v, w)}{n - m^0(v, w)},$$

where $m^b(v, w)$ counts the number of coordinates, whose binary value is both b:

$$m^b(v, w) = \sum_i \delta(v_i, b)\delta(w_i, b) \quad \text{with} \quad \delta(a, b) = \begin{cases} 1 & \text{if } a = b \\ 0 & \text{otherwise} \end{cases}$$

[5] http://robotics.stanford.edu/~rubner/emd/default.htm

2.3.6 String-based Distances

The matching coefficient (2.3.5) takes into account how many of the feature vector's strings match literally. There are many other more fine-tuned distance functions, some working on a syntactic character level, for example, the edit distance, and some working on a semantic level, for example, using WordNet, web search engines or ontologies.

Levenshtein's edit distance transforms one chain $s = (s_1, s_2, \ldots, s_{n^s})$ of n^s symbols $s_i \in S$ into another one t with potentially different length n^t, and is defined as the minimum overall transformation cost that arises through deletions (cost c_d), insertions (cost c_i) and exchanges (cost $c_\mathrm{x}\delta(s_j, t_i)$). Computing the minimum cost is normally done via dynamic linear programming that computes an $n^t \times n^s$ matrix c:

1. $c[0,0] = 0$
2. $c[0,j] = j \cdot c_\mathrm{d}$ for all $j \in \{1, \ldots, n^s\}$
3. $c[i,0] = i \cdot c_\mathrm{i}$ for all $i \in \{1, \ldots, n^t\}$
4. $c[i,j] = \min(c[i-1, j-1] + c_\mathrm{x}\delta(t_i, s_j), c[i, j-1] + c_\mathrm{d}, c[i-1, j] + c_\mathrm{i})$
   for all $(i,j) \in \{1, \ldots, n^t\} \times \{1, \ldots, n^s\}$ (small indices first)

The Levenshtein distance between s and t ends up in the matrix element $c[n^t, n^s]$. Figure 2.20 illustrates this process using the strings $s = hello$ and $t = halo$ with an insertion cost of $c_\mathrm{i} = 1.01$, a deletion cost of $c_\mathrm{d} = 1.1$ and an exchange cost $c_\mathrm{x} = 1$ for different letters. The optimal path of operations is marked with arrows from the top left matrix element $c[0,0]$ to the bottom right matrix element $c[n^t, n^s]$. It involves replacing h with h (cost 0), replacing e with a (cost 1), replacing l with l (cost 0), deleting l (cost 1.1) and replacing o with o (cost 0) totalling a cost of 2.1, which is the smallest possible.

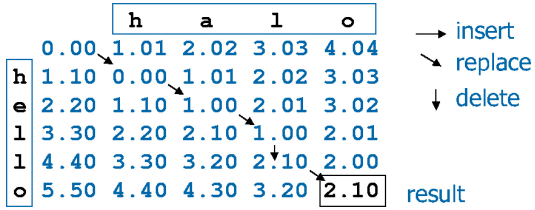


Figure 2.20: Levenshtein distance between *hello* and *halo* with $c_\mathrm{d} = 1.1$, $c_\mathrm{i} = 1.01$ and $c_\mathrm{x} = 1$

The algorithm is $O(n^s n^t)$ in time and can be made $O(n^t)$ in memory by observing that the algorithm only ever needs to have access to the preceding matrix row. Levenshtein (1966) introduced a version of this distance for which $c_\mathrm{x} = c_\mathrm{i} = c_\mathrm{d} = 1$ (the so called *edit distance*) in the context of transmitting binary codes over unreliable channels that delete, insert and invert bits. There are generalisations of this algorithm that include transpositions of characters (a common source of typos) and general cost matrices over the set of symbols allowing to assign a smaller cost for easy-to-mix-up letters (for example, c and k or those adjacent on the keyboard).

The Levenshtein distance has particular significance for our difference chain codes from Equation 2.3. The individual symbols of a string have a semantic meaning in that the eight values

$0, 1, \ldots, 7$ represent quantised angles $0\,°, 45\,°, \ldots, 315\,°$. Here it makes sense to penalise exchanges depending on how severe the direction encoded in the symbol has changed:

$$c_\mathrm{x}(s_j, t_i) = \min((s_j - t_i) \mod 8, (t_i - s_j) \mod 8). \tag{2.7}$$

This cost of exchange is meant to compute the absolute difference of the angles reflecting the fact that 7 and 0 are neighbouring angles in this quantisation scheme of $45\,°$.

The *Hamming distance* between two strings of the same length that counts the number of positions at which letters differ can be seen as a trivial case of the Levenshtein distance for which $c_\mathrm{d} = c_\mathrm{i} = \infty$ and $c_\mathrm{x} = 1$.

While the edit distance is purely based on the syntactic form of the words involved, looking at co-occurrence at document level can yield a deeper insight into the semantic similarity of strings. This co-occurrence can be computed from a training set of documents or taken from other external knowledge sources such as the world wide web or the more structured sources that ontologies are. We start by formalising co-occurrence counts.

Let a be a document-word matrix, where each row represents a document in form of a sparse binary word vector. Then

$$b = a^t a$$

is a symmetric matrix, whose element b_{ij} contains the number of documents in which word i co-occurs with word j. a^t denotes the transposed matrix for which $(a^t)_{ij} = a_{ji}$. Each column $b_{i\bullet}$ in the co-occurrence matrix b describes how strongly the word i co-occurs with all the other words of the repository vocabulary. The value

$$d_{\mathrm{co-occ}}(i,j) = 1 - \frac{b_{ij}}{\max(b_{ii}, b_{jj})} \tag{2.8}$$

is just one possible dissimilarity measure for how far word j is from word i with respect to their joint usage in documents. It changes with the granularity of what is considered to be a document (a window of 10 words, a sentence, a paragraph, a section, a chapter, a book).

One of the biggest resources of word usage is the internet, of which web search engines have indexed several billion web pages as of today. Most search engines tell you in how many web pages a particular word i is mentioned (c_i), and equally in how many web pages two different words i and j both appear (c_{ij}). Gracia and Mena (2008) named the expression

$$d_\mathrm{nw}(i,j) = \frac{\max(\log c_i, \log c_j) - \log c_{ij}}{\log N - \min(\log c_i, \log c_j)}$$

the *normalised web distance* between words i and j; here N denotes the number of web pages that this particular web search engine has indexed. d_nw was defined earlier by Cilibrasi and Vitányi (2007), who meticulously justified it as an approximation to a normalised information distance. Despite its name d_nw violates strict positivity $d_\mathrm{nw}(i,j) > 0$ for $i \neq j$: imagine two different words i and j that appear in exactly the same set of web pages; then $c_i = c_j = c_{ij}$ and $d_\mathrm{nw}(i,j) = 0$ although $i \neq j$. Even worse, it turns out that d_nw violates the triangle inequality, too. As web search engines sometimes return counts that are slightly inconsistent, it can happen in practice that they report numbers with $c_i < c_{ij}$, which will make d_nw slightly negative. There may be the additional difficulty to get a reliable estimate for N.

Despite all its shortcomings, d_nw is very popular as it gives uncomplicated access to word co-occurrence estimations in a vast corpus. Luckily its definition is scale-invariant, ie, if the number

N of indexed web pages multiplies by a factor f, and with it the numbers c_i, c_j and c_{ij}, then the value for $d_{\text{nw}}(i,j)$ stays invariant. However, the world wide web corpus changes over time, and as new words are introduced, phased out or change meaning, their normalised web distances will change, too.

2.4 Feature and Distance Standardisation

If one uses different feature types to describe a multimedia object, then it is desirable that these features are comparable in size. For example, if you had encoded aspects of music in terms of pitch frequency in Hz[6] and beats-per-second, then typical feature vectors would look like $(10,000, 0.8)$. While a difference of 0.2 in the second feature type effects a significant change in perception, a difference of 20 in the first feature type is negligible.

While these differences can be absorbed in the distance functions that work on the features, it is much better to standardise these feature values from the outset. Feature standardisation has two different objectives: one is to make different components of a feature vector that have different origin and meaning comparable in size; the other is to ensure that the size of the feature vector is bounded, and hence distances between two feature vectors. The reason for the second objective is that distances between feature vectors are used for result list ranking; if multimedia objects are to be ranked with respect to different features, then it is desirable that their respective distances are in the same range (see fusion section 2.6).

2.4.1 Component-wise Standardisation using Corpus Statistics

For the first objective, features can be standardised component-wise using component mean values and their component-wise mean absolute deviation. Both are computed per component and across the data set. Formally, let v_j^j be the i-th component of the j-th feature vector in a set $F = \{v^1, v^2, \ldots, v^N \in \mathbb{R}^n\}$. This corresponds to the situation where N multimedia objects are indexed with N feature vectors $v^1, v^2, \ldots, v^N \in \mathbb{R}^n$, where each feature vector has n components. The data-set-mean $\overline{v}_i$ of the i-th component of the feature vectors is defined as

$$\overline{v}_i = \frac{1}{N}\sum_{j=1}^{N} v_i^j.$$

Using this value one can compute the mean absolute deviation per component i

$$\overline{s}_i = \frac{1}{N}\sum_{j=1}^{N} |v_i^j - \overline{v}_i|,$$

which amounts to the "typical variation" of the feature component i. Replacing all feature vector components with

$$\tilde{v}_i^j = \frac{v_i^j - \overline{v}_i^j}{s_i}$$

will create feature vectors $\tilde{v}^1, \tilde{v}^2, \ldots, \tilde{v}^N \in \mathbb{R}^n$, whose components have a typical size of 1 and have a mean of 0. This process creates feature vectors with negative components, which most distance measures are oblivious to. Note though, that some distance measures do not allow for negative components, for example, the squared chord distance or probabilistic distance measures. For these it may be better to forgo centring the feature vectors about zero, and

$$\tilde{v}_i^j = v_i^j / s_i$$

will be the far better transformation.

[6] this is only an example: there is something else wrong with using frequency as a feature value in the first place, which is discussed below

2.4.2 Range Standardisation

If the feature components i are known to lie in the range $[a_i, b_i] \ni v_i^j$ each, then the feature vectors can be transformed according to

$$\tilde{v}_i^j = \frac{v_i^j - a_i}{b_i - a_i} \in [0,1].$$

One can determine suitable values $a_i, b_i \in \mathbb{R}$ easily through

$$a_i = \min_{j=1}^{N} v_i^j \quad \text{and} \quad b_i = \max_{j=1}^{N} v_i^j.$$

This transformation is most useful for uniformly distributed features, but might suffer from the effects of outliers, which adversely move the minimum and maximum boundaries.

2.4.3 Ratio Features

Some feature components are not on a linear scale. For instance, frequency or loudness is perceived on a logarithmic scale. One characteristic of these components is that they are so-called ratio variables for which $1/(k+1)$ and $1/k$ should have the same distance as k and $k+1$. A commonly applied transformation is to work with the logarithm

$$\tilde{v}_i^j = \log(v_i^j)$$

of the variable instead. Ratio features are a special case of features with non-linear response. For example, one might consider using the square root of the file size of an image to approximate its linear size.

2.4.4 Vector Normalisation

Standardisation makes components of a feature vector comparable in range. This is particularly indicated for feature vectors that have different interpretations or even physical units for different components, but may not be necessary at all in other cases. Even if feature vector components are standardised, there is still the issue that some feature vectors will be typically large (ie, have a large L_p norm) and others will be small. A feature vector that has 100 components, each of which standardised across the data set, is expected to be $\sqrt{10}$ times larger under the Euclidean norm than a component-standardised feature vector with 10 components. Vector normalisation

$$\tilde{v}^j = v^j / \mathrm{L}_p(v^j) \tag{2.9}$$

ensures that all the feature vectors have the same L_p norm, which in turn limits the distance of any two such vectors.

An alternative to vector normalisation is distance normalisation, where each vector is divided by a constant c_F that is the average pairwise distance over the feature set F

$$c_F = \frac{1}{|F|^2} \sum_{v_i, v_j \in F} d(v^i, v^j).$$

It is often prohibitive to compute this number, and then a fixed random sample $F' \subset F$ instead of F will give a good estimate for c_F. In any case, scaling the vectors

$$\tilde{v}^j = v^j / c_F \tag{2.10}$$

yields distances of typical feature vector sizes of 1.

It is important to realise that Equation 2.10 *uniformly* scales all vectors by the same factor. If only this particular feature is used for ranking, then the overall ranking will not change. Vector scaling is useful to calibrate relative influence of feature vectors during a fusion of different features. In contrast to this, vector normalisation (2.9) has a *different* effect on different vectors and may well perturb the retrieval results under a particular feature vector, ie, the performance can get better or worse with vector normalisation.

2.5 High-dimensional Indexing

A significant bottleneck when searching any large database for nearest neighbours is the amount of data that needs to be loaded from disk. In traditional relational databases, B-tree or hashing are the predominant disk-based indexing mechanisms for single attributes (dimensions). These approaches are useful for database accesses where each dimension is used independently to select entries. For nearest-neighbour computation *all* dimensions of a feature vector contribute to its distance to the query's feature vector.

It turns out that indexing high-dimensional vectors efficiently is very challenging owing to the *curse of dimensionality*. This term was first used by Bellman (1961)[7] and refers to the phenomenon that our understanding of the space in 2 or 3 dimensions breaks down as the dimensionality of the space increases. Assume for a moment that our feature space is described by a n-dimensional unit hypercube $[0, 1]^n$ containing uniformly distributed data points. Given a query point, how much of the range of each dimension must we consider to capture a proportion p of the data? To enclose a fraction p of the unit volume, the length l in each dimension is $l = p^{1/n}$. If we are trying to enclose only 1% of the data in 10 dimensions, this means we must consider 63% of the range of each dimension, for 100 dimensions this increases to 95% and for 500 dimensions it is 99%. Conversely, 99% of the volume of a 500-dimensional unit hypercube is located in its surface skin of 0.005 thickness! In other words, if we lived in high dimensions we better not peel potatoes as little would be left to eat.

Beyer et al (1999) showed with similar arguments to those above that, as dimensionality increases, all points tend to exhibit the same distance from a query point. This has the ultimate effect of making the nearest neighbour problem ill defined.

Real-world data, such as image features, are unlikely to be uniformly distributed and may exist on a lower-dimensional manifold. This will alleviate some of the symptoms of the curse, however, significant effects for nearest neighbour searching and indexing remain.

[7] illustrating the fact that a Boolean function of n arguments has 2^n cases

It is likely that a real feature space may have an intrinsic dimensionality lower than the apparent data space. Dimensionality reduction methods aim to extract significant information into lower dimensions. Principal component analysis is a common technique, and it is often used in combination with other methods. PCA works well but has drawbacks for indexing. Its complexity can make it impractical for very large datasets with high dimensionality, and there are difficulties with incrementally adding data.

A significant class of methods partition the feature space or data points into tree structures. The first of these for multi-dimensional space was the R-tree developed by Guttman (1984). There have been many variants of this, and they have proved successful in certain circumstances. High-dimensional feature spaces are sparsely populated, and so it becomes hard to partition the data effectively. This is significant for tree-based structures. Indeed, Weber et al (1998) showed that all tree structures are less effective than a linear scan of all data above a certain dimensionality. This led them to develop a vector approximation technique called the VA-file. This accepts the fact that the linear scan is inevitable and attempts to optimise it using compression. They achieve times of 12.5–25% of a linear scan.

Most approaches store and search each dimension of the feature separately. This is often referred to as vertical decomposition, column store or decomposition storage model and gives a very flexible approach as dimensions can be treated differently depending on their significance. For instance, the inverted VA-file of Müller and Henrich (2004) stores each dimension at different quantisation levels and only retrieves at the accuracy needed dependent on the query. The BOND system developed by de Vries et al (2002) uses a branch-and-bound algorithm so that data in later dimensions can be discarded. Finally, Aggarwal and Yu's iGrid (2000) and bitmap indices (Wu et al, 2004a; Sinha and Winslett, 2007) work with vertically decomposed features and use only the part of each dimension close to the query point to generate a similarity value. In the same spirit, Howarth and Rüger (2005c) suggest a new local distance function for only the objects close the query point.

Approximate nearest neighbour approaches relax the constraint of finding exact results to speed up search. Nene and Nayar's method (1997) recovers the best neighbour if it is within ε of the query point. Beis and Lowe (1997) developed a variant of the k-d tree using a best-bin-first algorithm. They used this to efficiently retrieve the nearest or a very close neighbour in a shape indexing context. In a more recent development, Muja and Lowe (2009) published an approach that will take a given dataset and desired degree of precision and automatically determine the best algorithm and parameter values for that. They also describe a new algorithm that applies priority search on hierarchical k-means trees, which they have found to provide a good performance on many datasets. Marius Muja distributes public domain code of their software library called FLANN[8] (Fast Library for Approximate Nearest Neighbours), which implements their ideas.

2.6 Fusion of Feature Spaces and Query Results

2.6.1 Single Query Example with Multiple Features

In this subsection, we assume that there is a single query example q and that each multimedia document m gives rise to a number of low-level features $f_1(m)$, $f_2(m)$, ..., $f_k(m)$, each of which would typically be a vector describing some aspect such as colour, texture, shape, timbre etc.

[8] http://people.cs.ubc.ca/~mariusm/index.php/FLANN/FLANN

Combined Overall Distance

Most systems accumulate the distances of these features to the corresponding features of the query q in order to define an overall distance

$$D_w(m,q) = \sum_{i=1}^{r} w_i d_i(f_i(m), f_i(q)) \tag{2.11}$$

between multimedia documents m and a query q. Here $d_i(\cdot,\cdot)$ is a specific distance function between the vectors from the feature i, and $w_i \in \mathbb{R}$ is a weight for the importance of this feature.

Note that the overall distance $D_w(m,q)$ is the number that is used to rank the multimedia documents in the database, and that the ones with the smallest distance to the query are shown to the user as query result, see Figure 2.1. Note also that the overall distance and hence the returned results crucially depend on the weight vector $w = (w_1, \ldots, w_r)$. In most interfaces, the user can either set the weights explicitly as in the interface shown in Figure 2.21, or the system can change the weights implicitly if the user has given feedback on how well the returned documents fit their needs.

Convex Combinations or not?

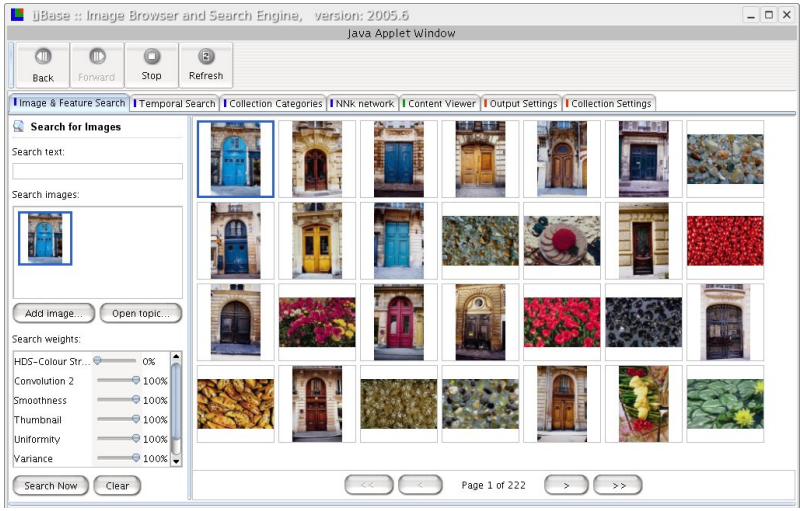
The weights w_i in Equation 2.11 are arbitrary real numbers, and so is the range of distance functions d_i. Often, however, the distances have been made to be in the same range, and the weights would be restricted to be in the unit interval and to sum to 1:

$$\sum_i w_i = 1 \quad \text{and} \quad 0 \le w_i \le 1$$

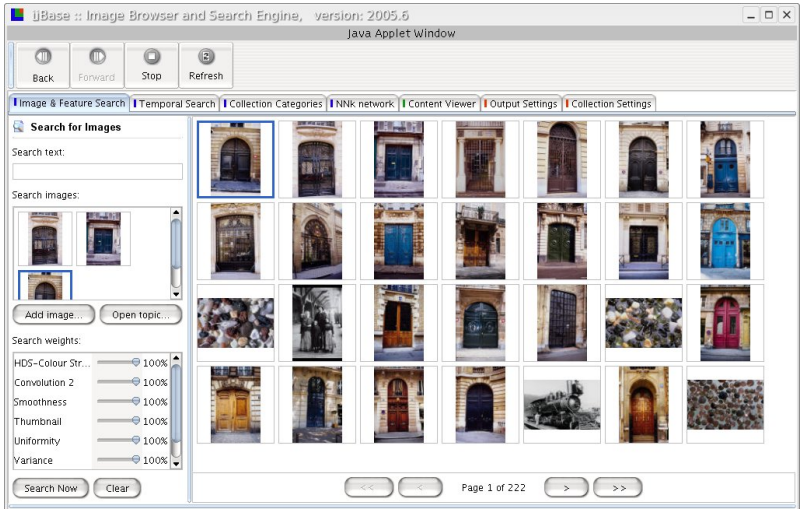
If the weights are restricted in this way then the sum in Equation 2.11 is called a *convex combination*. Some distances are already bounded owing to the nature of the underlying feature vectors (eg, the Manhattan distance between two normalised histograms is always in the range $[0,2]$) and others can be forced to be bounded through a process called feature standardisation (see Section 2.4). Convex combinations have the clear theoretical advantage that the ensuing overall distance $D_w(m,q)$ will then be bounded in the same way as the $d_i(\cdot,\cdot)$. This might be important if the query consists of multiple query examples, and the distances of these examples are to be fused later. The disadvantage of using convex weight combinations, however, is that they are less expressive. For example, users might want to specify a query by music example requiring that the rhythm and melody of the retrieved music piece should be the same as the query, only the timbre should not be like that at all. In this case, a negative weight for the timbre feature vector would be desirable.

Truncated Result Lists

Taken at face value, Equation 2.11 is incompatible with efficient nearest-neighbour computations in individual feature spaces: for one, the sets of nearest neighbours for the individual features are almost certainly different, and, on the other hand, some features may have a small weight or exhibit a small distance far down their nearest neighbour list for a particular query, so that a large proportion of this particular feature's index data are needed to determine a number, say k, of nearest neighbours with respect to the overall measure $D_w(m,q)$.



(a) Query by example (left panel) with initial results in the right panel



(b) A new query made of three images from (a) results in many more dark-door images

Figure 2.21: Visual search for images of dark doors starting with a bright-door example

One possible way out is to approximate $d_i(f_i(m), f_i(q))$, and hence $D_w(m, q)$, using truncated lists S_i^{gk} of, say, gk nearest neighbours for each individual feature i under $d_i(\cdot, f_i(q))$:

$$\tilde{d}_i(f_i(m), f_i(q)) = \begin{cases} d_i(f_i(m), f_i(q)) & \text{if } m \in S_i^{gk} \\ s_i & \text{otherwise} \end{cases}$$

Here $g \geq 1$ is a multiplier (say, $g = 10$), and s_i is a constant, for instance,

$$s_i = \max_{m \in S_i^{gk}} d_i(f_i(m), f_i(q)).$$

Basically s_i approximates the distance between q and any database object outside S_i^{gk} under feature i. It can be chosen arbitrarily larger to penalise any m that is not in the intersection of S_i^{gk} over i. Under this heuristic, the gk nearest neighbours for each feature i can be determined independently, possibly on different computers. Then there is no need to access the feature vectors of any m that is outside of the truncated lists S_i^{gk}.

CombSum, CombMin and CombMax

Rather than adding distances in Equation 2.11 one can also use the *minimum* of feature distances. Here the smallest distance determines the overall distance thus in effect requiring that *one* of the features be close for overall closeness. Alternatively, one can stipulate that the *maximum* of feature distances determines the overall distance, effectively requiring that *all* of the features be close for overall closeness. These combination mechanisms are also known as *CombSum*, *CombMin* and *CombMax* and can be extended to work with truncated lists of gk nearest neighbours.

Mc Donald and Smeaton (2005) compare these various fusion mechanisms on the TRECVid (see Subsection 3.2) data sets and consistently find that Equation 2.11 is best for combining multiple visual features over a single query. They also find that adding distances (scores) is best for combining a single visual feature over multiple queries.

Merging Individual Ranked Lists

Alternatively, rather than combining individual distances under various features, one can combine the ranked lists of nearest neighbours under the respective features. Rank-based methods largely ignore the feature-induced distances once they have been used to determine the order of multimedia objects. For example, by summing all the ranks that a multimedia object received under various features, one arrives at a new sorting criterion, which is called *Borda count*. It is possible to give preference to certain features using weights resulting in a *weighted Borda count*. Using the minimum of the ranks of a multimedia object with respect to different features corresponds to a *round-robin* method of merging ranks, basically interleaving ranked lists.

Learning Weights in Distance-score Space

Once relevance judgements are available one can ask the question what the best combination strategy is or even deploy machine learning algorithms to automatically determine weights or best fusion strategies (Bartell et al, 1994).

One way of using relevance judgements to determine an optimal weight vector utilises distances under different feature spaces: each judged object j (ie, a multimedia object where you know

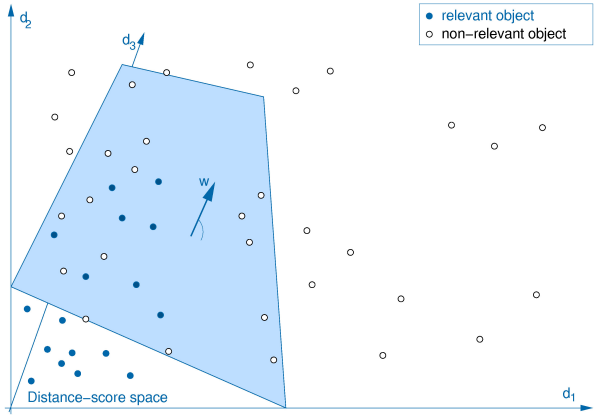


Figure 2.22: Separating relevant from non-relevant distance-score vectors in distance-score space

whether or not it is relevant to a particular query q), induces a vector $i \mapsto d_i(f_i(j), f(i)q)$ in a r-dimensional *distance-score space* (r is the number of different features). Placing a separating hyperplane between the positive distance-score vectors and the negative distance-score vectors that maximises the margin between these two groups immediately identifies the optimal weight vector for this query: it is a vector perpendicular to the optimal hyperplane. The idealised Figure 2.22 illustrates this property: in this example, clearly Feature 2 is most important to separate relevant from non-relevant vectors, as the separating hyperplane is nearly parallel to the $1 - 3$ plane; the weight vector's 2nd component w_2 is consequently the largest.

In practice, the two sets of vectors will not be clearly separable, as there are bound to be relevant multimedia objects far away from the query and non-relevant ones close by.

2.6.2 Multiple Query Examples

In the case of multiple query examples $q_1, q_2, \ldots, q_s$, it may either be that each example is of a different importance or even that some of the examples are negative examples with the meaning "but the query result should not be anything like these ones". Some of the mechanisms rely on the existence of both positive and negative examples. It is an ungrateful task for any user to come up with negative queries just for the sake of it, so some approaches randomly select examples from a large data set for this purpose assuming they will not be what the user wants.

Single Surrogate Query from Multiple Queries

If the examples are of the same kind (eg, all images or all music), then it is possible to average their respective feature vectors and thus reduce the case to a single surrogate query example. However, this makes the rather restrictive, and arguably unjustified, assumption that relevant database objects form a convex subset in feature space. Convex subsets have the property that averages, or more generally convex combinations, of its elements are still in the subset. Figure 2.23

illustrates why constructing a single surrogate query q from multiple examples (square dots) does not work in general.

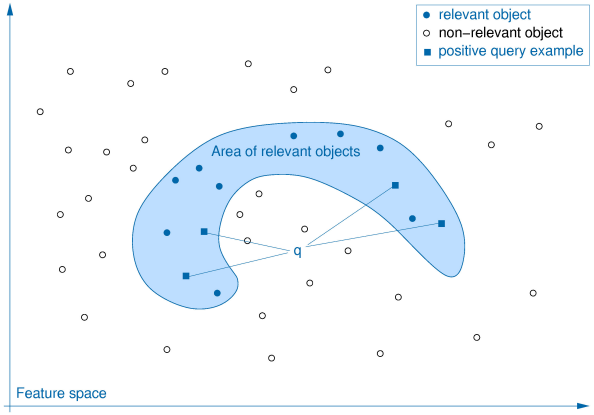


Figure 2.23: Constructing a single query q from multiple examples does not work

Combining Distances from Single Queries

In analogy to Equation 2.11, one can add up — and weight using numbers $u_j \in \mathbb{R}$ — distances brought about by the individual query examples $q_j \in Q$:

$$D^Q(m, Q) = \sum_j u_j D^j(m, q_j) \tag{2.12}$$

Equation 2.12, which I nickname the parent of all distances, computes the most versatile of all numbers that can be used to sort result lists from a set Q of query examples with positive or negative weights u_j. The distance functions D^j generally will follow Equation 2.11 but can be completely different for each individual query q_j, and involve different feature types, feature weights and distance functions. In analogy with the fusion of features, each $D^j(m, q_j)$ can be computed independently, even on different computers. In the interest of efficiency, the same tricks of using truncated result lists can be deployed here.

Merging Ranked Lists from Single Queries

In the same way as ranked lists from single features can be weighted and merged, this can be done with ranked lists that result from individual query examples, see Subsection 2.6.1 on page 60.

Memory-based Learning

In order to cater for different query examples, each one possibly exhibiting a different desired property, a distance averaging approach as in Equation 2.12 may be as undesirable as creating a single surrogate query as depicted in Figure 2.23.

In this case, intuitively, a better approach would be to reward multimedia objects that are close to any one of the positive query examples. This is what a *distance-weighted k-nearest neighbours approach* (Mitchell, 1997) does. Let Q be the set of query examples of which we know whether they are positive or negative. Q needs to have at least one positive example and one negative example, but we can always pepper Q with unjudged examples from a large set and treat these as negative.

For each of the multimedia objects m that are to be ranked, we compute the set Q^k of m's k nearest neighbours in Q and determine the subsets $P, N \subset Q^k$ of positive and negative examples, respectively. Naturally, we have $|P| + |Q| = k$. The number

$$R(m) = \frac{\sum_{p \in P} (D^p(m, p) + \varepsilon)^{-1}}{\sum_{n \in N} (D^n(m, n) + \varepsilon)^{-1} + \varepsilon},$$

where ε is a small positive number to avoid division by zero, determines how the multimedia objects should be sorted, given the query set Q.

Note that in this application, we compute nearest neighbour lists in the set of query examples (of an arbitrary m in the repository), while before we have always computed nearest neighbours in the full repository (of a query example). Although this way of combining evidence from a set Q of query examples has been shown to give better results than the combination of distances (Pickering and Rüger, 2003), it is rather resource-consuming. For large multimedia repositories, it may only be feasible to use the distance-weighted k-nearest neighbours approach for re-ranking candidate subsets rather than for ranking the full repository.

2.6.3 Order of Fusion

Rather a lot of evidence has gone into the similarity ranking block of the simple diagram in Figure 2.1: distances with respect to many different feature vector types and distances with respect to different query examples. Above subsections seem to be suggesting that one should fuse first wrt features and then wrt query examples. In fact, this could be the other way round or with a flat structure where all the evidence is combined at the same time. If one uses the parent of all distances, Equation 2.12, then it does not matter as nested finite sums commute:

$$\begin{aligned} D^Q(m, Q) &= \sum_j u_j \sum_i w_i d_i^j(f_i^j(m), f_i^j(q_j)) \\ &= \sum_i w_i \sum_j u_j d_i^j(f_i^j(m), f_i^j(q_j)) \\ &= \sum_{i,j} w_i u_j d_i^j(f_i^j(m), f_i^j(q_j)) \end{aligned}$$

If each step is approximated through truncated ranked lists — or merged, normalised, standardised, combined or machine-learned in a non-linear way otherwise — then the order of these processes will have an effect on the overall ranking.

2.7 Exercises

2.7.1 Colour Histograms

Colour is perceived as a point (r, g, b) in a *three-dimensional* space. Each colour component (red, green and blue) is usually encoded as an integer between 0 and 255 (1 byte); there are two principal methods to create colour histograms: you can compute three 1d histograms of each of the r, g and b components independently or you can compute one 3d colour histogram.

R	G	B	
0	0	0	black
255	0	0	red
0	255	0	green
0	0	255	blue
0	255	255	cyan
255	0	255	magenta
255	255	0	yellow
255	255	255	white

Figure 2.24: A striped colour test image

(a) If you divide each of the red, green and blue axes into n_r, n_g, n_b equal intervals, respectively, then the 3d colour cube $[0, 255]^3$ is subdivided into $n_r n_g n_b$ cuboids or bins. Show that by mapping (r, g, b) to

$$\left\lfloor \frac{n_r r}{256} \right\rfloor n_g n_b + \left\lfloor \frac{n_g g}{256} \right\rfloor n_b + \left\lfloor \frac{n_b b}{256} \right\rfloor$$

you get an enumeration scheme $0, \ldots, n_r n_g n_b - 1$ for the cuboids. A 3d colour histogram of an image is a *list of the numbers of the pixels in an image that fall into each of the different cuboids* in colour space. In other words, you look at each pixel in an image, compute above index from its colour (r, g, b) and increment the corresponding variable, which records the number of pixels that fall into this colour cuboid.

(b) Compute both types of colour histograms for the image in Figure 2.24 (the colours of the stripes are given by the table next to the image). Use $64 = 4^3$ bin cubes for the 3d-bin-histogram and 22 bins for each of the three colour-component histograms (yielding 66 bins altogether), so that both histogram types have a similar number of bins.

(c) Using the same visualisation method for 3d colour histograms as demonstrated in Section 2.2 yields Figure 2.25. Why are the colours in the visualisation different from the original colours of the image?

Visualise the three 1d histograms of the r, g and b components that you computed in (b). Given these histograms which colours might have appeared in the original image?

Which of the two colour histogram methods has retained more information about the colour distribution in the original picture? How did it come about that one of the two methods lost vital information despite having roughly the same number of bins (64 vs 66)?

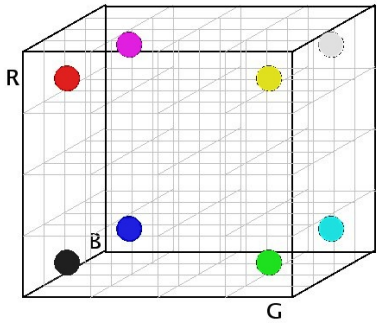


Figure 2.25: 3d colour histogram for the striped colour test image

2.7.2 HSV Colour Space Quantisation

The HSV colour space is popular because it separates the pure colour aspects from brightness. Figure 2.26 visualises the aspects of hue (H) that corresponds to a spectral frequency, saturation (S) that expresses how pure the colour is, and value (V) that expresses the apparent brightness of a colour. The hue values H are arranged on a circle and encoded from 0° to 360°, which is again the same as 0°. Zero saturation S means that the colour is grey and the higher the saturation is the less grey is mixed into it up to a value of 100% where the colour is deemed to represent a single spectral frequency, ie, a rainbow colour. H and S are polar coordinates where H is an angle and S is the radius. The pair (H, S) describes the chromaticity of a colour, while its apparent brightness V is independent from chromaticity. One other advantage of HSV over RGB is that it appears more intuitive to talk about colours in terms of their brightness and spectral names rather than the mixture coefficients of R, G and B.

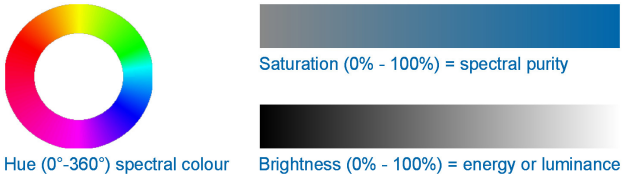


Figure 2.26: HSV coordinates

Show that all possible (H, S, V) values make up a cylinder and that the grey values reside on the central axis of the cylinder. Show that the conversion between RGB and HSV cannot be continuous.

One natural way of subdividing the HSV space into bins is to subdivide each coordinate linearly. Which of the so created *different* HSV bins will contain colour representations that are almost certainly in one single bin in RGB space (this is the effect of above discontinuity)? Which bins in HSV space do you need to merge, so that this discontinuity is no longer visible?

How would you need to subdivide HSV space so that its bins have the same volume as the set

of points in RGB space that are mapped into them?

2.7.3 CIE LUV Colour Space Quantisation

CIE[9] LUV (International Commission on Illumination, 1986) describes the physical appearance of colour, independent of the reproduction device, using the idea of a human standard observer. This colour space strives to be perceptually uniform in the sense that the Euclidean distance between any two points in colour space should resemble the human perception of the grade of colour difference. As a consequence, the transformation from RGB to CIE LUV is highly non-linear and the resulting colour space is not of a simple geometric form: Figure 2.27 depicts the chromaticity plane of CIE LUV. How should the CIE LUV colour space be quantised?

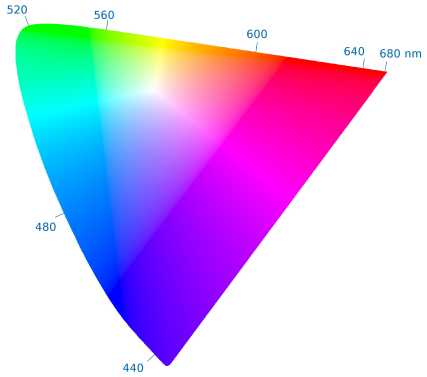


Figure 2.27: Chromaticity plane of the CIE LUV colour space

2.7.4 Skewness and Kurtosis

The third and fourth central moments are used to define skewness and kurtosis: skewness is defined as

$$s = \overline{p}_3 / \sqrt{\overline{p}_2}^3,$$

while kurtosis is defined as

$$k = \overline{p}_4 / \overline{p}_2^2 - 3.$$

Verify that subtracting 3 in the definition above makes the kurtosis of any normal distribution zero irrespective of its parameters. Show that the above definitions of skewness and kurtosis are dimensionless (ie, if the underlying quantity $p(i, j)$ had a physical dimension then s and k do not) and scale invariant (ie, if the underlying quantity was multiplied by a constant c then the values of s and c are not affected by this). Think of a way to define general standardised central moments that are dimensionless and scale invariant and compare with a textbook definition of standardised central moments (the textbook definition of the second standardised central moment is always 1).

[9] http://cie.co.at — Commission internationale de l'éclairage

Note that Equation 2.2 is neither dimensionless nor scale invariant — instead its aim is to generate values that are on the same scale.

2.7.5 Boundaries for Tamura Features

Determine the exact areas of an image in which coarseness, contrast and directionality can be determined, see Figure 2.6.

2.7.6 Distances and Dissimilarities

Show that the Bray-Curtis dissimilarity and the squared chord dissimilarity violate the triangle inequality, while the Canberra distance is a true distance, ie, that $d(v, w) = d(w, v)$, $d(v, w) = 0$ if and only if $v = w$ and $d(x, z) \leq d(x, y) + d(y, z)$.

Show that the Manhattan distance and the partial histogram intersection are equivalent, ie, lead to the same ordering if the feature vectors are normalised with respect to the L_1 norm and non-negative.

Show that the Euclidean distance and the cosine dissimilarity are equivalent, ie, lead to the same ordering if the feature vectors are normalised with respect to the L_2 norm and non-negative.

Draw the unit balls B_p in $\mathbb{R}^2$ with respect to the Minkowski norm L_p for $p \in \{0.5, 1, 2, \infty\}$. The formal definition is $B_p = \{x \in \mathbb{R}^2 | \, L_p(x) \leq 1\}$. Show that $B_{0.5}$ is *not* convex, ie, points on a straight line between 2 points within $B_{0.5}$ are not necessarily in $B_{0.5}$. Show that, as a consequence, $L_{0.5}$ violates the triangle inequality, ie, it does not induce a distance. Show that L_p induces a distance for $p \geq 1$.

2.7.7 Ordinal Distances — Pen-pal Matching

The table below shows a number of people's favourite writer, travel destination, colour, newspaper, city, genre and means of communication. Work out who would make the best pair of pen-pals. Which pair would be least compatible?

Ally	V Woolf	UK	red	Guardian	London	crime	twitter
Bert	L Carroll	Canada	green	Times	Tokyo	poetry	sms
Cris	L Carroll	Italy	green	Times	NY	drama	sms
Doro	V Woolf	Spain	blue	Mirror	London	fiction	e-mail
Enzo	V Woolf	Canada	green	Observer	Tokyo	drama	sms
Fred	M Proust	Spain	green	Mirror	London	crime	twitter

2.7.8 Asymmetric Binary Features

The table below shows a number of people's gender and character traits. Each of the traits occurs rather infrequently in the population (ie, they are *asymmetric* binary features: P = present, N = not present). Ignoring the gender, work out who would make the best pair of pen-pals. Which pair would be least compatible?

Name	gender	character					
Ally	F	N	P	N	N	N	N
Bert	M	N	P	P	N	N	N
Cris	F	P	P	P	N	P	P
Doro	F	P	N	N	P	N	N
Enzo	M	P	P	N	P	P	P
Fred	M	P	N	P	P	P	P

2.7.9 Jaccard Distance

Show that definition (2.3.5) for the Jaccard distance is identical to the definition (1.3) for resemblance when applied to sparse vector encodings of bags of words.

Write the matching coefficient (2.3.5) in terms of c_{xy} as in (1.4) and discuss the difference to the Jaccard distance.

2.7.10 Levenshtein Distance

It is easy to see that the general Levenshtein distance violates the symmetry requirement $d_{\mathrm{L}}(s,t) = d_{\mathrm{L}}(t,s)$ if the cost of deletion differs from the cost of insertion. However, prove that the edit distance is a proper metric, where the cost of deletion, the cost of insertion and the cost of exchanging different symbols are all 1.

2.7.11 Co-occurrence Dissimilarity

Compute the co-occurrence for the document word matrix of the "Humpty Dumpty" nursery rhyme on page 25. In this setting, what are the distances between *humpty* and *dumpty*, and between *king* and *horse*? What would change if Equation 2.8 used min instead of max in the denominator?

2.7.12 Chain Codes and Edit Distance

Encode the pixel boundary of a 4×4 square and of a 3×5 rectangle as difference chain codes and compute their edit distance.

2.7.13 Time Warping Distance

Modify the Levenshtein distance for difference chain codes such that local stretching can happen at no cost. Your solution should be scale invariant, for instance,

$$d_{\mathrm{Ldcc}}(123, 111223333) = 0.$$

Figure 2.28 gives you an idea how this distance could be processed in the case of the two strings 000710 and 01700000020, but your solution may well look different.

2.7.14 Feature Standardisation

Assume you have six feature vectors of the kind (age [year], height [m]) as follows: $v^1 = (10, 1.30)$, $v^2 = (20, 1.70)$, $v^3 = (30, 1.60)$, $v^4 = (40, 1.80)$, $v^5 = (50, 1.70)$ and $v^6 = (60, 9.90)$. Note that the last vector contains a deliberate outlier.

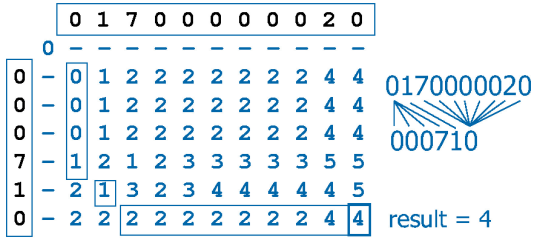


Figure 2.28: Time warping distance between curves represented by difference chain codes

Compute standardised feature vectors according to Subsection 2.4.1.

s_i is a "typical range" of the component i; sometimes the *empirical standard deviation*

$$\sigma_i = \sqrt{\frac{1}{N-1}\sum_j (v_i^j - \overline{v}_i)^2}$$

is used instead of s_i. In our case, $\sigma_1 = 18.7$ and $\sigma_2 = 3.38$ (rounded); argue why the standard deviation is more susceptible to the outlier of 9.90m height than the mean absolute deviation.

One could also use the *median* absolute deviation (defined as the middle-ranked value of all the absolute deviations sorted by size) instead of the mean absolute deviation. How would this affect the standardisation process in the presence of outliers. How about using the median instead of the mean (component-wise)?

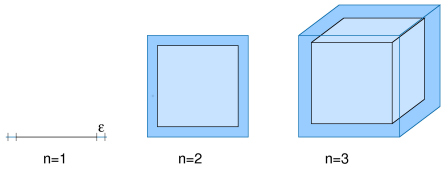


Figure 2.29: Skin of a cube in different dimensions

2.7.15 Curse of Dimensionality

Consider an n-dimensional feature space with a query vector at $q = 0$ and a hypercube $H_n = [-1, 1]^n$ of volume 2^n around it. Assume the hypercube contains uniformly distributed vectors, ie, each component is independently and identically distributed with a random uniform distribution in the range of $[-1, 1]$. The skin S_n^ε of H_n with thickness ε is defined as

$$S_n^\varepsilon = H_n \setminus [-1 + \varepsilon, 1 - \varepsilon]^n,$$

ie, it contains all points $x \in H_n$ that have at least one component x_i close to the boundary of the interval $[-1, 1]$, see Figure 2.29.

What is the probability that a randomly chosen vector of H_n lies in the skin S_n^ε? Plot this probability as a function of n for $\varepsilon = 0.01$. Show that the maximum distance (induced by the maximum norm L_∞, see Subsection 2.3.1) of q to any point s in the skin S_n^ε is bounded by

$$1 - \varepsilon \leq d_\infty(q, s) \leq 1,$$

ie, nearly every point in H_n has a distance to q of nearly 1.

2.7.16 Image Search

Sketch the block diagram of a colour-and-texture-based image search engine for curtain fabrics. Explain the general workings of a content-based search engine and contrast it with the workings of a text search engine in terms of retrieval and indexing technology.

Chapter 3

Evaluation campaigns in multimedia retrieval

This chapter presents an academic and research perspective on the impact and importance of ImageCLEF and similar evaluation workshops in multimedia information retrieval (MIR). Three main themes are examined: the position of ImageCLEF compared with other evaluation conferences; general views on the usefulness of evaluation conferences and possible alternatives and the impact and real-world meaning of evaluation metrics used within ImageCLEF. We examine the value of ImageCLEF, and related evaluation conferences, for the multimedia IR researcher as providing not only a forum for assessing and comparing outcomes but also serving to promote research aims, provide practical guidance (e.g., standard datasets) and inspire research directions.

3.1 Introduction

This chapter is not an exhaustive review of the impact of ImageCLEF and specific outcomes from ImageCLEF upon research. Rather it gives our multimedia information retrieval (MIR) group's perspective on the importance and usefulness of ImageCLEF in the academic context based on our experience participating in ImageCLEF and similar evaluation conferences and our view of MIR. In this section we will outline our experiences participating in ImageCLEF, define key approaches to IR evaluation and present the aims/needs of MIR research.

Our Multimedia Information Systems (MMIS) group at the Knowledge Media Institute (KMi) conducts research in the area of multimedia information retrieval including content-based search, automatic image annotation and video shot-boundary detection. MMIS and previously the Multimedia Group at Imperial College London, also led by Stefan Rüger, have participated in both TRECVid and ImageCLEF tasks since 2002 (Pickering and Rüger, 2002; Heesch et al, 2003, 2004; Howarth et al, 2005; Jesus et al, 2005; Magalhães et al, 2006; Overell et al, 2006, 2008; Llorente et al, 2008, 2009; Zagorac et al, 2009).

MMIS participation is generally a team effort with 2 or more members of the group working together to apply our latest research to the specific tasks for the evaluation campaign. Submission is often a time-consuming task principally due to the need to adapt different input/output formats to work with existing tools and match the required submission format. For example, in 2009 (Zagorac et al, 2009) we used technology developed for the PHAROS project (Bozzon et al, 2009) that used the MPEG-7/MPQF formats for input/output of the annotation and search tools. Therefore we

needed to convert the given media and queries into this format before processing and then convert the output to the required submission style. Processing time for the large volumes of media data also needs to be planned for. Participation is easier when an experienced team member, who has previously submitted runs to TRECVid or ImageCLEF, is available to help.

Section 2 describes a number of different evaluation conferences that serve a similar purpose to ImageCLEF but focus on different user tasks or different media types. These evaluations generally conduct performance assessment following what has been termed the "Cranfield paradigm" (Brookes, 1981; van Rijsbergen, 1989) based on tests performed at Cranfield in the 1960s (Cleverdon et al, 1966). With some variation in the order and which party performs each step, the general process is: a document collection is assembled, a set of test queries is developed, each system performs the queries and its output is evaluated using a reserved test set, where each document is assigned a relevance judgement. William Webber has written an extensive blog post[1] discussing how the approach used in the Cranfield tests came to be known as the Cranfield paradigm. Section 3 discusses some different viewpoints on the utility of Cranfield based evaluations for driving information retrieval research.

Evaluations based on the Cranfield approach are known as *system-based* or "batch" evaluations that compare information retrieval systems primarily on their ability to identify and properly rank documents deemed to be relevant. These evaluations use one or more specific, generally numerical, metrics ranging from straight-forward precision and recall calculations to more complex and comprehensive rank-based metrics such as mean average precision, precision at n and cumulative gain (Järvelin and Kekäläinen, 2002) that weight values in favour of returning the most relevant documents first. In contrast, *user-based* evaluations focus on the performance of the system from a user perspective in fulfilling an information need. These evaluations generally involve direct testing of a system implementation by a user in a situation designed to reflect the real-world. Voorhees (2002) defines the difference as "user-based evaluation measures the user's satisfaction with the system, while system evaluation focuses on how well the system can rank documents".

User-oriented evaluation for multimedia often needs to consider extra facets to traditional text search. Cooniss et al (2000, 2003) carried out two widely noted studies of the needs, characteristics, and actions of end users of visual information in the workplace (called VISOR 1 and 2). One of the useful distinctions in their reports is the one between searching for oneself and searching as an intermediary, eg, for a journalist on the other side of the phone. Smeulders et al (2000) identified three types of search, labelled 'target search', aiming at a specific multimedia object identified by title or other metadata; 'category search', where the user has no specific object in mind but can describe which features they would like; and 'search by association' where the user is less specific and happy to browse in order to retrieve multimedia by serendipity.

What are the evaluation needs of a multimedia information retrieval research group? First is to drive research direction through the exchange of cutting-edge ideas and the establishment of realistic test sets and basic performance benchmarks. Second is to push system performance through open, consistent and comparable evaluation processes that enable clear discussion the strengths, weaknesses and similarities of approaches. Finally, to perform holistic, real-world evaluations of the ability of the system to address user's information needs. The remainder of this chapter will outline the main evaluation venues for MIR, discuss the utility of system-based evaluation, focus on the use of metrics to summarise system performance based on relevance judgements and, finally, look at the future requirements for evaluation of multimedia information retrieval systems.

[1] William Webber, 'When did the Cranfield tests become the "Cranfield paradigm"?' http://blog.codalism.com/?p=817 (accessed 13th May 2010)

3.2 ImageCLEF in multimedia IR

Since its early conception information retrieval as a subject has always placed great emphasis on system evaluation (Rüger, 2010). Real user needs are simulated in a laboratory setting with three ingredients: large test collections, information need statements and relevance judgements. The test collection contains a large number of potentially interesting documents from a repository; each information need statement details the type of document that the user would like to retrieve, what the user hopes to see or hear and criteria for how the relevance of documents should be judged. The relevance judgements, also known as ground truth, tell us whether a particular document of the collection is relevant for a particular information need. The value of an evaluation setting like this is that the effectiveness of a particular retrieval method can be measured in a reproducible way. Although this approach has been criticised for its lack of realism and its narrow focus on the pure retrieval aspect of presumably much bigger real tasks, system evaluations are still the main basis on which retrieval algorithms are judged, and on the back of which research flourishes. In this respect evaluation conferences such as INEX for structured XML retrieval, ImageCLEF for image retrieval, MIREX for music retrieval, TRECVid for video retrieval and GeoCLEF for geographic retrieval have a significant and lasting impact on multimedia information retrieval research through reproducibility and comparisons. ImageCLEF is discussed extensively in this book. Here we give a brief summary of INEX, MIREX, TRECVid, GeoCLEF and other evaluation campaigns and compares their structure and aims with ImageCLEF. TRECVid, in particular, is extensively described as its purpose and aims align most closely with those of ImageCLEF.

INEX XML Multimedia Track

In 2002, the INEX[2] Initiative for the Evaluation of XML Retrieval started to provide a test-bed for evaluation of effective access to structured XML content. The organisation of INEX passed from the University of Duisburg to Otago University[3] in the year 2008.

Van Zwol et al (2005) set up an XML multimedia track that was repeated as part of INEX until 2007. It provided a pilot evaluation platform for structured document retrieval systems that combine multiple media types. While in 2005 the collection was made up from Lonely Planet travel guides, the 2006 evaluations used the much larger Wikipedia collection from the INEX main track (Westerveld and van Zwol, 2006). Both collections contain a range of media, including text, image speech, and video — thus modelling real life structured documents. The goal of the multimedia track was to investigate multimedia retrieval from a new perspective, using the structure of documents as the semantic and logical backbone for the retrieval of multimedia fragments.

In contrast to other evaluation fora, INEX's multimedia track was to retrieve relevant document fragments based on an information need with a structured multimedia character, ie, it focused on the use of document structure to estimate, relate, and combine the relevance of different multimedia fragments. One big challenge for a structured document retrieval system is to combine the relevance of the different media types and XML elements into a single meaningful ranking that can be presented to the user.

[2] http://inex.is.informatik.uni-duisburg.de
[3] http://www.inex.otago.ac.nz

MIREX

The MIREX[4] (Music Information Retrieval Evaluation eXchange) is a TREC-style evaluation effort organised by the International Music Information Retrieval Systems Evaluation Laboratory (IMIRSEL[5]) at the Graduate School of Library and Information Science[6], of the University of Illinois at Urbana-Champaign. It is a community-based evaluation conference for Music Information Information Retrieval systems and algorithms. Downie (2008) looks at the background, structure, challenges, and contributions of MIREX and provides some insights into the state-of-the-art in Music Information Retrieval research as a whole.

GeoCLEF

The GeoCLEF[7] track was introduced to the CLEF workshop in 2005 as an ad-hoc TREC style evaluation for geographic Information Retrieval systems; this provided a uniform evaluation for the growing GIR community and is becoming the de facto standard for evaluating GIR systems. GeoCLEF has moved its home to the University of Hildesheim[8].

The GeoCLEF 2005-08 English corpus consists of approximately 135,000 news articles, taken from the 1995 Glasgow Herald and the 1994 Los Angeles Times; the overall corpus also includes German, Spanish and Portuguese documents. There are 100 GeoCLEF queries from 2005–08 (25 from each year). These topics are generated by hand by the four organising groups. Each query is provided with a title, description and narrative. The title and description contain brief details of the query, while the narrative contains a more detailed description including relevance criteria. The 2005 queries have additional fields for concept, spatial relation and location. However, these fields were discarded in later years as unrealistic. Typical topics of GeoCLEF include *Shark Attacks off Australia and California* (Topic 001) or the rather more difficult *Wine regions around rivers in Europe* (Topic 026). Mandl et al (2008) present an overview of GeoCLEF 2007.

TRECVid

The TREC Video Retrieval Evaluation initiative (TRECVid[9]) is an independent evaluation forum devoted to research in automatic segmentation, indexing, and content-based retrieval of digital video. It started out in 2001 as a video track of the TREC[10] conference series and became an independent 2-day workshop of is own in 2003. TRECVid is sponsored by NIST[11] (National Institute of Standards and Technology) with additional support from other US government agencies. Participation in TRECVid has been rising since its early days, and in 2007 54 teams from all over the world took part. Smeaton et al (2006) give an overview of the TREC Video Retrieval Evaluation initiative.

The information need of an example topic for the 2003 TRECVid Interactive Track is described as "Find shots from behind the pitcher in a baseball game as he throws a ball that the batter swings at". Other search topics may be exemplified by short video clips or a combination of video clips

[4] http://www.music-ir.org/mirex
[5] http://music-ir.org/evaluation
[6] http://www.lis.uiuc.edu
[7] http://ir.shef.ac.uk/geoclef
[8] http://www.uni-hildesheim.de/geoclef
[9] http://trecvid.nist.gov
[10] http://trec.nist.gov
[11] http://www.nist.gov

and images. The 2003 TRECVid test collection repository consists of video shots from mainly US news programmes.

Every year 25 topics are released to all participating groups, who would have pre-processed and indexed the test collection prior to this. The rules for the *interactive task* of the search track allow searchers to spend 15 minutes per topic to find as many relevant shots as possible; they are free to create a search engine query from the given topic in any way they see fit, modify their query, and collect shots that they deem relevant. Each participating group returns the results of their searches to NIST, who are then responsible for assessing the returned shots from all the participating groups. The assessors, often retired intelligence workers, would look at a pool of results for each topic and assess the relevance of each shot in the pool for a topic. In order to be resourceful with the assessors' time, only the union of the top n, say 100, of all the results from different groups for a particular topic is put into this pool. The explicitly assessed shots for each topic form the relevance judgements. Shots that were not assessed during this procedure are those that none of the many participating systems reported in their respective top n results, and the implicit assumption is that these unassessed shots are *not* relevant. The reason for this is the prohibitive cost of assessing all shots against all topics.

This ground truth is then the basis on which participating retrieval systems can be compared. It is possible to use this setting for later evaluation outside the TRECVid programme: the only slight disadvantage is that the assessed algorithm would not have contributed to the pooling process; hence, if the new algorithm uncovered many relevant shots that no other algorithm of the participating groups has reported in their top n results, then these would be treated as irrelevant.

The interactive task is only one task amongst many. There are *manual tasks* where the searchers are allowed to formulate and submit a query *once* for each topics without further modification; there is an *automated task* where the generation of the computer query from a search topic is fully automated without any human intervention. These three tasks form the *search track* of the TRECVid evaluation conference, which is one of typically three to five tracks each year. Over the years other tracks have included:

Shot segmentation, ie, the sectioning of a video into units that result from a single operation of the camera, is a basic but essential task that any video processing unit has to carry out. Hard cuts, where adjacent shots are basically edited by simply concatenating the shots, are relatively easy to detect as the frames of a video change abruptly. Modern editing techniques deploy gradual transmissions, though, eg, fade out/in, which provide continuity between shots and thus are harder to detect. Shot segmentation algorithms vary widely in their efficiency, ie, how much faster (or slower) they are than playing the video. Generally, algorithms that need to decode the video stream into frames tend to be slower than algorithms that operate on the compressed video format.

The *story segmentation* track meant to identify the (news) story boundaries with their time. A news story is defined as a segment of news broadcast with a coherent focus. While a story can be composed of multiple shots (eg, an anchorperson introduces a reporter, who interviews someone in the field and uses archive material to explain background), a single shot can contain story boundaries, e.g. an anchorperson switching to the next news topic. Although this track is non-trivial, it has only been part of TRECVid for a couple of years.

In 2007 TRECVid introduced new video genres taken from a real archive in addition to its previous focus on news: news magazine, science news, news reports, documentaries, educational programming and archival video. The idea was to see how well the video retrieval and processing technologies apply to new sorts of data.

In addition to that, the BBC Archive has provided about 100 hours of unedited material (also

known as *rushes*) from five dramatic series to support an exploratory track of *rushes summarisation*: systems should construct a very short video clip that includes the major objects and events of the original video. A dedicated workshop at ACM Multimedia Over and Smeaton (2007) presented the results of these efforts.

The *Surveillance event detection* track is a more recent addition to TRECVid that operates on around 150 hours of UK Home Office surveillance data at London Gatwick International Airport.

The *Content-based copy detection* track tries to identify modified segments of a video under a variety of transformations such as a change of aspect ratio, colour, contrast, encoding, bit rate, addition of material, deletion of material, picture in picture in the video part or bandwidth limitation and variate mixing with other audio content in the audio part. Real world applications would be copyright control, de-duplication in large data repositories, grouping of video results in large video repositories or advertisement tracking.

Feature extraction tracks have played an important role throughout the lifetime of TRECVid. Many requests for archival video contain requests for specific features (see above discussion in this section). One of the frequently required aspects is that of a specific camera motion. In the low-level feature extraction version, camera motions such as *pan (left or right)* or *tilt (up or down)* had to be detected. Generally, owing to the semantic gap, high level feature extraction tasks are more difficult. They concern semantic concepts such as *indoor*, *outdoor*, *people*, *face*, *text overlay*, *speech* etc. These concepts can be very useful additional search criteria to home in on many real-world requests. Smeaton et al (2009b) have summarised the work done on the TRECVid high-level feature task and show the progress made across the spectrum of various approaches.

VideOlympics

The VideOlympics[12] (Snoek et al, 2008), held most recently at CIVR 2009, is not an evaluation campaign in the traditional sense but rather an opportunity for researchers with video retrieval systems to demonstrate their work through real-time user evaluations in a demo session format. It uses data from TRECVid and is therefore aimed principally at TRECVid participants. It is not intended to produce comparative results for publication but to inform the audience about the state-of-the-art in video retrieval and promote discussion about user interfaces for video search.

PASCAL Visual Object Classes (VOC) challenge

In 2005, the PASCAL Visual Object Classes challenge[13] appeared supported by the EU-funded PASCAL2 Network of Excellence on Pattern Analysis, Statistical Modelling and Computational Learning[14]. The goal of this challenge is to recognise objects from a number of visual object classes in realistic scenes. It is fundamentally a supervised learning learning problem in that a training set of labelled images is provided. The twenty object classes that were selected belonged to the following categories: person, animal, vehicle, and objects typically found in an indoor scene. The challenge is divided into three main tasks: the classification task, which predicts the presence or absence of an instance of the class in the test image; the detection task, which determines the bounding box and label of each object in the test image; segmentation task, which generates pixel-wise segmentations giving the class of the object visible at each pixel.

[12] http://www.videolympics.org
[13] http://pascallin.ecs.soton.ac.uk/challenges/VOC/
[14] http://www.pascal-network.org/

The workshop where participants are invited to show their results is co-located with a relevant conference in computer vision such as the International Conference on Computer Vision or the European Conference on Computer Vision. The 2010 edition added new tasks such as still image action classification and large scale visual recognition.

MediaEval and VideoCLEF

MediaEval[15] is a benchmarking initiative launched by the PetaMedia Network of Excellence in late 2009 to serve as an umbrella organization to run multimedia benchmarking evaluations. It is a continuation and extension of VideoCLEF, which ran as a track in the CLEF campaign in 2008 and 2009.

The initiative is divided into several tasks. In the 2010 edition, there are two annotation tasks called the tagging task but designed with two variations, the professional version and the wild wild web version. The professional version tagging task requires participants to assign semantic theme labels from a fixed list of subject labels to videos. The task uses the TRECVid data collection from the Netherlands Institute for Sound and Vision. However, the tagging task is completely different than the original TRECVid task since the relevance of the tags to the videos is not necessarily dependent on what is depicted in the visual channel. The wild wild web version task requires participants to automatically assign tags to videos using features derived from speech, audio, visual content or associated textual or social information. Participants can chose which features they wish to use and are not obliged to use all features. The dataset provided is a collection of Internet videos.

Additional tasks for the 2010 initiative are the placing task or geo- tagging where participants are required to automatically guess the location of the video by assigning geo-coordinates (latitude and longitude) to videos using one or more of: video metadata such as tags or titles, visual content, audio content, social information. Any use of open resources, such as gazetteers, or geo-tagged articles in Wikipedia is encouraged. The goal of the task is to come as close to possible to the geo-coordinates of the videos as provided by users or their GPS devices. Other tasks are the affect task whose main goal is to detect videos with high and low levels of dramatic tension; the passage task where, given a set of queries and a video collection, participants are required to automatically identify relevant jump-in points into the video based on the combination of modalities such as audio, speech, visual, or metadata; and the linking task, where participants are asked to link the multimedia anchor of a video to a relevant article from the English language Wikipedia.

One of the strongest points of this competition is that they attempt to complement rather than duplicate the tasks assigned to in the TRECVid evaluation campaign. Traditionally, TRECVid tasks are mainly focused on finding objects and entities depicted in the visual channel whereas MediaEval concentrates on what a video is about as a whole.

Past Benchmarking Evaluation Campaigns

This section summarises other relevant benchmarking evaluation campaigns that have previously been operative in this area. They are worth mentioning as their research questions, objectives, results, and the used datasets persist online.

The *Face Recognition Vendor Test* (FRVT)[16] 2006 was the latest in a series of large scale independent evaluations for face recognition systems organised by the U.S. National Institute of

[15] http://www.multimediaeval.org
[16] http://www.frvt.org/

Standards and Technology. Previous evaluations in the series were the FERET, FRVT 2000, and FRVT 2002. The primary goal of the FRVT 2006 was to measure progress of prototype systems and commercial face recognition systems since FRVT 2002. Additionally, FRVT 2006 evaluated the performance on high resolution still imagery, 3D facial scans, multi-sample still facial imagery, and re-processing algorithms that compensate for pose and illumination.

A short-lived evaluation campaign that only ran for one year in 2006, *ImagEVAL*[17] was a French initiative that tried to bring some answers to the question posed by Carol Peters, in the CLEF workshop in 2005, where she wondered why systems that show very good results in the CLEF campaigns have not achieved commercial success. The point of view of ImagEVAL was that the evaluation criteria "do not reflect the real use of the systems". Thus, this initiative was launched in France in 2006, fairly concentrated on the French research domain, although it was accessible to other researchers as well. The campaign was divided into several tasks relating to image analysis including object detection, querying, text detection and recognising transformed images. The focus of this evaluation campaign was certainly closer to the user-oriented perspective and it hoped to improve methods of technological evaluation so that end-user criteria could also be included.

During the 2000 Internet Imaging Conference, a suggestion was made to hold a public contest to assess the merits of various image retrieval algorithms. Since the contest would require a uniform treatment of image retrieval systems, the concept of a benchmark quickly entered into the scenario. This contest became known as the *Benchathlon*[18] and was held at the Internet Imaging Conference in January 2001. Despite their initial objectives no real evaluation ever took place although many papers were published in this context and a reference database created.

The *Classification of Events, Events, Activities and Relationships* (CLEAR)[19] evaluation conference was an international effort to evaluate systems that are designed to recognise events, activities, and their relationships in interaction scenarios. Its main goal was to bring together projects and researchers working on related technologies in order to establish a common international evaluation in this field. It was divided into the following tasks: person tracking (2D and 3D, audio-only, video-only, multimodal); face tracking; vehicle tracking; person identification (audio-only, video-only, multimodal); head pose estimation (2D and 3D); and acoustic event detection and classification. The latest edition, held in 2007, was supported by the European Integrated project "Computers In the Human Interaction Loop" (CHIL) and the U.S. National Institute of Standards and Technology (NIST).

Comparison with ImageCLEF

The principal difference between these evaluation conferences is in the document and query types that they focus on. This necessarily leads to differences in the way in which the tasks are structured and the evaluation metrics used. Core similarities remain – evaluating the state of the art in information retrieval for structured text, music, image, video or geographical queries. While the specific metrics may vary, they remain based on the notion of document relevance and ranking a retrieved list of documents. This is also true for annotation tasks where the confidence of a label is used.

A common feature among many evaluation campaigns is the regular changes to the tasks or strands of the challenge. Tasks not only get new content each year but may change their focus,

[17] http://www.imageval.org/e_presentation.html
[18] http://www.benchathlon.net/
[19] http://clear-evaluation.org/

evolving to meet the needs of the research community and the latest challenges. New tasks are constantly proposed and old tasks retired. The ongoing evolution and diversity of the challenges helps to keep evaluation campaigns relevant.

Multimedia objects are a highly multidimensional and collections often also include transcripts or other text-based metadata that is useful for information retrieval purposes. Until the 2009 TRECVid the video retrieval task required a run to be submitted that only used the text data to demonstrate that an improvement was achieved using the media content over that of using text alone. In the early days of TRECVid it was often found that content-based or visual methods displayed little or no improvement over using the video transcription to retrieve video segments. Recent results have consistently demonstrated that using content-based methods has improved system performance and hence this requirement has been dropped.

ImageCLEF and GeoCLEF obviously have roots in the CLEF multi-lingual text retrieval evaluation conference and thus also have a focus on cross-language retrieval. By necessity, multi-linguality requires the inclusion of text data in the document collection — images are generally language independent.

TRECVid has an option in retrieval track which allows searches to be performed by a user interacting with the search system to submit and refine queries. The resulting evaluation is only conducted on the ranked results list and, officially at least, does not include capturing user feedback on the system usability for comparison. VideOlympics, which is based on the interactive track of TRECVid, starts to move towards user-based evaluations but does not produce comparative evaluations, only demonstrations.

ImageCLEF has also increased its focus in recent years on the use of ontologies or knowledge models (e.g., Wikipedia) to improve performance. This allows better inclusion of contextual information in the queries and potentially better user experience — although this has yet to be exhaustively tested within ImageCLEF. Newly proposed evaluation metrics aim to judge the importance and effectiveness of structured knowledge in MIR.

3.3 Utility of Evaluation Conferences

A great deal has been written from the 1960s to the present time regarding the utility of batch system analysis of test collections for evaluating information retrieval approaches as compared with the real needs of users in their information environment. Apart from operational criticisms relating to the methods of generating "enough" data, the difficulty in defining relevance and determining appropriate queries, the major criticism is that system-based approaches are too far removed from the reality of user interactions and information requirements.

In defence of applying the Cranfield approach for system evaluation, Voorhees (2002) discussed the philosophical implications and concluded that, within limits, laboratory tests are a valid tool for performing this type of evaluation. This was based on analysis of a series of experiments run on TREC collections that demonstrated that comparative evaluations remain stable despite changes in the relevance judgements. Salton (1992) examines a number of key criticisms regarding laboratory based retrieval system evaluation and finally concludes that "there should be no question at this point about the usefulness and effectiveness of properly designed automatic text retrieval systems". Harman has, in general, been supportive of Cranfield based evaluation particularly in her role at TREC (Harman, 2005). We await with interest her keynote talk at SIGIR 2010 titled "Is the

Cranfield Paradigm Outdated?"[20].

In contrast, Järvelin (2009) argues strongly that Cranfield-style approaches are limited and insufficient to explain searcher behaviour principally on the basis that the resulting comparison and analysis lacks inclusion of the user contexts. Hersh et al (2000); Turpin and Hersh (2001) present results that argue that end-users perceive little or no difference between the performance of a baseline system and one shown to be "significantly" better in relevance-based evaluations. Almost 20 years ago Brookes (1981) questioned the continuing usefulness of applying the "Cranfield paradigm" for information retrieval evaluation stating that it was an evaluation from the "computer science" side and did not reach out to fulfil the needs of "information science". More recently, Järvelin (2009) stated "there is mounting evidence that we may not be able to improve human performance by further improving traditional research effectiveness".

In spite of appearances, the conclusions from the literature summarised here are not incompatible. Rather it is clear that evaluation of information retrieval systems requires consideration of the system *in situ* rather than solely *in vitro* or *in silico*. Certain tracks and tasks of the various evaluation conferences do consider the user. For example, both TRECVid and ImageCLEF have had interactive strands for retrieval tasks that involve user participation. TRECVid allows iterative querying of the system to develop the results list, mimicking user behaviour in real life where queries are refined based on the results list. ImageCLEF 2003, 2004 and 2005 had an interactive image retrieval task that used user questionnaires to explore variations of retrieval systems within a submission. That is, results from participants were not compared. From 2006 this task was merged with the main interactive CLEF track (iCLEF[21]).

Many of the papers written on the topic of information retrieval evaluation and referenced here predate the Internet and are heavily focused on traditional text-based IR. What is the implication for multimedia? Batch system analysis is dependent on the creation of sufficiently large and well-annotated test sets. Building such sets of multimedia documents is time consuming and expensive. Evaluation campaigns are invaluable in providing standard data sets and a forum to conduct such experiments. However, in many ways user-based evaluation is even more critical for multimedia document search and retrieval. The high density of information contained in images, audio or video and the often subjective interpretation create more complications for determining relevance and increase the importance of personal context.

Saracevic (1995) stated that IR was increasingly being embedded in other systems — e.g., the Internet, digital libraries — and noted that new evaluations in this context needed to be incorporated. Since this paper was published, both ImageCLEF and TRECVid have embraced the knowledge context found on the Internet with tasks that incorporate contextual knowledge about images from webpages, use of Wikipedia and inclusion of social networking information.

Finally, there is of course more to evaluation conferences than simply the chance to execute a batch system evaluation with a suitably large and well documented test set. The collaborative nature, the focus and time-pressure of producing a submission and the opportunity to openly share and explore approaches are in many ways the more valuable result of participation.

[20] http://www.sigir2010.org/doku.php?id=program:keynotes
[21] http://nlp.uned.es/iCLEF/

3.4 Impact and Evolution of Metrics

In the previous section we described a number of different evaluation campaigns and looked at some of the criticisms of evaluation conferences for information retrieval research. We found that while a holistic approach to evaluation is required there is benefit in applying batch system-based evaluations. Here we will look at the use of metrics to assess information retrieval performance. The definitions and explanations of the metrics used in ImageCLEF have been given elsewhere in this book. In this section we aim to discuss the real-world impact of using these metrics to assess information retrieval research and some of the problems that can be caused by over reliance on judging performance using only a narrow selection of numerical metrics. We also suggest some newly proposed metrics for evaluation that may help.

Perhaps the biggest challenge to conducting system-based evaluations is the need to assign a relevance judgement to each document in the data set for each query. This raises significant practical problems as manual annotation is time-consuming, expensive and subjective — all motivations behind research into automatic image annotation or content-based search. Evaluation campaigns often go some way towards sharing this cost among research groups. TRECVid, for example, conducts a shared annotation phase where participants manually annotate subsets of the training data to share and use in all systems. Pooling of results from all submissions is also used to reduce the volume of documents that need to be assessed in the testing phase. This may result in some relevant documents being missed but is generally accepted to provide a more efficient cost-effective method of using very large datasets.

Philosophically a larger question is how to define "relevance" for IR. Borlund (2003) provides an extensive review of the concept of relevance in IR and its importance in evaluation.Cosijn and Ingwersen (2000) also discuss the difficulties of consistently and accurately defining relevance and propose a model based on the notion of socio-cognitive relevance. Saracevic (1997) in his acceptance speech for the 1997 ACM SIGIR Gerald Salton Award talks about the impossibility of separating users from the notion of relevance — by its very definition it requires user involvement and user judgement.

Ellis (1996) describes the "dilemma of measurement" in information science that seeks to perform exact measurements in the scientific style but uses human judgement of relevance and concluded that the Cranfield tests "oversimplified the inherent complexity of the retrieval interaction in the pursuit of quantification". This tension between the desire for a clear, quantitative method for comparing and defining improvements in IR and the fundamental variations that occur when using human judgements about document relevance continues to drive research into IR evaluation methodologies.

Soboroff et al (2001) conducted interesting experiments that extend those by Voorhees (2000) into the impact of differing relevance judgements on comparative system performance. Using data from TREC the hypothesis that variations in relevance had minimal impact of the relative ranking of systems was assessed. Interestingly, they found that even random assignments of relevance based on the pooled TRECVid results produced rankings that correlated positively with the official TREC result although it was not possible to predict system performance. This reinforces the view that evaluation campaign results should be used carefully outside of the context of the comparative workshop. Earlier work by Zobel (1998), who proposed a new method for pooling system results, reached similar conclusions.

Buckley and Voorhees (2000) ask two questions regarding evaluation methods for IR: "how to build and validate good test collections" and "what measures should be used to evaluate retrieval effectiveness". They examine a number of common metrics based on precision and recall of relevant

documents, discuss the general rules of thumb (e.g., for size of the data or query set) and look at a method for quantifying the confidence that can be placed in the experimental conclusions. As a result, they suggest that IR evaluation papers should include results from several collections.

The precision and recall metrics used by Buckley and Voorhees (2000) are fairly standard within IR. Other terms that are commonly used include those that aim to quantitatively measure the overall improvement or gain achieved by one ranked list over another (Järvelin and Kekäläinen, 2002). That is, is the order of results provided by system A better than that provided by system B? (Sakai, 2007) reviews a number of graded-relevance retrieval metrics and concludes that they are "at least as stable and sensitive as Average Precision, and are fairly robust to the choice of gain values". Researchers are also exploring areas such as novelty and diversity to compare the performance of systems from a more user-friendly perspective (Clarke et al, 2008).

Newer metrics that aim to address limitations with measures that use recall have also been proposed. Moffat and Zobel (2008) suggest *rank-biased precision* derived from a model of user behaviour as a replacement for average precision that measures the behaviour as observed by the user. This publication also lays out a clear case for and against the use of measures such as average precision and the benefits of metrics that consider a broader range of the user's perspective.

The Photo Annotation task at ImageCLEF 2009 calculated a new evaluation metric based on ontology scoring from Nowak et al (2010) that aimed to measure how information obtained from ontologies improved system performance. This metric supported the focus on multi-modal approaches to photo annotation. Measures such as this, while not necessarily improving the user focus of evaluation, do extend the scope of evaluation campaigns and enrich the discussion surrounding the system performance beyond that of single quantitative rankings.

Finally, too narrow a focus on single quantitative evaluation metrics can lead over-fitting of the system to produce optimal results for one or more evaluation campaigns that cannot be transferred to real world performance. Both TRECVid and ImageCLEF aim to mitigate this by providing multiple metrics for judging performance and comparing systems internally. There is also a clear expectation that precludes participants making exaggerated claims about the system performance outside of the evaluation workshop — particularly in a commercial setting. In TREC and TRECVid this takes the form of an explicit user agreement signed by participants.

3.5 Conclusions

In this chapter we have described the ImageCLEF/TRECVid participation experience of an MIR research group and examined the issues surrounding evaluation campaigns in IR. The main issues in IR evaluation focus on the weaknesses of system-based evaluation in isolation, the problems in assessing IR system performance outside of real-world context and without user input and the seductive difficulty in finding a quantitative measure of IR success. We have not presented an exhaustive analysis of all of the available literature on evaluation in information retrieval — there is an extensive volume of literature stretching back over almost 50 years of research in information science.

It is unfortunate that parties external to the evaluation communities can sometimes view them as a "competition" and judge system performance in isolation based only on the numbers. Research has shown that the use of system evaluation based on the Cranfield approach can be a valid and useful approach to drive the development of information retrieval. It is also clear that these judgements cannot necessarily be applied outside of the evaluation workshop context to determine the absolute, real-world usefulness or effectiveness of one system over another.

User-based evaluation of multimedia information retrieval systems is challenging due to difficulties in finding appropriate users, setting up systems with consistent and functionally complete user interfaces (that do not impact excessively on the perception of performance) and running experiments that are comparable and reproducible. Evaluation campaigns do a fantastic job of helping researchers conduct quality, large-scale system-based evaluations that drive research and improve technology. As multimedia information retrieval moves forward, we believe that holistic user focused evaluations will become increasingly important. How can the communities of researchers and users mobilised by evaluation campaigns contribute to this process?

ImageCLEF is useful and fulfils a significant need in multimedia IR. It is important to also consider a holistic view of information retrieval systems and not to focus solely on the ranking or single performance values. Evolving tasks that incorporate external context and begin to include users and interactivity are improving the outcomes for MIR. Emerging metrics that focus on other aspects are providing new insights into IR system performance and will be beneficial to incorporate into future evaluation campaigns. The diversity of tasks in evaluation campaigns helps to approximate user needs in small, specific situations. We believe this fragmentation helps to simulate the variety of contexts and situations that occur in MIR and contributes to improving real-world information retrieval systems. The diversity found within evaluation campaigns will continue to drive MIR research and play a vital role in future developments.

Acknowledgements. The authors (SL, AL and SR) would like to acknowledge the contributions of many current and past members of the MMIS group in participating in ImageCLEF and TRECVid — Paul Browne, Shyamala Doraisamy, Daniel Heesch, Peter Howarth, Rui Hu, Haiming Liu, João Magalhães, Simon Overell, Marcus Pickering, Adam Rae and Alexei Yavlinsky, among many others.

Bibliography

C Aggarwal, A Hinneburg and D Keim (2001). On the surprising behavior of distance metrics in high dimensional space. In *International Conference on Database Theory*, pp 420–434. Springer LNCS 1973. DOI: 10.1007/3-540-44503-X_27.

C Aggarwal and P Yu (2000). The IGrid index: reversing the dimensionality curse for similarity indexing in high dimensional space. In *ACM International Conference on Knowledge Discovery and Data Mining*, pp 119–129. DOI: 10.1145/347090.347116.

L von Ahn and L Dabbish (2004). Labeling images with a computer game. In *ACM International Conference on Human Factors in Computing Systems*, pp 319–326. DOI: 10.1145/985692.985733.

M Baillie and J Jose (2004). An audio-based sports video segmentation and event detection algorithm. In *IEEE International Conference on Computer Vision and Pattern Recognition*, pp 110. DOI: 10.1109/CVPR.2004.298.

B Bartell, G Cottrell and R Belew (1994). Automatic combination of multiple ranked retrieval systems. In *ACM International Conference on Research and Development in Information Retrieval*, pp 173–181.

J Beis and D Lowe (1997). Shape indexing using approximate nearest-neighbour search in high-dimensional spaces. In *IEEE International Conference on Computer Vision and Pattern Recognition*, pp 1000–1006. DOI: 10.1109/CVPR.1997.609451.

R Bellman (1961). *Adaptive control processes: a guided tour*. Princeton University Press.

K Beyer, J Goldstein, R Ramakrishnan and U Shaft (1999). When is "nearest neighbor" meaningful? In *International Conference on Database Theory*, pp 217–235. Springer LNCS 1540. DOI: 10.1007/3-540-49257-7_15.

W Birmingham, R Dannenberg and B Pardo (2006). Query by humming with the VocalSearch system. *Communications of the ACM 49*(8), 49–52. DOI: 10.1145/1145287.1145313.

D Blei and M Jordan (2003). Modeling annotated data. In *ACM International Conference on Research and Development in Information Retrieval*, pp 127–134. DOI: 10.1145/860435.860460.

P. Borlund (2003). The concept of relevance in IR. *Journal of the American Society for information Science and Technology 54*(10), 913–925.

A Bozzon, M Brambilla, P Fraternali, F Nucci, S Debald, E Moore, W Neidl, M Plu, P Aichroth, O Pihlajamaa, C Laurier, S Zagorac, G Backfried, D Weinland and V Croce (2009). PHAROS:

an audiovisual search platform. In *ACM International Conference on Research and Development in Information Retrieval*, pp 841. DOI: 10.1145/1571941.1572161.

A Broder (1997). On the resemblance and containment of documents. In *Compression and Complexity of Sequences*, pp 21–29. DOI: 10.1109/SEQUEN.1997.666900.

B.C. Brookes (1981). Information technology and the science of information. *Information retrieval research. London: Butterworths*, 1–8.

C. Buckley and E.M. Voorhees (2000). Evaluating evaluation measure stability. In *Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval*, pp 40. ACM.

V Bush (1945). As we may think. *The Atlantic Monthly 176*(1), 101–108. Reprinted in http://dx.doi.org/10.1145/227181.227186.

P Cano, E Batlle, T Kalker and J Haitsma (2005). A review of audio fingerprinting. *41*(3), 271–284. DOI: 10.1007/s11265-005-4151-3.

A Cavallaro and T Ebrahimi (2004). Interaction between high-level and low-level image analysis for semantic video object extraction. *Journal on Applied Signal Processing 2004*(6), 786–797. DOI: 10.1155/S1110865704402157.

R Cilibrasi and P Vitányi (2007). The Google similarity distance. *IEEE Transactions on Knowledge and Data Engineering 19*(3), 370–383. DOI: 10.1109/TKDE.2007.48.

C.L.A. Clarke, M. Kolla, G.V. Cormack, O. Vechtomova, A. Ashkan, S. B ”uttcher and I. MacKinnon (2008). Novelty and diversity in information retrieval evaluation. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, pp 659–666. ACM.

CW Cleverdon, J. Mills and EM Keen (1966). Factors determining the performance of indexing systems,(Volume 1: Design). *Cranfield: College of Aeronautics*.

L Cooniss, A Ashford and M Graham (2000). Information seeking behaviour in image retrieval. VISOR 1 final report. Technical report, Library and Information Commission Research Report, British Library.

L Cooniss, J Davis and M Graham (2003). A user-oriented evaluation framework for the development of electronic image retrieval systems in the workplace: VISOR 2 final report. Technical report, Library and Information Commission Research Report, British Library.

Erica Cosijn and Peter Ingwersen (2000). Dimensions of relevance. *Information Processing & Management 36*(4), 533 – 550.

M Datar, N Immorlica, P Indyk and V Mirrokni (2004). Locality-sensitive hashing scheme based on p-stable distributions. In *ACM Annual Symposium on Computational Geometry*, pp 253–262. DOI: 10.1145/997817.997857.

R Datta, D Joshi, J Li and J Wang (2008). Image retrieval: ideas, influences, and trends of the new age. *ACM Computing Surveys 40*(2), 1–60. DOI: 10.1145/1348246.1348248.

A del Bimbo and P Pala (1997). Visual image retrieval by elastic matching of user sketches. *IEEE Transactions on Pattern Analysis and Machine Intelligence 19*(2), 121–132. DOI: 10.1109/34.574790.

T Deselaers, D Keysers and H Ney (2008). Features for image retrieval: an experimental comparison. *Information Retrieval 11*(2), 77–107. DOI: 10.1007/s10791-007-9039-3.

S Doraisamy (2005). *Polyphonic music retrieval: the n-gram approach*. PhD thesis, Imperial College London.

S Downie (2008). The music information retrieval evaluation exchange (2005-2007): a window into music information retrieval research. *Acoustical Science and Technology 29*(4), 247–255. DOI: 10.1250/ast.29.247.

S Downie and M Nelson (2000). Evaluation of a simple and effective music information retrieval method. In *ACM International Conference on Research and Development in Information Retrieval*, pp 73–80. DOI: 10.1145/345508.345551.

P Duygulu, K Barnard, N de Freitas and D Forsyth (2002). Object recognition as machine translation: learning a lexicon for a fixed image vocabulary. In *European Conference on Computer Vision*, pp 349–354. Springer LNCS 2353. DOI: 10.1007/3-540-47979-1_7.

D. Ellis (1996). The dilemma of measurement in information retrieval research. *Journal of the American Society for Information Science 47*(1), 23–36.

P Enser and C Sandom (2002). Retrieval of archival moving imagery — CBIR outside the frame? In *International Conference on Image and Video Retrieval*, pp 85–106. Springer LNCS 2383. DOI: 10.1007/3-540-45479-9_22.

P Enser and C Sandom (2003). Towards a comprehensive survey of the semantic gap in visual image retrieval. In *International Conference on Image and Video Retrieval*, pp 163–168. Springer LNCS 2728. DOI: 10.1007/3-540-45113-7_29.

S Feng, R Manmatha and V Lavrenko (2004). Multiple Bernoulli relevance models for image and video annotation. In *IEEE International Conference on Computer Vision and Pattern Recognition*, pp 1002–1009. DOI: 10.1109/CVPR.2004.1315274.

H Freeman (1961). On the encoding of arbitrary geometric configurations. *IEEE Transactions on Electronic Computers 10*(2), 260–268. DOI: 10.1109/TEC.1961.5219197.

A Gilliland-Swetland (1998). Defining metadata. In M Baca (Ed), *Introduction to Metadata: Pathways to Digital Information*. Getty Information Institute.

J Gracia and E Mena (2008). Web-based measure of semantic relatedness. In *International Conference on Web Information Systems Engineering*, pp 136–150. Springer LNCS 5175. DOI: 10.1007/978-3-540-85481-4_12.

A Guttman (1984). R-trees: a dynamic index structure for spatial searching. In *ACM International Conference on Management of Data*, pp 47–57. DOI: 10.1145/971697.602266.

J Haitsma and T Kalker (2003). A highly robust audio fingerprinting system with an efficient search strategy. *Journal of New Music Research 32*(2), 211–221. DOI: 10.1076/jnmr.32.2.211.16746.

J Hare and P Lewis (2004). Salient regions for query by image content. In *International Conference on Image and Video Retrieval*, pp 264–268. Springer LNCS 3115. DOI: 10.1007/b98923.

J Hare and P Lewis (2005). Saliency-based models of image content and their application to auto-annotation by semantic propagation. In *Multimedia and the Semantic Web Workshop at the European Semantic Web Conference*.

J Hare, P Lewis, P Enser and C Sandom (2006). Mind the gap: another look at the problem of the semantic gap in image retrieval. In *Multimedia Content Analysis, Management and Retrieval, SPIE Vol 6073*, pp 1–12. DOI: 10.1117/12.647755.

D. Harman (2005). The importance of focused evaluations: A Case Study of TREC and DUC. *Charting a New Course: Natural Language Processing and Information Retrieval 16*, 175–194.

D Heesch, P Howarth, J Magalhães, A May, M Pickering, A Yavlinsky and S Rüger (2004). Video retrieval using search and browsing. In *TREC Video Retrieval Evaluation*.

D Heesch, M Pickering, S Rüger and A Yavlinsky (2003). Video retrieval using search and browsing with key frames. In *TREC Video Retrieval Evaluation*.

W. Hersh, A. Turpin, S. Price, B. Chan, D. Kramer, L. Sacherek and D. Olson (2000). Do batch and user evaluations give the same results? In *Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval*, pp 24. ACM.

F Hillier and G Lieberman (1990). *Introduction to mathematical programming*. McGraw-Hill.

D Hillmann and E Westbrooks (Eds) (2004). *Metadata in practice*. American Library Association.

P Howarth and S Rüger (2005a). Fractional distance measures for content-based image retrieval. In *European Conference on Information Retrieval*, pp 447–456. Springer LNCS 3408. DOI: 10.1007/b107096.

P Howarth and S Rüger (2005b). Robust texture features for still-image retrieval. *IEE Proceedings on Vision, Image and Signal Processing 152*(6), 868–874. DOI: 10.1049/ip-vis:20045185.

P Howarth and S Rüger (2005c). Trading precision for speed: localised similarity functions. In *International Conference on Image and Video Retrieval*, pp 415–424. Springer LNCS 3568. DOI: 10.1007/11526346_45.

P Howarth, A Yavlinsky, D Heesch and S Rüger (2005). Medical image retrieval using texture, locality and colour. In *Lecture Notes from the Cross Language Evaluation Forum 2004*, Springer LNCS 3491, pp 740–749.

R Hu, S Rüger, D Song, H-M Liu and Z Huang (2008). Dissimilarity measures for content-based image retrieval. In *IEEE International Conference on Multimedia and Expo*, pp 1365–1368. DOI: 10.1109/ICME.2008.4607697.

International Commission on Illumination (1986). CIE colorimetry.

S Intner, S Lazinger and J Weihs (2006). *Metadata and its impact on libraries*. Westport, CT: Libraries Unlimited. DOI: 10.1336/1591581451.

K. Järvelin (2009). Explaining user performance in information retrieval: Challenges to IR evaluation. *Advances in Information Retrieval Theory*, 289–296.

Kalervo Järvelin and Jaana Kekäläinen (2002). Cumulated gain-based evaluation of ir techniques. *ACM Trans. Inf. Syst. 20*(4), 422–446.

J Jeon, V Lavrenko and R Manmatha (2003). Automatic image annotation and retrieval using cross-media relevance models. In *ACM International Conference on Research and Development in Information Retrieval*, pp 119–126. DOI: 10.1145/860435.860459.

R. Jesus, J. Magalhães, A. Yavlinsky and S. Rüger (2005). Imperial college at trecvid. In *TREC Video Retrieval Evaluation*, Gaithersburg, MD.

C Lagoze and S Payette (2000). Metadata: principles, practices and challenges. In A Kenney and O Rieger (Eds), *Moving Theory into Practice: Digital Imaging for Libraries and Archives*, Mountain View, CA. Research Libraries Group.

M Lalmas (2009). *XML retrieval*. Synthesis Lectures on Information Concepts, Retrieval, and Services. Morgan & Claypool Publishers. DOI: 10.2200/S00203ED1V01Y200907ICR007.

V Lavrenko, R Manmatha and J Jeon (2003). A model for learning the semantics of pictures. In *Neural Information Processing Systems*, pp 553–560.

V Levenshtein (1966). Binary codes capable of correcting deletions, insertions and reversals. *Soviet Physics — Doklady 10*(8), 707–710. Translated from *Doklady Akademii Nauk SSSR, 163(4),* pp 845–848, 1965.

M Lew, N Sebe, C Djeraba and R Jain (2006). Content-based multimedia information retrieval: state of the art and challenges. *ACM Transactions on Multimedia Computing, Communications, and Applications 2*(1), 1–19. DOI: 10.1145/1126004.1126005.

S Little and S Rüger (2009). Conservation of effort in feature selection for image annotation. In *IEEE Workshop on Multimedia Signals Processing*.

C Liu, J Yuen and A Torralba (2009a). Nonparametric scene parsing: label transfer via dense scene alignment. In *IEEE International Conference on Computer Vision and Pattern Recognition*, pp 1972–1979. DOI: 10.1109/CVPRW.2009.5206536.

Z Liu, Y Wang and T Chen (1998). Audio feature extraction and analysis for scene segmentation and classification. *VLSI Signal Processing 20*(1–2), 61–79. DOI: 10.1023/A:1008066223044.

A Llorente, S Little and S Rüger (2009). MMIS at ImageCLEF 2009: Non-parametric Density Estimation Algorithms. In *Working notes for the CLEF 2009 Workshop*, Corfu, Greece.

A Llorente, S Zagorac, S Little, R Hu, A Kumar, S Shaik, X Ma and S Rüger (2008, Nov). Semantic video annotation using background knowledge and similarity-based video retrieval. In *TREC Video Retrieval Evaluation (TRECVid, Gaithersburg, MD)*.

D Lowe (1999). Object recognition from local scale-invariant features. In *IEEE International Conference on Computer Vision*, Volume 2, pp 1150–1157. DOI: 10.1109/ICCV.1999.790410.

D Lowe (2004). Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision 60*(2), 91–110. DOI: 10.1023/B:VISI.0000029664.99615.94.

J. Magalhães, S. Overell, A. Yavlinsky and S. Rüger (2006). Imperial college at TRECVID. In *TREC Video Retrieval Evaluation*, Gaithersburg, MD.

J Magalhães and S Rüger (2006). Logistic regression of semantic codebooks for semantic image retrieval. In *International Conference on Image and Video Retrieval*, pp 41–50. Springer LNCS 4071. DOI: 10.1007/11788034_5.

J Magalhães and S Rüger (2007). Information-theoretic semantic multimedia indexing. In *International Conference on Image and Video Retrieval*, pp 619–626. DOI: 10.1145/1282280.1282368.

A Makadia, V Pavlovic and S Kumar (2008). A new baseline for image annotation. In *European Conference on Computer Vision*, pp 316–329. Springer LNCS 5304. DOI: 10.1007/978-3-540-88690-7_24.

T Mandl, F Gey, G Di Nunzio, N Ferro, R Larson, M Sanderson, D Santos, C Womser-Hacker and X Xie (2008). GeoCLEF 2007: the CLEF 2007 cross-language geographic information retrieval track overview. In *Advances in Multilingual and Multimodal Information Retrieval*, pp 745–772. Springer LNCS 5152. DOI: 10.1007/978-3-540-85760-0_96.

K Mc Donald and A Smeaton (2005). A comparison of score, rank and probability-based fusion methods for video shot retrieval. In *International Conference on Image and Video Retrieval*, pp 61–70. Springer LNCS 3568. DOI: 10.1007/11526346_10.

D Messing, P van Beek and J Errico (2001). The MPEG-7 colour structure descriptor: image description using colour and local spatial information. In *International Conference on Image Processing*, Volume 1, pp 670–673. DOI: 10.1109/ICIP.2001.959134.

D Metzler and R Manmatha (2004). An inference network approach to image retrieval. In *International Conference on Image and Video Retrieval*, pp 42–50. Springer LNCS 3115. DOI: 10.1007/b98923.

T Mitchell (1997). *Machine learning*. McGraw Hill.

A. Moffat and J. Zobel (2008). Rank-biased precision for measurement of retrieval effectiveness.

M Muja and D Lowe (2009). Fast approximate nearest neighbors with automatic algorithm configuration. In *International Conference on Computer Vision Theory and Applications*, Volume 1, pp 331–340.

W Müller and A Henrich (2004). Faster exact histogram intersection on large data collections using inverted VA-files. In *International Conference on Image and Video Retrieval*, pp 455–463. Springer LNCS 3115. DOI: 10.1007/b98923.

S Nene and S Nayar (1997). A simple algorithm for nearest neighbor search in high dimensions. *IEEE Transactions on Pattern Analysis and Machine Intelligence 19*(9), 989–1003. DOI: 10.1109/34.615448.

Stefanie Nowak, Hanna Lukashevich, Peter Dunker and Stefan Rüger (2010). Performance measures for multilabel classification - a case study in the area of image classification. In *ACM SIGMM International Conference on Multimedia Information Retrieval (ACM MIR)*, Philadelphia, Pennsylvania.

P Over and A Smeaton (Eds) (2007). *TVS 2007: proceedings of the international workshop on TRECVid video summarization*. DOI: 10.1145/1290031. ACM, ISBN 978-1-59593-780-3.

S Overell, A Llorente, H-M Liu, R Hu, A Rae, J Zhu, D Song and S Rüger (2008, Sep). MMIS at ImageCLEF 2008: Experiments combining different evidence sources. In *CLEF workshop, working notes,*, Aarhus, Denmark.

S. Overell, J. Magalhães and S. Rüger (2006). Place disambiguation with co-occurrence models. In *CLEF 2006*, Alicante, Spain.

E Persoon and K-S Fu (1977). Shape discrimination using Fourier descriptor. *IEEE Transactions on Systems, Man and Cybernetics 7*(3), 170–179. DOI: 10.1109/TSMC.1977.4309681.

M Pickering and S Rüger (2002). Multi-timescale video shot-change detection. In *Text Retrieval Conf, NIST (Trec, Gaithersburg, MD, Nov 2001), NIST Special Publication 500-250*, pp pp 275?278.

M Pickering and S Rüger (2003). Evaluation of key-frame based retrieval techniques for video. *Computer Vision and Image Understanding 92*(2), 217–235. DOI: 10.1016/j.cviu.2003.06.002.

J Puzicha, T Hofmann and J Buhmann (1997). Non-parametric similarity measures for unsupervised texture segmentation and image retrieval. In *IEEE International Conference on Computer Vision and Pattern Recognition*, pp 267–272. DOI: 10.1109/CVPR.1997.609331.

C. J. van Rijsbergen (1989). Towards an information logic. In *SIGIR '89: Proceedings of the 12th annual international ACM SIGIR conference on Research and development in information retrieval*, New York, NY, USA, pp 77–86.

Y Rubner, C Tomasi and L Guibas (2000). The earth mover's distance as a metric for image retrieval. *International Journal of Computer Vision 40*(2), 99–121. DOI: 10.1023/A:1026543900054.

Stefan Rüger (2010). *Multimedia information retrieval*. Lecture notes in the series Synthesis Lectures on Information Concepts, Retrieval, and Services. Morgan-Claypool.

T. Sakai (2007). On the reliability of information retrieval metrics based on graded relevance. *Information Processing & Management 43*(2), 531–548.

Gerard Salton (1992). The state of retrieval system evaluation. *Information Processing and Management 28*(4), 441 – 449. Special Issue: Evaluation Issues in Information Retrieval.

A Salway and M Graham (2003). Extracting information about emotions in films. In *ACM Conference on Multimedia*, pp 299–302. DOI: 10.1145/957013.957076.

A Salway, A Vassiliou and K Ahmad (2005). What happens in films? In *IEEE International Conference on Multimedia and Expo*, pp 4. DOI: 10.1109/ICME.2005.1521357.

T. Saracevic (1995). Evaluation of evaluation in information retrieval. In *Proceedings of the 18th annual international ACM SIGIR conference on Research and development in information retrieval*, pp 138–146. ACM.

T. Saracevic (1997). Users lost: Reflections on the past, future, and limits of information science. In *ACM Sigir Forum*, Volume 31, pp 16–27. ACM.

R Sinha and M Winslett (2007). Multi-resolution bitmap indexes for scientific data. *ACM Transactions on Database Systems 32*(3), *Article 16*, 1–39. DOI: 10.1145/1272743.1272746.

A Sinitsyn (2006). Duplicate song detection using audio fingerprinting for consumer electronics devices. In *IEEE International Symposium on Consumer Electronics*, pp 1–6. DOI: 10.1109/ISCE.2006.1689403.

A Smeaton, P Over and W Kraaij (2006). Evaluation campaigns and TRECVid. In *ACM International Workshop on Multimedia Information Retrieval*, pp 321–330. DOI: 10.1145/1178677.1178722.

A Smeaton, P Over and W Kraaij (2009b). High-level feature detection from video in TRECVid: a 5-year retrospective of achievements. In A Divakaran (Ed), *Multimedia Content Analysis: Theory and Applications*, pp 151–174. Springer. DOI: 10.1007/978-0-387-76569-3_6.

A Smeulders, M Worring, S Santini, A Gupta and R Jain (2000). Content-based image retrieval at the end of the early years. *IEEE Transactions on Pattern Analysis and Machine Intelligence 22*(12), 1349–1380. DOI: 10.1109/34.895972.

Cees G. M. Snoek, Marcel Worring, Ork de Rooij, Koen E. A. van de Sande, Rong Yan and Alexander G. Hauptmann (2008, January-March). VideOlympics: Real-time evaluation of multimedia retrieval systems. *IEEE MultiMedia 15*(1), 86–91.

I. Soboroff, C. Nicholas and P. Cahan (2001). Ranking retrieval systems without relevance judgments. In *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, pp 66–73. ACM.

H Tamura, S Mori and T Yamawaki (1978). Textural features corresponding to visual perception. *IEEE Transactions on Systems, Man and Cybernetics 8*(6), 460–472. DOI: 10.1109/TSMC.1978.4309999.

T Tolonen and M Karjalainen (2000). A computationally efficient multi-pitch analysis model. *IEEE Transactions on Speech and Audio Processing 8*(6), 708–716. DOI: 10.1109/89.876309.

A Torralba and A Oliva (2003). Statistics of natural image categories. *Network: Computation in Neural Systems 14*, 391–412. DOI: 10.1088/0954-898X/14/3/302.

A.H. Turpin and W. Hersh (2001). Why batch and user evaluations do not give the same results. In *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, pp 225–231. ACM.

G Tzanetakis and P Cook (2002). Musical genre classification of audio signals. *IEEE Transactions on Speech and Audio Processing 10*(5), 293–302. DOI: 10.1109/TSA.2002.800560.

E. Voorhees (2002). The philosophy of information retrieval evaluation. In *Evaluation of cross-language information retrieval systems*, pp 143–170. Springer.

Ellen M. Voorhees (2000). Variations in relevance judgments and the measurement of retrieval effectiveness. *Information Processing & Management 36*(5), 697 – 716.

A de Vries, N Mamoulis, N Nes and M Kersten (2002). Efficient k-nn search on vertically decomposed data. In *ACM International Conference on Management of Data*, pp 322–333. DOI: 10.1145/564691.564729.

T Wallace and P Wintz (1980). An efficient three dimensional aircraft recognition algorithm using normalized Fourier descriptors. *Computer Graphics and Image Processing 13*(2), 99–126. DOI: 10.1016/S0146-664X(80)80035-9.

A Wang (2003). An industrial-strength audio search algorithm. In *International Conference on Music Information Retrieval*, pp 7–13.

R Weber, H-J Stock and S Blott (1998). A quantitative analysis and performance study for similarity search methods in high-dimensional space. In *International Conference on Very Large Databases*, pp 194–205.

T Westerveld and R van Zwol (2006). The INEX 2006 multimedia track. In *Comparative Evaluation of XML Information Retrieval Systems, International Workshop of the Initiative for the Evaluation of XML Retrieval*, pp 331–344. Springer LNCS 4518. DOI: 10.1007/978-3-540-73888-6_33.

I Witten, D Bainbridge and D Nichols (2010). *How to build a digital library* (2nd ed). Morgan Kaufmann.

K Wu, E Otoo and A Shoshani (2004a). On the performance of bitmap indices for high cardinality attributes. In *International Conference on Very Large Databases*, pp 24–35.

A Yavlinsky, E Schofield and S Rüger (2005). Automated image annotation using global features and robust nonparametric density estimation. In *International Conference on Image and Video Retrieval*, pp 507–517. Springer LNCS 3568. DOI: 10.1007/11526346.

S. Zagorac, A. Llorente, S. Little, H-M. Liu and S. Rüger (2009). Automated content based video retrieval. In *TREC Video Retrieval Evaluation (TRECVid, Gaithersburg, MD)*.

C Zahn and R Roskies (1972). Fourier descriptors for plane closed curves. *IEEE Transactions on Computers 21*(3), 269–281. DOI: 10.1109/TC.1972.5008949.

M Zeng and J Qin (2008). *Metadata*. New York: Neal-Schuman.

Justin Zobel (1998). How reliable are the results of large-scale information retrieval experiments? In *SIGIR '98: Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, New York, NY, USA, pp 307–314. ACM.

R van Zwol, G Kazai and M Lalmas (2005). INEX 2005 multimedia track. In *Advances in XML Information Retrieval and Evaluation, International Workshop of the Initiative for the Evaluation of XML Retrieval*, pp 497–510. Springer LNCS 3977. DOI: 10.1007/11766278.