The NN^k Technique for Image Searching and Browsing

by

Daniel Heesch

A dissertation submitted
for the degree of
Doctor of Philosophy

DEPARTMENT OF COMPUTING
IMPERIAL COLLEGE LONDON
UNIVERSITY OF LONDON
UNITED KINGDOM

November 2005

Abstract

Retrieval of images from large image archives based solely on their visual similarity to a query image provides an exciting alternative to conventional text-based search. For content-based retrieval images are represented in terms of visual features. The question of how to combine these for similarity computation is typically addressed by eliciting relevance feedback from the user on the retrieved images. We argue in this thesis that the prevailing approach to relevance feedback suffers from three significant shortcomings: firstly, it leaves unsolved the question of how to combine features for the first retrieval; secondly, the advantage of automated content-extraction over manual annotation is greatest for large collections but if the query image is not constrained to come from the indexed collection, content-based retrieval entails imagewise comparisons leading to prohibitive response times; thirdly, users may only have vaguely defined information needs or may change their needs in the course of the interaction. The large majority of relevance feedback techniques are ill-suited for such undirected exploration. We propose a new framework of user interaction that addresses these limitations. It is centred on what we call the NN^k idea. The NN^k of an image are all those images that are most similar to it under *some* combination of features. They can be viewed as representatives of the possible semantic facets an image may exhibit to different users. The NN^k idea is first applied to the problem of automated retrieval where it suggests a two-step method of relevance feedback that is shown to outperform existing techniques. In the continuation of the thesis we broaden the view and introduce NN^k Networks as static structures for browsing image collections. NN^k Networks are directed graphs in which every image is connected to all its NN^k. NN^k Networks obviate the need to articulate an information need pictorially. Moreover, by being entirely precomputed we achieve interaction times that are independent of the collection size. We investigate topological properties of the networks and analyse how well they capture the semantic structure of a collection. These formal analyses are complemented by a large-scale quantitative evaluation on 32,000 images and a set of realistic search tasks. Both approaches suggest that NN^k Networks provide a very effective alternative to automated retrieval both for directed searching and undirected browsing.

Acknowledgements

The author is delighted to acknowledge his supervisor Dr Stefan Rüger. Also gratefully acknowledged are the examiners Dr Joemon Jose (University of Glasgow) and Prof Mounia Lalmas (Queen Mary, University of London) for their valuable comments on this thesis. Special thanks go to Lisa, Max and Alexei for their subtle and manifold contributions.

This work has been carried out with the partial support from the EPSRC.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

"Hence also it has been found that a classification founded on any single character, however important that may be, has always failed."

Charles Darwin, The origin of species (1859)

1.1 Similarity

Similarity in appearance is often revealing about other, and potentially much deeper functional and causal commonalities between objects, events and situations. Things that are similar in some respect may owe their existence to similar causes and are likely to behave in a similar way. Prediction, problem-solving, and classification are cognitive skills that rely in great parts on the existence of regularities that are revealed to us through similarity. Intelligent interaction with the world is difficult to imagine without this peculiar sense of sameness.

The ability to infer from known similarities between two things to the existence of further similarities has often been used as the definition of analogical reasoning (Helman, 1988). Its power can be seen by noting that it was instrumental in the formulation of what are arguably the two most profound revolutions in the history of science: Darwin's theory of origin of species by common descent (Darwin, 1859) and Einstein's theory of general relativity. The former owes much to the intuition that similarity betrays common causes and that "propinquity of descent — the only known cause of the similarity of organic beings — is the bond, hidden as it is by various degrees of modification, which is partially revealed to us by our classification." (p. 399). Einstein took his first steps towards a generalisation of the special theory of relativity by noticing that the effect of gravity is very similar to that of uniform acceleration and, according to his equivalence principle, physically indistinguishable.

The pivotal role of similarity for human thinking has long been recognised in cognitive psychology and philosophy, and correspondingly intense have been the efforts to provide formal theories of it. Geometric models are among the most influential (Torgerson, 1958). They assume that the similarity relationships between objects can be represented geometrically in a psychological space in which dimensions correspond to salient properties such as brightness or saturation of colour. Multi-dimensional scaling (MDS) has been developed for the particular purpose of reconstructing this space (Shepard, 1962a,b; Torgerson, 1965). Given pair-wise distances between objects such as could be obtained from users' similarity judgments, MDS establishes the positions of objects in an N-dimensional space such that the distances between them accurately reflect the empirically observed similarities.

Geometric models assume that the similarity between objects can be represented by the metric distance between their high-dimensional representations. Tversky (1977) observed that the conditions of a metric are not generally satisfied by empirically obtained similarity judgments. For example, the triangle inequality is violated if an object is similar to objects A and B by virtue of sharing one property with A and another with B without A and B having any properties in common. In his contrast model, similarities are instead characterised in terms of the number of features shared between objects and those that are distinct. Features may be either present or absent, and similarity is the result of a feature-matching process.

Both types of models offer alternative explanations of how similarity is computed once objects have been cast into the appropriate representation. Both acknowledge that similarity is a compound measure that integrates over multiple attributes. Neither of them, however, addresses what is arguably a more fundamental problem: Not all features are equally useful for arriving at a similarity judgment. Objects may seem similar merely by virtue of sharing accidentals. In the context of analogical reasoning some features may be important for some inferences and not for others. "Both today and yesterday occurred in this week, yet we do not infer the further similarity that today like yesterday, is a Friday." (Russell, 1988, p. 2). The question thus arises how we arrive at the psychological space in the case of geometric models or at the set of relevant features in Tversky's contrast model. How do we discriminate between dimensions or features that matter and those that don't? Similarity-based reasoning rests crucially on our ability to distinguish essentials from accidentals. The conceptual simplicity of similarity conceals a conundrum full of operational difficulties.

1.2 Similarity-based image retrieval

Similarity-based reasoning requires efficient modes of retrieval. It is perhaps only in experimental settings that subjects have direct sensory access to the patterns which they are asked to compare. In most situations an observed pattern is evaluated by comparing it with patterns stored in memory. The efficiency with which we can classify and thus recognise objects suggests that the retrieval process is itself based on similarity. According to Steven Wolfram the use of memory "underlies almost every major aspect of human thinking. Capabilities such as generalization, analogy and intuition immediately seem very closely related to the ability to retrieve data from memory on the basis of similarity." He extends the ambit of similarity-based retrieval to the domain of logical reasoning which ultimately involves little more than "retrieving patterns of logical argument that we have learned from experience" (Wolfram, 2004, p. 627).

That early logic-based approaches to artificial intelligence proved not quite as successful as originally hoped has partly to do with their limited scope for modelling similarity. We may conjecture, more specifically, that similarity-based information retrieval is an essential prerequisites for human-like intelligence. As argued in the previous section one of the principal challenges of information retrieval lies in the choice of attributes that are relevant to a particular retrieval problem. This is a problem pertaining to information retrieval in general but it is greatly compounded in the case of image retrieval in two significant ways. First, documents readily suggest a representation in terms of its constituent words. This might not be the best representation but it is a principled start and one to which

Tversky's contrast model can readily be applied. Images do not suggest such a natural decomposition into semantic atoms. There is a certain arbitrariness about image representations. Secondly, images typically admit to a multitude of different meanings. Each one of an image's semantic facets may have its own set of supporting features so that the right set of features depends not only on the object itself but also on the user. Image dependency provides a potential application of statistical machine learning techniques but it is increasingly recognised that in order to reach beyond the level of expert systems that rely on domain-specific knowledge bases, computers require general world knowledge, perhaps even an ontogeny as suggested by proponents of embodied AI. With these requirements still out of reach, it is becoming clear that the most promising short-term approach is to seek a synergy between users and computers by bringing together the strengths of computers with those of users. Users can provide essential context and meaning. "We thus witness the emergence of a new computer philosophy. No longer is it the goal to endow machines with as many human qualities as possible and to render them intelligent. Intelligence does not arise in the machine but only through the co-operation and interaction of a user with a machine."(Mainzer, 2004, p. 164).

While interaction provides a potential means to resolve the problem of image dependency, it is strictly indispensable for disambiguating between different user-specific meanings of an image. While in the early years of image retrieval interaction was merely seen as a necessary part of the task not as an opportunity, this view has changed. Nowadays, it forms an integral part of most image retrieval systems. The rapid growth of image retrieval over the past decade may suggest that a considerable methodological diversification has by now taken place. Yet it seems that image retrieval reveals the same pattern of evolution as many a new discipline: the initial phase is characterised by a co-existence of the most varied approaches, by experimentation and the often *ad hoc* application of techniques from neighbouring disciplines. Out of this fruitful chaos gradually emerges a research programme as the community develops a clear understanding of the field's challenges and objectives. Most, albeit certainly not all, of current research in content-based image retrieval is concerned with iteratively adjusting feature weights or improving the representation of the query. The common assumption is that users articulate their needs in the form of an example image. In this thesis we will argue that some of these assumptions are too narrow and that the framework of image retrieval is in need of substantial broadening.

1.3 Organisation of the thesis

The main body of the thesis divides into seven chapters. Chapter 2 introduces current models of interaction in image retrieval. The emphasis is on relevance feedback techniques that have been developed and refined over the past few years. This emphasis reflects the fact that relevance feedback has become the standard approach to optimising retrieval systems, but also because we will introduce our own relevance feedback technique in Chapter 4. The second half of the chapter is concerned with browsing models in which interaction takes a simpler, operationally more prominent and functionally more important role. We will see that the majority of these models can be reduced to iterated retrieval and therefore fail to exploit the advantages that browsing structures can potentially offer, in particular query-free interaction and fast response times.

Chapter 3 lays the foundation for the remainder of the thesis. Given some image q, we define the set of lateral neighbours or NN^k of q. These are images that satisfy certain distance properties with respect to q. We will argue that they can be viewed as pictorial exemplifications of the set of possible semantic facets that lie within the representational scope of the chosen feature set. We describe how this image set can be determined and develop various approximations that reduce the complexity of NN^k computation. We identify the set of NN^k as a subset of the convex hull of a point set and thus highlight its close relationship to the Skyline or Pareto operator in the context of relational databases. The rest of the chapter is concerned with examining how the number of NN^k scales with the collection size and the size of the chosen feature set.

After this groundwork, Chapter 4 develops a first application of the NN^k idea to relevance feedback in automated image retrieval. The fundamental strength of NN^k search stems from the ability to associate every NN^k with an optimal combination of visual features. The selection of one or more NN^k by the user thus yields a set of optimal metrics with which we can retrieve in the subsequent step. A comparison with state-of-the-art relevance feedback techniques demonstrates its effectiveness for collections of more than 30,000 images.

Whilst successfully addressing the problem of metric optimisation upon relevance feedback, NN^k search shares two major shortcomings with conventional interaction models. Unless the query is constrained to come from the collection from which we retrieve, the time complexity of NN^k search is linear in the number of images in the collection resulting in prohibitive response times as the collection size increases. NN^k search is concerned with homing in on a set of target images and relies on users not only having a fixed information need but also a pictorial representation thereof. These conditions are not necessarily met in real situations. NN^k search does not facilitate the development of information needs and does not adjust flexibly if these should change. We shall therefore view NN^k search primarily as a motivation to propose a much broader framework of interactive image search that inherits much of its rationale from NN^k search but overcomes its principal limitations. The framework will be developed in the ensuing four chapters beginning in Chapter 5 with a high-level introduction to NN^k Networks.

In the first of the two central chapters on NN^k Networks, Chapter 6 and Chapter 7, we analyse their topological characteristics such as their degree of connectedness and the incidence of isolated subgraphs, the average distance between images and the degree of local clustering. Each of these properties are important determinants of the suitability of NN^k Networks for image search. We refer to these investigations as the formal part of the topological analysis as it ignores the semantics associated with vertices and the relevance or non-relevance of associations. The second part of the chapter turns to a semantic analysis that aims to disclose how sets of relevant images are distributed across the network. We propose three measures of compactness and apply them to the subgraphs formed by relevance classes. Our analysis reveals that relevance subgraphs are remarkably compact suggesting that once a user has found one relevant image, more relevant images can be found by following relevance paths through the network.

After these theoretical investigations of NN^k Networks, Chapter 7 turns to important practical issues that have arisen in the preceding chapters. These include the presence of disconnected components and the need to update the networks efficiently in dynamic environments. The second part of the chapter is devoted to a large-scale quantitative evaluation of NN^k Networks for category search and can be viewed as an empirical confirmation of the claims which the topological analysis of Chapter 6 has given rise to.

Chapter 8 consolidates the work on NN^k Networks by suggesting ways to automatically extract the structure that was analysed in Chapter 6 and exploited in Chapter 7. We propose to cluster the dual of the network and provide visual and quantitative assessment of the quality of the clustering.

Chapter 9 gives a summary of the thesis and highlights its principal achievements. The NN^k framework as developed in this thesis suggests multiple ways of expansion, refinement and broadening. Preliminary investigations of these possible strands of future research are pursued in the second part of that chapter.

The thesis is relatively self-contained and makes only the most modest assumptions regarding the readers' mathematical background. To make the thesis most widely accessible, many of the technical terms readers may not be familiar with are collected and explained in a glossary at the end of the thesis.

The content of the thesis has variously been published or is currently under review for publication: the second chapter has been submitted as a chapter for a forthcoming book on semantics-based image retrieval. Parts of the fourth chapter have appeared in CIVR 2004 while the six chapter provides an extended treatment of two papers presented at ECIR 2004 and CIVR 2005, respectively. The quantitative evaluation of Chapter 7 has been described in detail in the Proceedings of TRECVID 2003.

Chapter 2

Interaction Models in Image Retrieval

2.1 Introduction

This chapter sets the stage for the new framework which we will begin to develop in Chapter 3. It is conveniently divided into two parts. In the first part, Section 2.2, we will examine existing approaches to relevance feedback in the context of query by example. This is no doubt the prevailing methodological setting in which content-based information retrieval is currently being investigated and for which most of the relevance feedback techniques have been developed. In the second part of the chapter, Section 2.3, we shift our focus to retrieval systems that make provision for more exploratory search of image collections by engaging users in some form of browsing. We shall see that some of the browsing techniques are functionally complementary to the relevance feedback techniques introduced in the first part and that combining them should lead to a fruitful synergy. Since the NN^k framework provides alternative solutions to both automated retrieval and browsing, both parts are of equal pertinence to our thesis.

2.2 Automated search

Relevance feedback methods vary along several dimensions which makes any linear exposition somewhat arbitrary. We shall take as our principal axis of distinction not the type of relevance feedback elicited from the user, but how this feedback is exploited in response. Along this axis we can identify three broad classes of relevance feedback methods: query adaptation, metric optimisation and classification. While the boundary between the first and the second class is relatively clear-cut, this cannot be said of the second and the third classes but we believe the distinction is nonetheless helpful.

2.2.1 Query adaptation

Query adaptation describes the process whereby the representation of an initial query is modified automatically based on relevance feedback. Query adaptation was among the first relevance feedback techniques developed for text retrieval (Rocchio, 1971; Salton and McGill, 1982) and has since been adapted to image retrieval (Rui et al., 1997; Ishikawa et al., 1998; Porkaew et al., 1999; Zhang and Su, 2001; Aggarwal et al., 2002; Kim and Chung, 2003; Urban et al., 2003). Query adaptation subsumes as the two most important classes query point moving and query expansion.

Query point moving

The idea of query point moving is illustrated in Figure 2.1 where relevant (+) and non-relevant objects are displayed in a two-dimensional feature space with the query initially being in the bottom right quadrant. The images marked by the user correspond to the bold circles. The goal of query adaptation is to move the query point towards the relevant images and away from the non-relevant images. On the not unreasonable assumption that relevant images form natural groups in feature space, query point moving is expected to improve retrieval performance.

In Urban et al. (2003), images are represented in terms of a set of keywords and a colour histogram. Likewise, a query is composed of a textual and a visual part. Given a set of images for which the user has indicated some degree of relevance, the visual part of the query is computed as the weighted average over the relevant images.

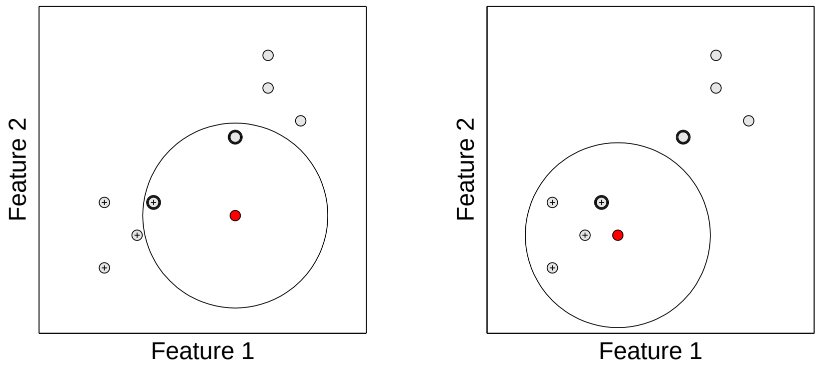


Figure 2.1: Moving the query point towards positive examples.

In Rui et al. (1997) the representation of the query is altered by taking into account both relevant and non-relevant images. The method is the direct analogue for image retrieval of Rocchio's formula (Rocchio, 1971). In particular, given sets R and N of feature vectors corresponding to relevant and non-relevant images, respectively, the learned query vector at step $t+1$ is computed as

$$q_{t+1} = \alpha q_t + \beta \left(\frac{1}{|R|} \sum_{x \in R} x \right) - \gamma \left(\frac{1}{|N|} \sum_{x \in N} x \right),$$

where α, β and γ are parameters whose values need to be suitably chosen. If α and γ are both set to zero, then the new query representation is merely the centroid of the relevant images. The goal of the method is to move the query point closer towards relevant images and further away from non-relevant images.

Ishikawa et al. (1998) and Rui and Huang (2000) develop optimisation frameworks that find the best query point as the point that minimises the summed distances to all relevant images. The optimal query representation turns out to be the weighted average of the representations of all relevant images. The distinguishing characteristic of both works is that they optimise both the query point and the distance function, a topic to which we shall return in a later section.

Before moving on to query expansion, we shall mention an interesting variation of the idea of query point moving. Instead of changing the feature representation of the query, Bang and Chen (2002) propose to modify the representation of each of the images in the collection. In addition to moving relevant images towards the query and non-relevant images away from the query, all images that have not been marked receive a contribution from the movement of each labelled image. All images change their position following the principal movement of the relevant and non-relevant images. The total movement can be expressed in the form

$$x' = x + \alpha \sum_{m \in \mathbb{M}} v_m e^{-\beta d(x,m)}(q - x),$$

where $\mathbb{M}$ is the set of feature vectors of those images on which relevance feedback has been given and v_m is a relevance score. The result cannot be expressed in terms of simple query point movement but the methods clearly share a similar rationale. The reported experiments suggest superior performance over the simple Rocchio formula when image classes are homogenous.

Query expansion

On the assumption that relevant images cluster, the method of query point moving has an intuitive appeal. Its limitations become evident when considering the possibility of relevant images forming visually distinct subsets that correspond to multiple clusters in feature space. Under this circumstance the above approach may fail to move the query into the desired direction as relevant images may suggest multiple and mutually conflicting directions. A simple modification of the above approach confers a greater robustness. It involves replacing the original query point by multiple query points each located near different subsets of the relevant images. This modification turns query adaptation into what may more aptly be described as query expansion (Porkaew et al., 1999; Kim and Chung, 2003; Urban and Jose, 2004a). By treating each of the subqueries individually and merging the results, query expansion can widen the catchment area of the similarity function by including regions where the density of relevant images appears higher. For successful deployment of the technique one needs to be careful about how to define similarity of an image to a multi-point query and, to a lesser extent, how to choose the precise location of the query points (Urban and Jose, 2004b). In Porkaew et al. (1999) relevant images are clustered and query points chosen to be cluster centroids. The overall distance of an image to the multi-point query is computed as the weighted average over the distances to each query point, i.e.

$$D(x, \mathbb{Q}) = \sum_{i:q_i \in \mathbb{Q}} w_i d(x, q_i).$$

Such a scheme is simple to implement but retains the feature it seeks to overcome by linearly averaging over individual distances. In fact, it can be shown that the overall result is equivalent to query point moving where the new query point is given by

$$Q = \sum_{i:q_i \in \mathbb{Q}} w_i q_i.$$

If d is taken to be the Euclidean distance, for example, then the iso-distance lines for a multi-point query remain circles now centred at Q. This is shown on the left plot in Figure 2.2. The method reduces therefore to the solution suggested by Ishikawa et al. (1998) and Rui and Huang (2000).

To properly account for and capitalise on the cluster structure, it seems more reasonable to treat the multi-point query as a disjunctive query, an approach taken in more recent work (Wu et al., 2000;

Kim and Chung, 2003; Urban and Jose, 2004a). Kim and Chung (2003) argue compellingly that the convex aggregation method of (Porkaew et al., 1999) is inadequate to deal with disjunctive queries and suggest a remedy in the form of

$$D(x, \mathbb{Q}) = \sum_{i:q_i \in \mathbb{Q}} d(x, q_i)^{\alpha},$$

where q_i is the representation of the ith cluster and $\alpha \in]0, 1]$ is a parameter that confers the desired non-convexity. The iso-distance lines of the model with α ranging from 1 to 1/4 are shown in Figure 2.2. For $\alpha = 1$, the model reduces to that of Porkaew et al. (1999). For smaller α, the distance function favours small distances over large distances with the result that an image close to at least one cluster is assigned a small overall distance.

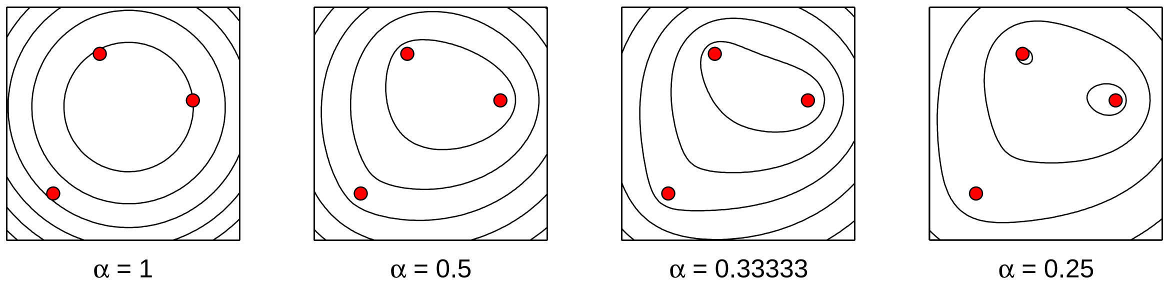


Figure 2.2: Multi-point queries: From query point moving towards disjunctive queries.

All of the above methods have in common that they are concerned with combining distance scores. An alternative approach to multi-point queries is taken in Urban and Jose (2004a) who adapt a rank-based aggregation method, median-rank aggregation (Fagin et al., 2003), to multi-point image queries and establish superior performance over the simple score-based method of Porkaew et al. (1999). The rank of an image p is the number of images whose distance scores are smaller or equal to that of p. This particular method involves computing the image ranks with respect to each of the different query points. The median of those ranks becomes the final rank assigned to that image. The choice of the median over the mean confers robustness against hugely deviating individual ranks. If an image has a large distance to only a small minority of the query points, these will have no effect on the final rank of that image. This is desirable as it provides support for more disjunctive queries, but the method is equally robust against unusually small distances. If an image is close to one query point and very large to all others, its high rank achieved in one comparison is unlikely to have an effect on the final rank although the natural interpretation would arguably be that the image happens to fall into one of the natural groups of which the query point is a representative. Median-rank aggregation should therefore work particularly well when the relevant images are spread out in feature space without necessarily forming distinct clusters so that relevant images might also be expected in the spaces between query-points.

A general note of caution must be made regarding the conclusions that can be drawn from a comparison of different aggregation methods. The weighted average of Ishikawa et al. (1998), Porkaew et al. (1999) and Rui and Huang (2000) might result in similar performance as a disjunctive method if the relevant images that are returned belong to only one of the clusters even if the relevant images as a whole may exhibit a rich cluster structure. This, however, is not unlikely if the initial set of images tend to come from the same region in feature space.

2.2.2 Distance metric optimisation

Rationale

As we have noted in Chapter 1, one of the challenges of similarity search relates to the question of how to combine features to arrive at an overall measure of similarity between two images. Given some set of features, each represented as a real-valued vector, arguably the simplest distance method involves fusing all individual feature vectors into one by concatenating them. The overall distance between two images is then simply the distance between two feature vectors x and y. In hierarchical models, distances are computed between individual features and the resulting distances are aggregated. In both models, we have to compute the distance between two vectors. One natural distance metric in a vector space is the cosine distance

$$D(x, y) = 1 - \frac{x \cdot y}{|x||y|},$$

which is a measure of the angle between x and y and has proven effective for comparing term vectors in text retrieval. In image retrieval, commonly used distance metrics are instances of the general Minkowski metric,

$$D(x, y) = \left[\sum_i |x_i - y_i|^\alpha \right]^{\frac{1}{\alpha}}, \quad \alpha > 0.$$

This reduces to the Euclidean metric for $\alpha = 2$ and to the L_1 metric for $\alpha = 1$. The Minkowski metric can readily be parametrised by adding a weight to each component-wise difference. For $\alpha = 2$, we obtain a weighted Euclidean distance

$$D(x, y) = \left[\sum_i w_i (x_i - y_i)^2 \right]^{\frac{1}{2}}, \tag{2.1}$$

where the weights are commonly constrained to sum to one and to be non-negative. Relevance feedback can now help to adjust these weights so that relevant images tend to be ranked higher in subsequent rounds.

The general idea is illustrated in Figure 2.3 on the same synthetic data set used for Figure 2.1. Under

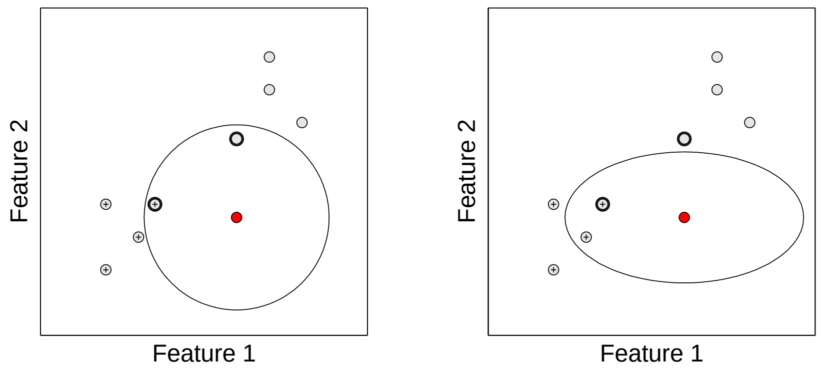


Figure 2.3: Changing parameters of the metric.

the weighted Euclidean metric with equal weights, $w_1 = w_2$, the iso-distance lines in a two-dimensional vector space are circles centred at the query vector. In this example, the one image marked relevant is much closer to the query with respect to the second feature. On the assumption that a relevant image has more relevant images in its proximity, we wish to discount distances along the dimension along which the relevant image differs most from the query. Here this is achieved by decreasing the weight w_1 with the effect that the iso-distance lines become ellipsoids with their long axes parallel to that of the least important feature. In hierarchical models these distances need to be combined. By far the most popular aggregation method is the weighted linear sum that we encountered earlier in the context of multi-point queries,

$$D(x, q) = \sum_{i=1}^{k} w_i d_i(x, q), \tag{2.2}$$

where we now sum over k features rather than over query points. The great majority of relevance feedback methods are concerned with adjusting weights of each individual feature component either in a flat (Ishikawa et al., 1998) or in a hierarchical feature model (Sclaroff et al., 1997; Rui et al., 1998; Schettini et al., 1999; Rui and Huang, 2000; Heesch and Rüger, 2003; Urban and Jose, 2004b). We shall refer to the two types of weights as component weights and feature weights, respectively.

Early models

In the hierarchical model proposed by Rui et al. (1998) the weight of a component is taken to be inversely proportional to the standard deviation of that component among relevant images. This heuristic is based on the intuition that a feature component which shows great variation among the relevant images does not help to discriminate between relevant and non-relevant images. Although any function that monotonically decreases with the variance would appear to be a good candidate, it turns out that dividing by the standard deviation agrees with the optimal solution that was later derived in the optimisation framework of Ishikawa et al. (1998). The feature weights are adjusted by taking into account both negative and positive examples. To illustrate how important simple heuristics were in the methodology of early image retrieval systems, we shall describe the method in some more detail. Relevance feedback takes the form of an integer number which the user assigns to each of the top N images. This relevance score is used to update feature weights as follows: We begin by establishing the N most similar images for each of the k features individually. For each feature we then determine which of these N images is also among the N most similar images overall and set its weight to the sum of the relevance scores of those images. The feature weight is thus the sum of relevance scores of all those images in the intersection between two ranked lists. The technique is motivated by the intuition that a feature that retrieves many of the images that have been labelled as relevant should be assigned a higher weight. Although the experimental results suggest a substantial improvement in retrieval performance on a database containing more than 70,000 images the figures should be treated with great caution as the number of images on which relevance feedback is given lies in the somewhat unrealistic range of 850 to 1100.

In the flat model of Peng et al. (1999) the distance between two feature vectors is computed according to Equation 2.1. The weights w_i are some monotonically increasing function of the 'relevance' of the ith component, defined as

$$r_i(q) = \mathrm{E}[f|x_i = q_i],$$

where $f\colon \mathbb{R} \rightarrow [0, 1]$ is a regression function trained on positive and negative examples provided by the user. The relevance, therefore, is the average value of f (over all x) given that the ith component is that of the query. Imagine as an extreme case that all positive examples share the same value in one component as the query while the value of the same component varies freely among the negative examples. The component is quite evidently informative about the relevance of an image and the expected value of f conditional on that value should be comparatively large and close to one.

Optimising generalised Euclidean distances

An elegant generalisation of the relevance feedback method of Rui et al. (1998) was developed by Ishikawa et al. (1998). Motivated by the observation that relevant images may not necessarily be aligned with one of the feature dimensions, the weighted Euclidean distance used in Rui et al. (1998) cannot fully account for their distribution. This can be rectified by considering a generalisation of the Euclidean distance, introduced by Chandra Mahalanobis under the name D-statistic in the study of biometrical data and now simply known as the Mahalanobis distance. It is more conveniently written in matrix notation as

$$D(x, q) = (x - q)^T M(x - q),$$

where M is a square matrix. If M is a diagonal matrix, then the expression reduces to Equation 2.1 with the weights w_i corresponding to the diagonal elements. If M is a full matrix, then the expression contains products between the differences of any two components. In two dimensions with

$$M = \begin{bmatrix} a & b \\ c & d \end{bmatrix}$$

this writes as

$$D(x, q) = a(x_1 - q_1)^2 + (b + c)(x_1 - q_1)(x_2 - q_2) + d(x_2 - q_2)^2,$$

and similarly for higher dimensions. The iso-distance lines of the general Mahalanobis metric are ellipsoids that do not need to align with the coordinate axes. The components of M are found by minimising the sum of the distances between the images marked relevant and the query. The interesting twist of the model is that the query itself is re-estimated at each step. The optimisation thus determines not only M but also the query with respect to which the distances are minimised. The method combines the two techniques of query point moving and metric update in one optimisation framework. The objective function is

$$\min_{M,q} \sum_{i=1}^{N} v_i (x^i - q)^T M(x^i - q), \tag{2.3}$$

where N is the number of relevant images and v_i a relevance score given by the user. Note that x^i here denotes the vector of the ith image, not the ith component of some vector x. Under the additional constraints that $\det(M) = 1$ and M is symmetric the solutions for q and M are

$$q = \frac{\sum_{i=1}^{N} v_i x_i}{\sum_{i=1}^{N} v_i}$$

and

$$M = [\det(C)]^{\frac{1}{n}} C^{-1},$$

where C is the covariance matrix of the positive examples. In order for C^{-1} to exist, relevance feedback needs to be given on at least as many images as there are feature components. If this is not the case, a pseudo-inverse can be used instead of C^{-1}. The singular value decomposition (SVD) of C is $C = A\Lambda B^T$, where Λ is a diagonal matrix: $\text{diag}(\lambda_1, \ldots, \lambda_K)$. Each λ is either zero or positive. If there are L non-zero λs, the pseudo-inverse of C is

$$\begin{aligned} C^+ &= A\Lambda^+ B^T \\ \Lambda^+ &= \text{diag}\left(\frac{1}{\lambda_1}, \ldots, \frac{1}{\lambda_L}, 0, \ldots, 0\right). \end{aligned}$$

Although the method is being applied successfully (Urban and Jose, 2004a), when dealing with large, sparse image features as they commonly occur in image retrieval, performances have been reported to be rather unstable (Rui and Huang, 2000).

Based on our preceding discussion, some of the limitations of the approach taken by Ishikawa et al. (1998) should be evident: First, the approach tackles the problem of query point moving but does not support multi-point queries; secondly, it exploits only positive feedback which might be rather scarce at the beginning of the search; thirdly, it assumes a flat image representation model with all features for one image concatenated into one single vector. This inflates the number of parameters to be learned with the effect of rendering parameter estimation less robust.

To address the last shortcoming, Rui and Huang (2000) extend the optimisation framework of Ishikawa et al. (1998) by adding feature weights. For each feature, distances are computed using the generalised Euclidean metric and the overall similarity is obtained according to Equation 2.2. As in Ishikawa et al. (1998) the aim is to minimise the summed distances between relevant images and the query. The objective function takes the form of Equation 2.3 except for an additional inner sum,

$$\min_{M,q,w} \sum_{i=1}^{N} v_i \sum_{j=1}^{k} w_j (x_{ij} - q_j)^T M (x_{ij} - q_j),$$

where, as before, v are relevance scores, w_j is the weight of the jth feature and x_{ij} is the jth feature vector of the ith relevant image. The optimal solutions for q and M are the same as in Ishikawa et al. (1998) while the feature weights are given by

$$w_j \propto \frac{1}{\sqrt{\sum_{i=1}^{N} v_i d(x_{ij}, q_j)}},$$

where the denominator is the sum of the weighted distances between the query and all relevant images under feature j.

Optimisation with negative feedback

The choice of the objective function in Ishikawa et al. (1998) and Rui and Huang (2000) appears quite reasonable: minimising the summed distances of the new query to all relevant images should indeed improve retrieval performance somewhat if only because the same relevant images that were retrieved in the first round are ranked higher. But it is clearly not a necessary choice. Note in particular that the objective function does not take into account negative feedback which has been shown to prevent the retrieval results from converging too quickly towards local optima (Vasconcelos and Lippman, 2000; Heesch and Rüger, 2002; Müller et al., 2002).

Depending on the choice of the objective function, making allowance for negative feedback might prevent us from finding analytic solutions. Aggarwal et al. (2002), for example, adopt the general framework of Ishikawa et al. (1998) and minimise Equation 2.3 subject to the additional constraint that there are no non-relevant images within some ϵ-neighborhood of q. This is achieved by automatically modifying the relevance scores v_i. In particular, given some solution q and M of Equation 2.3 with an initially uniform set of relevance scores, the relevance score of the relevant image that is farthest from the current query point q is set to zero and the scores of any other positive images are set to the sum of their quadratic distances from the negative examples. Minimising the objective function again with the thus altered scores yields a new solution q and M which is likely to contain more relevant images. This scheme is iterated until the ϵ-neighborhood contains only relevant images.

Another example of metric optimisation involving negative feedback is Lim et al. (2001). Users are asked to re-rank retrieved images and the system subsequently minimises the sum of the differences between the user-given ranks and the computed ranks. Because of the integral nature of ranks, the error function is not analytic and numerical optimisation is required to find the feature weights.

A method very similar to Lim et al. (2001) that admits to an analytic solution is proposed in Heesch and Rüger (2003). Relevance feedback is given by positioning images closer to or further away from the query situated at the centre. The user provides a real-valued vector of new distances and the objective function is the sum of the squared errors between the distances computed by the system and the new distances supplied by the user.

Multi-dimensional scaling

Ideas about navigating more flexibly through image collections find their first significant articulation in Rubner et al. (1997). The starting point is the observation that current retrieval systems present returned images according to their distances to the query and ignore altogether the information about the mutual distances within the set of returned images. Returned images that are visually similar may not necessarily be displayed close to each other. The proposed solution involves multi-dimensional scaling (Kruskal, 1964). Given a set of objects and their mutual distances, we can place each object in a high-dimensional metric space such that the distances are exactly preserved. For practical purposes, the preferred dimensionality of the space is two for which distances can only be approximated. The result is an approximate two-dimensional embedding that preserves, as far as possible, the distances between objects. The technique can be applied to the set of retrieved images but can also be used as a means of displaying the entirety of small collections in a perceptually meaningful way. Users navigate through a collection by selecting one of the retrieved images as the new query, so that browsing ultimately takes the form of iterated search.

Another attempt of a synthesis between automated search and browsing is described in Santini and Jain (2000) and Santini et al. (2001). Similar to Rubner et al. (1997), the proposed system seeks a distance-preserving projection of the images onto two dimensions. In addition to selecting an image from the display as the new query, users may also alter the mutual distances between images. In Santini and Jain (2000), the system then finds feature and component weights in a hierarchical feature model that minimise the mismatch between the relations imposed by the user and the computed distances.

Manifolds and non-metric distances

All methods described in this section compute distances using a global metric. On the assumption that images fall on some manifold in the feature space He et al. (2004) propose to approximate the metric structure of the manifold at the location of the query. The approximation makes use of positive examples, which are assumed to be close to the query under the geodesic distance. The algorithm proceeds by computing the k-nearest neighbours of each of the positive examples. The union of these sets constitutes the set of candidates that may potentially be retrieved. The geodesic distance is approximated by the topological distance on a graph whose vertices correspond to elements of the candidate set along with the query and the positive examples. Edges are constructed between any two images if their unweighted Euclidean distance does not exceed some threshold. The geodesic distance is then approximated by the topological distance on the graph, that is, the length of the shortest path between two images. Retrieval on the manifold returns the set of images with small topological distances to the query.

Li et al. (2002) provide empirical evidence on a synthetic dataset for a simple and very reasonable hypothesis: images that are relevant to some query may resemble the query with respect to different features. The important conclusion they draw is that learning one set of weights, for example of a weighted Minkowski metric, is insufficient for retrieval purposes. They suggest to rectify this by introducing what they call dynamic partial distance functions. Given k-dimensional image vectors p and q, they define

$$\delta_i = |p_i - q_i|, \quad i = 1, \dots, k,$$

and a set Δ_m containing the m smallest δ_i, where $1 \leq m \leq k$. The dynamic partial distance function is then defined as

$$d_{m,r}(p, q) = \left(\sum_{\delta_i \in \Delta_m} \delta_i^r \right)^{\frac{1}{r}}.$$

For $m = k$, the distance reduces to the Minkowski metric. For $m < k$ the technique dynamically selects different features for different pairs of objects. As a result of not employing all features the triangle inequality ceases to hold, i.e. one can conceive of situations where $d(p, q) + d(q, r) < d(q, r)$. For $m < k$ the function thus ceases to be a metric. Qamra et al. (2005) apply these functions successfully to the problem of detecting near-duplicates in large image collections.

2.2.3 Image search as classification

A third class of techniques treats the problem of similarity search as one of classification. Some of these can be interpreted as estimating parameters of some similarity function and are thus similar to the class of metric optimisation techniques described in the preceding section.

Probabilistic approaches

Methods that approach the classification problem from a Bayesian perspective explicitly model probability densities. The aim of these methods is to assign class probabilities to an image based on the class-specific feature densities estimated from relevance feedback. Let p be an image, x its feature representation and R and N be the sets of relevant and non-relevant images. By Bayes' rule we have

$$P(p \in R|x) = \frac{P(x|p \in R)P(p \in R)}{P(x)}.$$

In Nastar et al. (1998) the feature density of relevant images $P(x|p \in R)$ is assumed to be Gaussian and features are assumed to be independent so that $P(p \in R|x)$ is a product of Gaussians,

$$P(p \in R|x) \propto \prod_{i=1}^{k} P(x_i|p \in R).$$

If we only consider relevant examples, the mean and standard deviation can readily be found using the principle of maximum likelihood. Nastar et al. (1998) suggest an iterative technique that also takes into account negative examples. It does this by determining the proportion of negative examples falling into a 3σ confidence interval around the current mean and the proportion of positive examples falling outside of it. The error is simply the sum of the two terms. To better account for multi-modality a mixture of Gaussians can be used, an extension that has the slight disadvantage of requiring numerical optimisation for parameter estimation (Vasconcelos and Lippman, 2000; Yoon and Jayant, 2001).

Meilhac and Nastar (1999) drop the strong assumption of Gaussianity of the feature densities and use a Parzen window for non-parametric density estimation. Unlike in Nastar et al. (1998) feature densities are now estimated for both relevant and non-relevant images and the decision rule is

$$I(x_i) = -\log[P(x_i|p \in R)] + \log[P(x_i|p \in N)]$$

for each feature. Assuming independence of features we obtain

$$I(x) = -\sum_{i=1}^{k}(\log[P(x_i|p \in R)] - \log[P(x_i|p \in N)]).$$

The additiveness of this density estimation method makes it incremental, i.e. at every feedback round a fixed number of terms is added to the decision function.

The Bayesian framework developed by Cox et al. (1998, 2000) for target search is based on an explicit model of what users would do given the target image they want. The system relies on Bayes' rule to predict the target given their action. This is achieved by updating a probability distribution over possible image targets. An entropy minimising display algorithm is developed that attempts to maximise the information obtained from a user at each iteration of the search. This is one of the first studies of relevance feedback that breaks with the convention of displaying the most relevant images. While substantially reducing the number of iterations required for target search in comparison with the standard display method, it does not scale well to category search and does not support the selection of more than one image per iteration.

Discriminant classifiers

An alternative approach to classification that does not require an explicit modelling of feature densities involves finding a discriminant function that maps features to class labels using labelled training data.

An increasingly popular classifier is the support vector machine or SVM (Vapnik, 1995). SVMs typically map the data to a higher-dimensional feature space using a possibly non-linear transform associated to a reproducing kernel. Linear discrimination between classes is then attempted in this feature space. SVMs have a number of advantages over other classifiers making them particularly suitable for relevance feedback methods (Hong et al., 2000; Chen et al., 2001; Tong and Chang, 2001; Jing et al., 2003; Crucianu et al., 2004; He et al., 2004): (i) the decision function of an SVM allows images to be ranked according to relevance; (ii) for most choices SVMs avoid too restrictive distributional assumptions regarding the data; (iii) SVMs are flexible. For example, prior knowledge about the problem can be used to choose and tune the kernel; (iv) SVMs allow fast learning on the small number of examples provided through feedback and also fast classification for collections of medium size.

In the context of image retrieval the training data consist of the relevant and non-relevant images marked by the user. Learning classifiers reliably on such small samples is a particular challenge. One potential remedy is that of active learning (Cohn et al., 1994). The central idea of active learning is that some training examples are more useful for training the classifier than others. It is guided by the more specific intuition that points close to the hyperplane, that is, in regions of greater uncertainty regarding class membership, are most informative and should be presented to the user for labelling (instead of a random subset of unlabelled points). An application of active learning to image retrieval is proposed in Tong and Chang (2001) where an SVM is trained over successive rounds of relevance feedback. In each round the system displays the images closest to the current hyperplane. Once the classifier has converged, the system returns the top k relevant images farthest from the final hyperplane. Although the method involves the user in several rounds of potentially ungratifying feedback, the performance of the trained classifier improves over that of alternative techniques such as query point moving and query expansion.

A very similar method with improved retrieval performance is developed by He et al. (2004) based on the observation that the expected misclassification of unseen images is minimised if we let the user label the image that results in the greatest reduction of the margin, which is not generally the image closest to the hyperplane.

2.2.4 An interlude

We do by no means contend that our exposition of relevance feedback methods is exhaustive. However, the works we have decided to pass over do not introduce fundamentally new ideas. The interested reader is referred to Zhou and Huang (2003) for a slightly more comprehensive, albeit less detailed overview of earlier relevance feedback methods and Crucianu et al. (2004) for a more recent update with an emphasis on classification approaches. A detailed survey of relevance feedback strategies in the context of document retrieval is by Ruthven and Lalmas (2003). Much of what is said there applies equally to image retrieval.

Let us now take a step back and evaluate the merit of relevance feedback techniques as described in the preceding sections. The reported performance gains are often considerable and suggest that relevance feedback has rightfully established itself as a core element of state-of-the-art retrieval methodology. However, any claims regarding performance have to be judged carefully against the experimental particulars, especially the collection size, the performance measures, and the difficulties of the queries. Often collection sizes are relatively small and rarely exceed several thousand images with categories being unrealistically uniform. Popular performance measures from information retrieval such as mean average precision are sensitive to small changes in the position of top-ranked images and an increase in performance under such measures need not translate into any tangible gain for the user.

It is also noteworthy that image retrieval has yet to find its way out of the research environment. That it has not could be taken as circumstantial evidence that current methods may not sufficiently cater for the requirements of real users. Below we describe what we believe are two major shortcomings of current approaches.

Parameter initialisation

The utilisation of relevance feedback for query expansion and multi-modal density estimation has attracted much attention and appears justified on the ground that the feature distributions of most relevance classes tend indeed to be multi-modal forming natural groups in feature space. But unless the query itself consists of multiple images representing these different groups, we should not reasonably expect images from different groups to be retrieved in response to the query. Rather, the retrieved images will at best contain images from the cluster to which the query image is closest under the current metric.

But not only do relevance classes tend to form visually distinct clusters, images often belong to a number of relevance classes. This is an expression of the semantic ambiguity which pertains in particular to images and which relevance feedback seeks to resolve. With queries consisting of single images, the principal question is which class the query belongs to, not so much what the different natural groups are into which the class decomposes. But while some systems cater for multi-modality, none explicitly deals with polysemy. By fixing parameters for the first retrieval, systems effectively impose a particular semantic interpretation of the query. We will argue that a more profitable approach consists in exposing the various semantic interpretations an image may admit to *prior* to the first retrieval.

Parameter initialisation is particularly important when the proportion of relevant images in the collection is small. Users need to have assembled a suitably large set of relevant images in the first place and if the first set of retrieved images is largely non-relevant, most systems do not generally allow users to move to other areas in the image space. In fact, it is well known of metric optimisation techniques that they tend to converge to local optima (Aggarwal et al., 2002; Heesch et al., 2003). This problem can be alleviated by considering negative examples as they tend to have a major, albeit somewhat haphazard, effect on the parameters thus ensuring a continuous influx of yet unseen images (Müller et al., 2000; Vasconcelos and Lippman, 2000). Because the navigation thereby achieved is difficult to control, negative examples on their own prove inadequate, however, for finding images in large collections.

The problem of parameter initialisation has so far received insufficient attention. One notable exception is the work by Aggarwal et al. (2002) that we had mentioned earlier in a different context. The system segments the query image, modifies each segment in various ways and asks users to mark out those modified query images which they would continue to regard as relevant. The feature weights are then computed as in Rui et al. (1997) by considering the variance among the relevant segments. The primary motivation of the method is to reduce network load in client-server environments by gathering relevance feedback before any images are retrieved. The side-effect is that the first set of retrieved images is more likely to contain relevant images and that, therefore, subsequent rounds of relevance feedback are more effective. The method provides a possible solution to the problem of parameter initialisation. Its downside is its dependence on good segmentation. This is not only difficult to achieve on general image collections but also carries a non-negligible computational cost. Clearly more work in this direction is required. The first application of our NN^k idea, which we develop in Chapter 3, will be to this problem of parameter initialisation.

Exploratory search

With very few exceptions, the methods described above rely on the assumption that users know what they are looking for. The methods are designed to allow users to home in on a set of relevant items within a few iterations and do not support exploration of the image collection. Although some of the methods described above, most notably Rubner et al. (1997) and Santini and Jain (2000), aim to provide alternatives to the conventional relevance feedback methods, even these, upon closer examination, amount to query by example. What distinguishes them from other approaches is chiefly the utilisation of the mutual relationships between retrieved images as a source of feedback. They do not make provision for more exploratory interaction of the collection and fail to satisfy users who are less certain about their information needs. We shall see in the next section that more flexible interaction models have been developed over the last couple of years that address this issue more successfully.

2.3 Search through browsing

Browsing environments offer an alternative to the conventional method of query by example. That browsing has not quite received the same attention as query by example may have several reasons: Browsing appears to shift the burden of retrieval back to the user leaving to a large extent unexploited the computational resources computers can offer. Also, browsing does not seem to require much computer intelligence and thus invites being dismissed as being rather primitive, and less effective than query by example methods when dealing with large volumes of data. We hope to show in this section that these views are largely ill-founded and that browsing has a much underestimated potential. Some of the immediate advantages of browsing over query by example are suggested below.

2.3.1 In defense of browsing

Mental image

The query in content-based image retrieval typically takes the form of an example image. Although it could be argued that this is a reasonable way of articulating information needs, images may not be at hand as easily as keywords. Often users may not have a suitable example query at their immediate disposal and would need to access an image collection first to obtain an example image with which to query. In order to browse image collections, a mental representation of the query is all that is needed to guide the search process.

Recent progress in automated image annotation (Lavrenko et al., 2003; Feng et al., 2004; Yavlinsky et al., 2005; Zhang et al., 2005) suggests that visual search may soon be reduced successfully to traditional text retrieval. Note, however, that images, whether in our mind or not, are inordinately more expressive and can capture what words can not. Thus, to find what words cannot express, a visually guided search could still be the more promising route. Automated annotation, meanwhile, could provide a method for identifying reasonable starting points from where to begin the browsing process.

Developing information need

Retrieval by example image presupposes that users can formulate their information needs pictorially, and thus it assumes that users have an information need in the first place. This may often not be the case. Rather than being well-defined from the beginning, the information need may instead develop in the course of, and as a result of, the particular interaction which users engage in. In such situations enabling users to gain an overview of the collection and to navigate quickly between different regions of the image space becomes of much greater importance. Even sufficiently accurate automated annotation cannot on its own provide such flexible interaction with a collection.

Responsiveness

Much of the research in content-based retrieval is concerned with improving retrieval performance. For large collections, time complexity becomes as important an issue. The time complexity that governs typical retrieval methods is linear in the number of images. Even when hierarchical indexing structures are used, performance has been shown to degrade rapidly in high-dimensional feature spaces (Weber et al., 1998). For particular relevance feedback techniques, approximative methods may be developed that exploit correlations between successive nearest neighbour searches (Wu and Manjunath, 2001), but there does not exist a universal cure.

Contrast this again to browsing. The structure into which a collection is cast for browsing may be precomputed. Provided the structure can be stored in ways that support efficient access, interaction can be very fast and, for all practical purposes, can be made independent of the size of the collection.

Exploiting the cognitive abilities of users

The ability of the human visual system to recognise patterns reliably and quickly is a marvel yet to be fully comprehended. Endowing systems with similar capabilities has proven an exceedingly difficult task. A similar observation can be made with respect to other areas that have traditionally been marked out as the territory of artificial intelligence. Given our limitations in understanding and emulating human cognition, the most promising way to leverage the potential of computers is to combine their strengths with those of users and to achieve a synergy through interaction.

During browsing users are continuously asked to make decisions based on the relevance of items in relation to their current information need. A substantial amount of time is spent, therefore, by engaging users in what they are best at, while exploiting computational resources to render interaction fast.

2.3.2 Hierarchical structures

The extent to which these advantages can be exploited in practice depends crucially on how we structure the collection for browsing. While the actual process of browsing may seem trivial operationally, effective browsing requires intelligent organisation of the collection's content.

A common method of structuring data is that of hierarchical clustering (Yeung and Liu, 1995; Zhang and Zhong, 1995; Yeung and Yeo, 1997; Chen et al., 1998, 2000; Pečenović et al., 2000; Barnard and Forsyth, 2001; Benitez and Chang, 2003). Navigation in the resulting tree structure takes the form of moving up and down between different levels. An early application to image retrieval is by Yeung and Liu (1995) and Yeung and Yeo (1997) who study clustering methods for organising video key frames. Zhang and Zhong (1995) apply the self-organizing map (SOM) algorithm to arrange a collection of images on a two-dimensional grid which is subsequently aggregated into a hierarchy.

A more efficient but otherwise similar method is proposed in Chen et al. (2000). Images are first clustered into a binary tree based on pairwise similarities. The binary tree is subsequently transformed into a quad tree. Users can view different levels of the quad tree in an optimised two-dimensional layout with the content of any one cluster indicated by a representative image from that cluster. Browsing takes place by selecting images from the display resulting in a close-up view of the corresponding cluster. Since the structure is precomputed, the computational cost incurred at browsing time is slight. A similar structure is proposed by Benitez and Chang (2003). Using both textual annotation and visual features, a pyramid of concept networks is constructed using perceptual and semantic knowledge extracted from the data. Browsing takes place between automatically identified concepts rather than between the images themselves.

2.3.3 Networks

An alternative structure is that of image networks. Unlike trees, networks need not be acyclic. By not restricting connections to particular strata of a hierarchy, networks can potentially support more flexible navigation.

Pathfinder networks

An increasingly popular type of networks for both information visualisation and browsing are pathfinder networks (Schvaneveldt, 1990). These are constructed by simplifying a potentially much more complex network through the removal of particular edges. In Fowler et al. (1992) pathfinder networks are used to structure the relationships between terms from document abstracts, between document terms and between entire documents. The user interface supports access to the browsing structure through prominently marked high-connectivity nodes. An application of pathfinder networks to image retrieval is found in Chen et al. (2000), where pathfinder networks are constructed and compared with three different classes of image features using the similarity between images as the edge weight.

Nearest neighbour networks

A major and relatively early work on interlinked information structures is that by Croft and Parenty (1985). The paper is concerned with the question to what extent the representations and algorithms of sophisticated document retrieval systems can be applied to database systems. For the purpose of comparison, the authors propose one such document retrieval system. It involves the representation of the documents as a network of document and term nodes with weighted connections between each type of node: Document-term links indicate how important a term is in the representation of a document; term-term links indicate how closely two terms are related; and document-document links indicate the similarity of the content of the documents. The main advantage of the network structure is its ability to capture a wealth of information about the document content and thus to support a greater variety of search strategies than conventional representations. Instead of representing the similarity between any two documents, the authors suggest keeping only links between a document and the document most similar to it, and similarly for terms. Term-term and document-document links thus connect nearest neighbours. Each document can then be thought of as giving rise to what they call star clusters comprising the document itself and those documents it is connected to through nearest neighbour links. Croft and Parenty realise that the structure could support interactive search: "As well as the probabilistic and cluster-based searches, the network organisation could allow the user to follow any links in the network while searching for relevant documents. A special retrieval strategy, called browsing, could be based on this ability." (p. 380) An illustration of the resulting network structure is given in Figure 2.4, reproduced from the original paper. Because each document usually has only one nearest neighbour, the resulting network is relatively sparse. The number of document-document edges does not exceed by much the number of documents and clusters are disconnected rendering browsing along document-document nodes alone impractical.

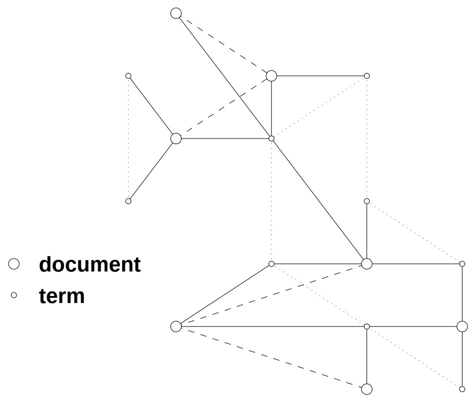


Figure 2.4: Croft and Parenty's nearest neighbour networks

The idea of nearest neighbour networks was taken up in an insightful paper by Cox (1992) and developed more fully in Cox (1995). Cox challenges the prevailing view of the time that browsing is merely an adjunct to query by example and proposes proximity or nearest neighbour networks as a suitable data structures for efficient and effective browsing. Particularly relevant to our work is his idea of multi-modal browsing as it is most akin to the rationale of NN^k networks. Cox envisages constructing individual proximity networks for different criteria and interlinking these. "For example in a document browsing system, we would have networks of words, perhaps similar to WordNet. We would have networks with similarities based on citations, similarities based on words and perhaps similarities based on the history of retrieval. I see the user having many interlinked networks through which to browse." (Cox, 1995, p. 55–56). The idea is applied to a collection of simple drawings that have associated attributes like the number, the shape, the size and the positions of the objects contained in the drawing but it is clearly appreciated that the networks could be used for any objects between which we can measure similarity.

Interconnected nearest neighbour networks were not pursued further in the image retrieval community. For one because the system Cox developed was little more than a proof of concept. Secondly, the paper falls in a time when content-based retrieval was still in its infancy. The first research programme that would grow out of the initial phase of exploration was that of query by example leaving little room for alternative approaches. It is no coincidence, therefore, that the first attempts to introduce browsing into image retrieval were made within the context of query by example.

2.3.4 Dynamic structures

The crucial question pertaining to precomputed graph structures in general is how to design them such that they incorporate our uncertainty regarding the optimal combination of features. In all of the above applications, the similarity metric used for graph construction is essentially fixed.

The ostensive model proposed by Campbell (2000) is conceptually similar to the model by Rubner et al. (1997). The most obvious difference is that in the former images are represented as term vectors and are retrieved by text. The model has since been extended to deal with visual features (Urban et al., 2003). Instead of displaying the results of a query through multi-dimensional scaling, images closest under the current feature set are displayed in a fan-like pattern to one side of the query image. Users can select an image from the retrieved set which is placed in the centre with a new set of images being retrieved in response. Since previous images are kept on the display the visual impression of the user is that of establishing a browsing path through the collection. Its innovative part is the relevance feedback method that guides the browsing process. When an image is selected during browsing, the system attempts to determine the optimal query vector given that image and the sequence of past images. The importance of an image for estimating the weight of a term depends monotonically on its relative position along the browsing path. More recent images are more indicative of the current information need. In Urban et al. (2003), for example, image content is described in the form of a colour histogram vector c. Given a browsing history consisting of a sequence of images with feature vectors $r_1, r_2, \ldots, r_n$ (with r_1 corresponding to the most recent image), the colour representation of the new query is given by $\sum_{i=1}^{n} w_i c_i$ with $w_i = 2^{-i}$: more recently selected images (small i) are thus given a greater weight. The ostensive model, therefore, attempts to track changing information needs and optimise the query accordingly. Operationally, and much like Rubner et al. (1997) and Santini and Jain (2000), ostensive browsing is iterated query-based image retrieval. Which set of images are retrieved depends on which path the user has traced out to arrive at the current point. Because the number of such different paths grows quickly with the size of the image collection, it is impractical to precompute a global structure along which to browse. It is only possible to establish and extend through browsing a local structure that consists of the browsing path and for each node of the new set of retrieved images. Nonetheless, for the user the impression is one of navigating in a relatively unconstrained manner through the image space. Unlike other systems, users do not have to rank or label images, or change their relative locations. The interaction is thus light and effective.

2.4 Conclusions

This chapter has provided a detailed overview of existing relevance feedback techniques in content-based image retrieval as well as a description of different ways of organising image collections for the purpose of browsing.

Query by example techniques assume a precise information need and incorporate relevance feedback to automatically improve the query representation or the similarity metric so that users may home in on the target category within a few iterations. A principal limitation that prevents these relevance feedback techniques from scaling to larger collections is not only the linear time complexity of nearest neighbour search but also their reliance on positive examples of which there may be none when the target category is small. It becomes of pivotal importance, therefore, that search parameters are initialised intelligently prior to the first retrieval, a problem that has so far received scant attention.

Browsing structures support more exploratory interaction and have the advantage that they may be precomputed so that user interaction can potentially be very fast. The structures can typically be expressed as graphs which may take the form of unrooted directed or undirected networks such as pathfinder networks, nearest neighbour networks or NN^k networks, or simply trees obtained, for example, through hierarchical clustering. Browsing takes place by moving between vertices of the graph, which in the case of hierarchical structures involves movement between different levels. The problem with most of these precomputed structures is their failure to take seriously the fact that image meaning is user-dependent. By assuming a particular form of the distance metric, the set of relationships between images as captured in the browsing structure become immutable and the structure semantically biased. The advantage of fast navigation therefore comes at a price: users are no longer in a position to alter the criterion under which similarity is judged. The structures thus deride the principal tenet that provides relevance feedback techniques with their *raison d'être*. Zhou and Huang (2003) arrive at a similar conclusion when they observe that "the rationale of relevance feedback contradicts that of pre-clustering."

The next chapter will develop the foundation on which we shall first develop a novel technique for relevance feedback in Chapter 4, and then a browsing framework from Chapter 5 onwards that overcomes the limitations of pre-computed structures.

Chapter 3

Introducing NN^k

3.1 Introduction

A fundamental question that has to be addressed in content-based multimedia retrieval is how to infer semantic similarity between objects from their feature representations. The problem arises from the observation that features are generally not equally suited to pick up semantically pertinent commonalities and that the evidence from different features needs to be combined in ways that depend on the query object. The problem is compounded when objects are semantically rich and admit to a number of different interpretations, such as images. As an illustration of this polysemy in images consider Figure 1. Leftmost is a query image depicting an autumnal forest scene. The three images on the right are those closest to the query image under three different features representing, respectively, image texture (left), local colour distribution (middle), global colour content (right). By ignoring colour content altogether the texture feature does not pick up the autumnal character of the query but captures well the distinct texture of leafy vegetation. The global colour feature is insensitive to how colours are distributed locally and ranks an image high if the overall proportions are similar as in a close-up view of fallen leaves. We may view the three images as representing different semantic facets of the query: “tree-like vegetation”, “trees with autumn foliage”, and “autumn colours”. Which of these a user is interested in, and thus which feature is to be given greater weight, cannot be decided on the basis of the query image alone. In information retrieval, the user is the ultimate arbiter of similarity. Since in the context of heterogenous image collections and a varied user community, there are generally no reasons to give priority to any particular features, the most reasonable approach is to

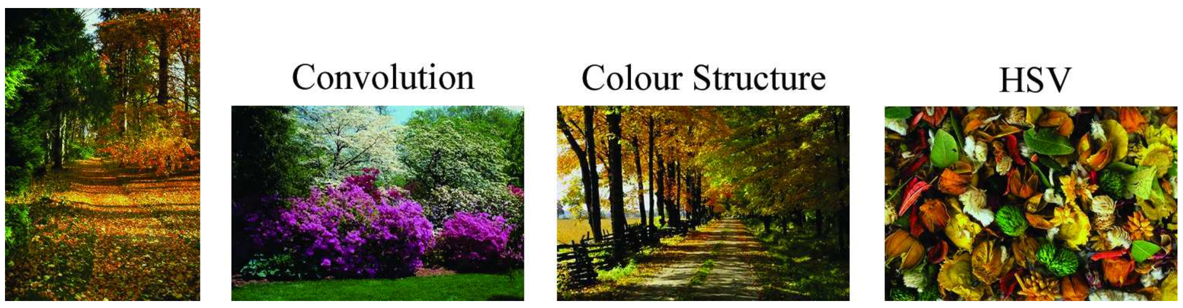


Figure 3.1: Example of an image and its most similar images under different features.

extract and expose the multitude of different semantic facets that can be captured by these features. This chapter formalises the idea: we introduce the idea of an image's NN^k as the set of all images that are most similar to that image under some feature or combination of features. The NN^k provide us with a pictorial representation of the set of image meanings which a user may be interested in. In Section 3.2 we define the set of NN^k and describe a method for determining it. In the course of that section we prove a number of important properties of this set and develop a geometric interpretation of the NN^k in terms of the convex hull of a point set. In Section 3.3 we propose several ways to reduce the computational complexity of NN^k computation.

The idea of NN^k will remain central to all following chapters. It will be used to develop a method of feature weight estimation in Chapter 4 and of image browsing from Chapter 5 onwards.

3.2 NN^k and their determination

3.2.1 Definition and rationale

Definition: An image p is defined to be an NN^k of image q if and only if there exist at least one convex combination of feature-specific distance functions $d_f(\cdot, q)$ for which p has minimal distance to q. Formally, p is an NN^k of q iff

$$\arg\min_i \left(\sum_{f=1}^{k} w_f d_f(i, q) \right) = p, \tag{3.1}$$

for some $w = (w_1, w_2, \ldots, w_k)$ where $w_f \geq 0$ and $\sum w_f = 1$.

For the purpose of future chapters it is useful to give a name to the image whose NN^k we determine. We shall subsequently refer to such an image as the *focal* image, and as being *focal with respect to* its NN^k. We use the term NN^k both for singular and plural but shall often write "set of NN^k" to denote plurality.

Because the set of NN^k comprises all images which are closest to a focal image under some feature combination, it can be thought of as a representation of the different semantic facets of the focal image. Not all features will be equally well suited for capturing these, and some NN^k will be semantically unrelated to the focal image. Likewise, we expect some semantic facets to lie outside of the representational scope of the features and thus not to be exemplified by any of the NN^k. The quality of the NN^k clearly depends on the choice of features. Importantly however, if the focal image has a semantic facet that can be captured by any of the features, it is likely to be distilled by computing the focal image's NN^k. We shall see in subsequent chapters that this property of NN^k can be exploited for both image search and efficient image browsing.

3.2.2 Regular discretisation of $\mathbb{W}^k$

Note that in this and the following chapters, we will be using the familiar identification of points and k-dimensional vectors applied to the origin, write w for the point with coordinates $(w_1, w_2 \ldots, w_k)$, and use the concepts of a vector space and of a point set interchangeably, whichever is more convenient.

Because of the condition that all weights are non-negative, the set of all distances generated by the above combination is convex, and it is infinite because weights are allowed to vary continuously. We cannot therefore compute the NN^k for every possible weight combination and need to resort to sampling methods.

Let us denote with $\mathbb{W}^k$ the space of all w that satisfy the convexity constraint. The approximation that naturally suggests itself involves a discretisation of $\mathbb{W}^k$, and the evaluation of the distance function D at a finite number of points in $\mathbb{W}^k$. Computationally most tractable are regular discretisations with a fixed separation between grid points along each axis. Let n be the number of intervals along each of the k axes and let this be the resolution of the grid. The number of grid points along each axis is $n+1$ with the set of grid points given by

$$G = \left\{ \mathbf{w} = \left(\frac{1}{n}\right)\pi \,\middle|\, \pi \in \mathbb{N}^k, \sum_{i=1}^{k} \pi_i = n \right\}.$$

The problem of discretising $\mathbb{W}^k$ thus amounts to the problem of finding all vectors in $\mathbb{N}^k$ whose components sum to the resolution of the grid.

In practice, we do not want to enumerate all grid points in $\mathbb{N}^k$ and check for each whether the convexity condition is satisfied. Rather, we wish to enumerate explicitly only permissible vertices. The pseudocode that achieves this is straightforward and given below.

```
1  void enumeratePi(int mass, int dim) {
2    for( int i=0; i<=mass; i++ ) {
3      pi[dim] = i;
4      if( dim>2 ) {
5        enumeratePi(mass-i, dim-1);
6      } else {
7        pi[1] = mass-i;
8        getNearestNeighbour(pi);
9      }
10   }
11 }
```

The function enumeratePi is initiated with the resolution n and the number of features k. We can think of n as a mass that needs to be distributed in integer portions across k objects or dimensions. This is accomplished in lines 2 and 3 starting with the dimension indexed by dim. The remaining mass mass-i is recursively allocated to the remaining dimensions in line 5. When reaching the last dimension (dim =1), we allocate the remaining mass to it (line 7) and with the newly established vector pi compute the nearest neighbour (line 8). The function getNearestNeighbour() determines the value of the LHS of Equation 3.1 for the particular weight vector specified by pi. Note that the pseudo-code is valid only for dim$\geq$1. Experimentally we find that the number of vertices for which getNearestNeighbour() is called is precisely the magnitude of the set G (which we can find analytically as will be shown shortly).

One would expect the number of permissible vertices to increase rapidly with the dimensionality of $\mathbb{W}^k$ and with the resolution of the grid. Because the vectors π need to satisfy a convexity constraint, however, the cardinality of G turns out to be much less than would be the case for a k-dimensional hypercube, i.e. less than $(n+1)^k$. Table 3.1 shows the exact numbers for varying n and k.

		n							
		0	1	2	3	4	5	·	9
	1	1	1	1	1	1	1	·	1
	2	1	2	3	4	5	6	·	10
	3	1	3	6	10	15	21	·	55
k	4	1	4	10	20	35	56	·	220
	5	1	5	15	35	70	126	·	715
	·	·	·	·	·	·	·	·	·
	10	1	10	55	220	715	2002	·	48620

Table 3.1: Number of vertices as a function of the resolution n and dimensionality k of the grid.

Finding an explicit formula for the number of vertices is a problem of enumerative combinatorics and can be rephrased as follows: Given an integer n, find the number of compositions of n into k parts. Note that we are interested in compositions rather than partitions since the order of the segmentation is important. For example, for $n = 3$ and $k = 2$, the set of all possible compositions is $\{(0, 3), (1, 2), (2, 1), (3, 0)\}$ and thus of cardinality 4 (as given in Table 3.1). If each part of a composition is constrained to be non-zero, the solution is just $\binom{n-1}{k-1}$. The truth of this formula can be seen by interpreting it as the number of distinct ways of choosing $k - 1$ of the first $n - 1$ integers as end points for the k segments dividing $[0, n]$ (Andrews, 2004). In our case, a composition may also contain segments of size zero with the important consequence that n does not constitute an upper bound for k. If $k > n$ at least one of the parts is simply zero. Notice from Table 3.1 the symmetry of $c(n, k)$ about the diagonal $k = n + 1$, i.e. $c(n, k) = c(k - 1, n + 1)$. In fact, upon closer inspection, we find that the numbers along the diagonals $n + k - 1 = i$, $i \geq 1$, correspond to the columns in Pascal's triangle of binomial coefficients. This leads us to conjecture that the solution is given by $\binom{n+k-1}{n}$ since $n + k$ is the length of the diagonal that includes (n, k). This would account for the symmetry since $c(n, k) = \binom{n+k-1}{n} = \binom{n+k-1}{k-1} = \binom{(k-1)+(n+1)-1}{k-1} = c(k - 1, n + 1)$.

Theorem 1. *The number of vectors $\pi \in N^k$ for which $\sum_{i=1}^{k} \pi_i = n$ is equal to*

$$c(n, k) = \binom{n + k - 1}{n}.$$

Proof. We rephrase the problem in terms of paths on a two-dimensional grid. In particular, consider the $k \times (n + 1)$ grid in Figure 3.2 where the rows are labelled from 1 to k, and the columns from 0 to n.

Each path can be associated uniquely with a k-dimensional vector $\pi \in N^k$ as follows: let the ith component of the vector be the step size of the binomial path along the row with label i. For example, the binomial path in Figure 3.2 would correspond to a vector with respective components 1,1,0,1,0. The sum of the vector components is just the width of the rectangle n, as assumed by the theorem. Clearly, any binomial path can in this way be associated uniquely with a vector, and vice versa. Since the relationship is bijective, $c(n, k)$ is just the number of binomial paths from $(0, 1)$ to (n, k).

To find the number of binomial paths, note from Figure 3.2 that

$$c(n, k) = c(n - 1, k) + c(n, k - 1), \tag{3.2}$$

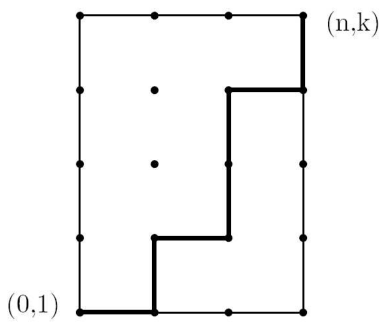


Figure 3.2: $c(n, k)$ equals the number of binomial paths from (0,1) to (n,k).

since every path through (n, k) has to go through either $(n, k-1)$ or $(n-1, k)$ from where there is only one possible path to (n, k). The boundary conditions are $c(n, 1) = 1$ (as there is only one one-dimensional vector whose components add to n) and $c(0, k) = 1$ (as there is only one k-dimensional vector whose components add to zero). Equation 3.2 is a recurrence equation that can readily be solved using the powerful technique of generating functions. Define $A_k(x) = \sum_{n=0}^{\infty} c(n, k)x^n$. Multiplying each side of Equation 3.2 by x^n and summing over n, we obtain

$$\sum_{n=0}^{\infty} c(n, k)x^n = \sum_{n=1}^{\infty} c(n-1, k)x^n + \sum_{n=0}^{\infty} c(n, k-1)x^n,$$

which in terms of our generating function $A_k(x)$ can be written as

$$\begin{aligned} A_k(x) &= xA_k(x) + A_{k-1}(x) \\ &= \frac{1}{1-x}A_{k-1}(x) \\ &= \left(\frac{1}{1-x}\right)^{k-1} A_1(x). \end{aligned}$$

Using the fact that $c(n, 1) = 1$, this is just

$$A_k(x) = \left(\frac{1}{1-x}\right)^{k}. \tag{3.3}$$

By definition of $A_k(x)$, $c(n, k)$ is the coefficient of x^n in the Taylor series expansion of $A_k(x)$ about zero. The Taylor series of the RHS of Equation 3.3 can be found using the very definition of a Taylor series. In particular,

$$\begin{aligned} c(n, k) &= \frac{\partial^n}{\partial x^n}\left[\left(\frac{1}{1-x}\right)^{k}\right]\Bigg|_{x=0} \times \frac{1}{n!} \\ &= k \times (k+1) \ldots (k+n-1)\frac{1}{n!} \\ &= \frac{(k+n-1)!}{(k-1)!\, n!} \\ &= \binom{n+k-1}{n}. \end{aligned}$$

□

Having at our disposal an explicit formula for the number of permissible vertices allows us to control the computational complexity of NN^k determination. Given k, the number of features, we can adjust n such that the number of vertices does not exceed a certain upper bound U, that is, we choose the largest n such that $c(n, k) \leq U$. This guards against the exponential increase that would otherwise ensue from increasing the number of features.

3.2.3 Graphical representation of NN^k

In the case of $k = 3$, the set of permissible vertices lie in the two-dimensional subspace of $\mathbb{R}^3$, the 2-simplex. Since each point of the subspace is associated with exactly one NN^k, the set of NN^k form a partition of the weight space. With the approximation method described above, the boundaries of the partitions can in principle be determined arbitrarily close by increasing the resolution n. Figure 3.3 shows the partitioning for a particular query from the Corel collection, three features and varying resolution.

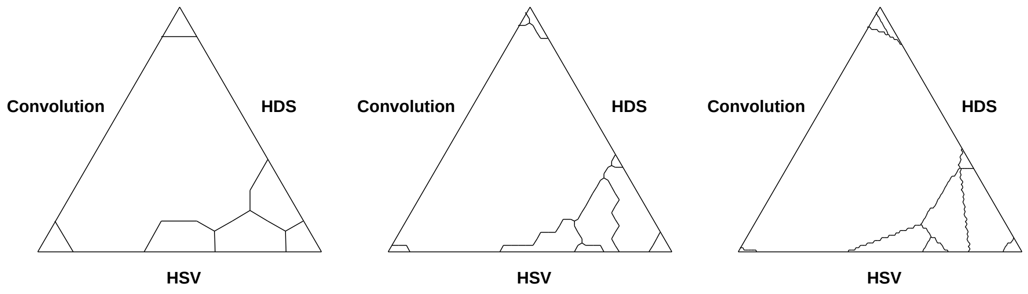


Figure 3.3: Partitioning of the weight space for $n = 5, 10$ and 100.

3.2.4 Iso-NN^k regions and their properties

The partitioning of the weight space into regions possesses some interesting properties which we shall analyse in this section. It is convenient for our subsequent analysis, and indeed for the remainder of the thesis, to make the following definitions:

Definition: An iso-NN^k region is the region in $\mathbb{R}^k$ that contains all the weight vectors for which the image with smallest distance to a particular focal image is the same.

Note that iso-NN^k regions are only open sets if they lie completely within the interior of the set of permissible weight vectors. If they share a boundary with $\mathbb{W}^k$ then they are neither closed nor open.

Definition: The support of an NN^k is the relative volume of its iso-NN^k region.

In practice, we approximate the support as the number of weight combinations for which an NN^k is most similar to the focal image divided by the total number of weight combinations for which NN^k have been determined, i.e.

$$S(p, q) = \frac{|\mathbb{P}|}{c(n, k)},$$

where $\mathbb{P}$ contains all the weight vectors for which the image is the NN^k of the focal image, and $c(n, k)$ is given by Theorem 1.

Looking back at Figure 3.3 and in particular at the approximation for $n = 100$, one is tempted to conjecture that the boundaries between iso-NN^k regions are linear. This is indeed the case: The boundaries between iso-NN^k regions are points for two dimensions, straight lines for three dimensions and planes for four dimensions. The following theorem generalises this to any dimension.

Theorem 2. *Iso-NN^k regions are bounded by $(k-2)$-dimensional hyperplanes.*

Proof. At the boundary of two adjacent iso-NN^k regions, the distances of the corresponding NN^k, say p_1 and p_2, are the same, i.e.

$$\begin{aligned}
\sum_{i=1}^{k} w_i d_i(p_1,q) &= \sum_{i=1}^{k} w_i d_i(p_2,q) \\
\sum_{i=1}^{k-1} w_i d_i(p_1,q) + \left(1-\sum_{i=1}^{k-1} w_i\right) d_k(p_1,q) &= \sum_{i=1}^{k-1} w_i d_i(p_2,q) + \left(1-\sum_{i=1}^{k-1} w_i\right) d_k(p_2,q) \\
d_k(p_1,q) + \sum_{i=1}^{k-1} w_i(d_i(p_1,q) - d_k(p_1,q)) &= d_k(p_2,q) + \sum_{i=1}^{k-1} w_i(d_i(p_2,q) - d_k(p_2,q)).
\end{aligned}$$

Each side of the equation defines a $(k-1)$-dimensional hyperplane and so, unless the hyperplanes are parallel, their intersection will in turn be a $(k-2)$-dimensional hyperplane. □

Note that it is theoretically possible for the two hyperplanes to be the same and therefore for the intersection to be a $(k-1)$-dimensional hyperplane. This would be the case if p_1 and p_2 had the same distance to the query under each feature, a possibility that can be excluded in practice.

Figure 3.3 also provides anecdotal evidence that the approximations of the iso-NN^k regions converge towards a convex set as the resolution of the grid increases. That is, it appears that iso-NN^k regions are convex.

Theorem 3. *Iso-NN^k regions are convex.*

Proof. Let us denote with $R \in \mathbb{W}^k$ one of the iso-NN^k region of some focal image q and let us denote the corresponding NN^k as p. To show that R forms a convex region, we need to establish that for any two points $a, b \in R$, any convex combination of a and b is also in R. The set of convex combinations defines a line with standard parametrisation

$$\begin{aligned}
\Gamma : w(u) &= (1-u)a + ub, \quad u \in [0,1] \\
&= a + u(b-a).
\end{aligned}$$

Let d denote the vector of feature distances between p and q and consider the image of Γ under the function $F : w \mapsto w \cdot d$. The image is a segment of the real line and has parametrisation in terms of u

$$\begin{aligned}
F(\Gamma) : z_p(u) &= \sum_{i=1}^{k} (a_i + u(b_i - a_i))d_i \\
&= \sum_{i=1}^{k} (a_i d_i) + u \sum_{i=1}^{k} ((b_i - a_i)d_i).
\end{aligned}$$

$z_p(u)$ represents the overall distance between p and q and it varies linearly with u. For $u = 0$ and $u = 1$, the NN^k of q is p. Hence, by definition of an NN^k, $z_p(0) < z_r(0)$ and $z_p(1) < z_r(1)$ for all other images r, $r \neq p$. Because $z(u)$ is linear in u, $z_p(u) < z_r(u)$ for all $0 \leq u \leq 1$ and hence p is the NN^k of q for all convex combinations of a and b. □

An immediate corollary of Theorem 3 is that all regions are simply-connected: they have no holes. Another consequence which will be of importance later is that if there exist two regions with the same NN^k, then the two regions cannot be disjoint.

We can summarise the above by noting that for k features, iso-NN^k regions are convex $(k-1)$-dimensional polytopes.

3.2.5 Support as an *a priori* measure of relevance

We have seen how each NN^k p can be associated with a scalar $S(p, q)$ that measures the number of weight vectors for which that image is the NN^k of a given focal image q. Because the NN^k relation is not symmetric, neither is $S(p, q)$. In general $S(p, q) \neq S(q, p)$. In the absence of other sources of evidence it seems reasonable to consider a semantic facet, represented by a particular image, the more relevant the greater the number of weight sets for which the image has a smaller distance to q than any other image. $S(p, q)$ therefore affords us with an intuitive measure of relevance of the semantic facet q represented by p. The relevance measure can be used to rank images. Clearly, this ranking is only partial since it excludes non-NN^k. It is also likely to exclude many, if not most, of the images a user would consider relevant. It is strictly a ranking of different interpretations of the focal image each represented by a particular NN^k.

In somewhat loose analogy to Bayesian inference, the distribution of relevance may be considered a posterior distribution given the query. Further conditioning is achieved by letting the user interact with the system. The final distribution of relevance can be likened to a posterior distribution conditioned on both a query and a user.

3.2.6 Characteristic weight sets

We shall see in the next chapter that once we have computed the NN^k for a focal image, it is desirable to associate each NN^k with a single weight set that acts as a representative of its iso-NNk region. The grid search method affords us with a number of ways to do this. Recall that each NN^k is assigned a volume of the weight space that measures its relevance to the focal image in the absence of other information. This volume is an open and continuous set of k-dimensional points which can be summarised in various ways. A representation that is particularly attractive because of its compactness is that of a characteristic weight set which summarises the set by a k-dimensional point.

Centre of mass

The centre of mass of an iso-NN^k region can be approximated by the arithmetic average of all the weight vectors sampled from that region, i.e.

$$\overline{w} = \frac{1}{|\mathbb{P}|} \sum_{w \in \mathbb{P}} w,$$

where $\mathbb{P}$ denotes the finite set of weight vectors sampled. An interesting and important question is whether the centre of mass is guaranteed to fall inside the region it is supposed to represent. Note therefore that the centroid of any point set lies inside the convex hull of that point set (Benson, 1966). Since the region from which we sample forms, by Theorem 3, a convex set, the centroid is contained in the region. Hence, representing iso-NN^k regions by the centre of mass of its discrete approximation is a method that can be employed in all cases.

Minimum distance weight

By encoding a set of vectors by their arithmetic average, we ignore the additional information that is associated with each weight vector, namely the distance under that weight vector between the query and the corresponding NN^k. This distance is by definition smaller than that of any other image, but it is still expected to vary. Another representative weight vector, therefore, is that w which causes the distance of the NN^k to the query to be smaller than for any other weight vector from that iso-NN^k region. If we chose to retrieve with this weight vector, the image would have the smallest possible

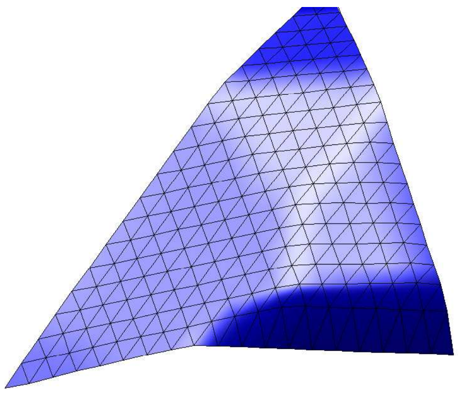


Figure 3.4: Distances across the weight simplex between a query and its NN^k.

distance to the query for all weight sets for which it is also the NN^k. The latter specification is necessary since it may well be the case that the image attains an even smaller distance to the query for other weight sets for which it ceases to be the NN^k. Because the overall distance function is linear in the weights, its extrema are found at the boundaries of the iso-NN^k regions. Figure 3.4 helps to illustrate this. For each weight set of the two-dimensional simplex, we have plotted as the third dimension the distance between a particular query and the NN^k for that weight set. The resulting surface is continuous and consists of a set of intersecting hyperplanes.

3.2.7 Convexity reconsidered

We have until now assumed that the overall distance $D(p,q)$ between two images is obtained as a convex combination of feature-specific distances $d_f(p,q)$. In other words, all components of the weight vector w are non-negative. It is instructive to consider the case where this assumption is relaxed. In particular, we may instead assume

$$D(p,q) = \sum_{f=1}^{k} w_f d_f(p,q), \quad w_f \in \mathbb{R}, \quad \sum_{f=1}^{k} w_f = 1.$$

The resulting set of distances is an affine set and D becomes an affine combination of d_f. This generalisation to affine sets would allow us to assign negative weights to features with the effect of penalising images with high similarity to the query under that feature. An immediate consequence of what appears to be but a trivial generalisation of the convex setting is that there now always exists a weight set under which *any* conceivable image turns out to be the NN^k of any other. The truth of this observation becomes evident when considering the geometric interpretation of distance computation. D is the length of the vector resulting from the orthogonal projection of d onto w. Hence, D is zero if and only if w and d are orthogonal to each other and takes its maximum value when they are parallel. Now, given any d we can always find a $w \in \mathbb{R}^k$ such that $w \perp d$. In fact, there exist an infinite number of w, all lying in the $k-1$ dimensional hyperplane H that includes the origin and is orthogonal to d. If the distance vectors of the N images, $d^1, d^2, \dots d^N$ are not affinely independent, i.e. the $N-1$ vectors $d^2 - d^1, d^3 - d^1, \dots, d^N - d^1$ are not linearly independent, then the orthogonal projections onto H of at least two of them are the same, and thus there does not exist a unique NN^k. The fact that all computations are carried out with finite precision reduces the chances of affine independence somewhat. Since two vectors are dependent only if there exists a particular relationship between each two of their components, the improbability increases with dimensionality, and we can safely assume, that affine independence does indeed hold in practice.

Upon abandoning the convexity constraint, therefore, the set of NN^k ceases to be a privileged subset of the data. For the method to work, convexity appears to be indispensable. This is not quite so, however. If we allowed feature weights to take negative values, every image would be an NN^k, but in the great majority of cases the corresponding weight space would be very small and would contain the trivial solution, i.e. the weight vector that is orthogonal to the distance vector of the image. Only a subset of the images also have non-trivial solutions where the projection of the distance vector is minimal but non-zero. These are exactly those images that are also the NN^k under convexity. Thus, even if we do not stipulate convexity, we can identify a proper subset of the images as being meaningful NN^k. Convexity is not essential but a convenient assumption as it precludes trivial solutions from the outset.

Convexity therefore does not matter, but this is only the case because all distances are non-negative. If they were not, negative weights would allow us to find non-trivial NN^k that are not also NN^k under the tighter constraint of convexity.

One way to see that a restriction of w to $\mathbb{W}^k$ ensures that not every image is, generally, an NN^k of a query is by considering two images p and q with feature-specific distances d^p and d^q, respectively, to another image r. If $d_i^p \geq d_i^q$ for all features i, d^q is said to *dominate* d^p (Preparata and Shamos, 1985). In this case, $w \cdot d^p > w \cdot d^q$ for any weight set w. Hence, image p cannot be the NN^k for that particular query.

3.2.8 NN^k as the extreme points of a convex hull

The best upper bound on the number of potential NN^k is the number of images in the collection, and is independent of the number of features. We can hence conceive of situations in which every image qualifies as the NN^k of the query, but the chances are negligible since the set of NN^k have to satisfy a tight constraint: any k-dimensional point whose k coordinates maximise, or minimise, a convex function must lie on the convex hull of the point set. Because NN^k minimise the convex function $w \cdot d$, they lie on the convex hull of the set of points d. Let P_j be the distance vector with smallest j component. The NN^k are then the vertices of the convex hull contained in the k-dimensional, convex subspace defined by the set P_1, P_2, $\ldots P_k$.

In Figure 3.5 (left) we plot the distances of a few thousand images to a query with respect to two features. Each point corresponds to a vector of feature-specific distances d. The set of NN^k are the seven points of the bottom left part of the convex hull contained in the rectangle defined by the two points P_1 and P_2 (both of which are themselves NN^k). A close-up view of the region containing the NN^k is given on the right of Figure 3.5 . It illustrates how we can geometrically find the set of weights w for which a particular image of the convex hull is the NN^k: we extend the boundary lines of the convex hull connecting d to its neighbours r and t and determine vectors u by projecting d onto lines orthogonal to these boundary lines. If w is a unit vector in the direction of u, then $|u| = w \cdot d$. It is clear from the figure that the highlighted point identifies the NN^k for any weight having a direction somewhere between u_1 and u_2. Outside of this range, the neighbours of d will lead to a smaller

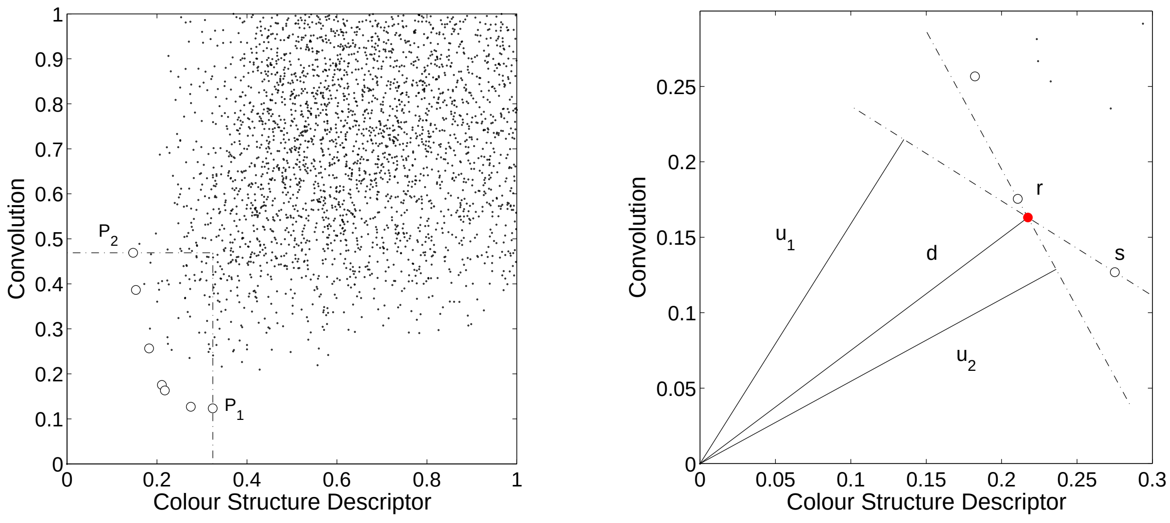


Figure 3.5: Plot of distances between images and a query, the seven NN^k on the bottom left are part of the convex hull (left). Weights w lying between u_1 and u_2 will lead to a projection of d which is minimised for that NN^k (right).

projection: for $\angle(w) > \angle(u_1)$, $w \cdot s < w \cdot d$, for $\angle(w) < \angle(u_2)$, $w \cdot r < w \cdot d$ (here $\angle(x)$ denotes the angle between the horizontal axis and the vector x).

In two dimensions the set of permissible weights satisfying the convexity constraint form a line and can be represented on the same figure. This is done in Figure 3.6. Each NN^k is associated with such a set of w which is the two-dimensional equivalent of the three-dimensional weight space partitions of Figure 3.3.

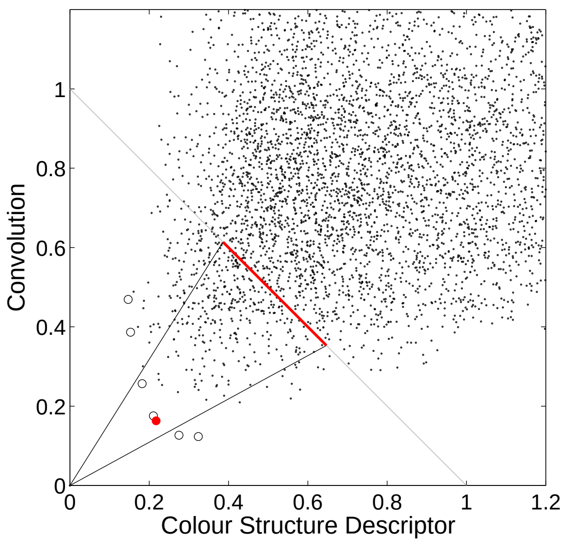


Figure 3.6: In two dimensions, the weights associated with each NN^k form a line segment in $\mathbb{R}^2$.

Note that the set of NN^k of Figure 3.6 is a subset of the NN^k of Figure 3.3. In particular, the seven line segments that divide the edge of the triangle along which the HSV feature weight is zero correspond exactly to the seven NN^k found in two dimensions. This is generally the case. If we denote with F_1 and F_2 two sets of features and with R_1 and R_2 the two corresponding set of NN^k, then $F_1 \subseteq F_2 \implies R_1 \subseteq R_2$. Thus, the problem of finding the NN^k of an image is equivalent to finding a subset of the extreme points of the convex hull. The problem of determining the convex hull is a classic problem of computational geometry and central to many applications ranging from pattern recognition to image processing. With N being the number of points, the task can be accomplished in $O(N \log N)$ time in two dimensions. Unfortunately, the complexity shoots up to $O(N^{\lfloor k/2 \rfloor + 1})$ for $k > 3$ (Preparata and Shamos, 1985). The identification of NN^k with extreme points of the convex hull, while certainly being instructive in its own right, cannot easily be exploited for more efficient NN^k determination.

In Figure 3.5 the two features exhibit some degree of positive correlation and the number of extreme points in the bottom left part of the convex hull is correspondingly small. Figure 3.7 shows a synthetic data set with the same number of points as in Figure 3.5 but with the two feature-specific distances being negatively correlated. It illustrates the fact that the relationship is, contrary perhaps to intuition, rather weak. In this particular case there are only four extreme points in the orthant of interest.

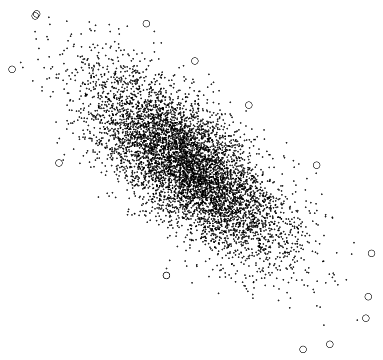


Figure 3.7: The extreme points of the convex hull for a synthetic data set with negative correlation between features.

3.2.9 The Skyline operator

The interpretation of NN^k as extreme points highlights its close connection to the Skyline operator with which Börzsöny et al. (2000) have proposed to extend the set of standard SQL operations. The Skyline is defined as the set of points that are not dominated by any other points. Such points are those for which there does not exist any other point that is better under some criterion for each of the considered attributes. When searching for accommodation as a university student, attributes of interest may be proximity to the university as well as the amount of rent to pay. In this context, a place dominates another if it is both cheaper and closer to the university than some other place, and the Skyline contains all those places for which there does not exist any place that is both cheaper and closer.

The Skyline is thus the solution of the well-studied maximum vector problem (Preparata and Shamos, 1985). It turns out that the convex hull of the data is a subset of the solution set, both of which can be thought of as representations of the boundary of the data set. Although very similar to the convex hull, the Skyline is easier to compute. In fact, algorithms can be devised that run in $O(N(\log N)^{(k-2)}) + O(N \log N)$ time. For any function which maps attribute values monotonically to some overall score, the solution maximising that scoring function will be part of the Skyline. It does, in other words, not contain any points that are nobody's favourite.

3.2.10 Scaling with k and the size of the image collection

It is interesting to investigate the relationship between the number of NN^k and the number of features. While the number of vertices visited in a systematic grid scan increase both with the number of features and with the resolution of the grid, the actual number of NN^k evidently depends only on the number of features and their nature. As we have argued in Section 3.2.8, adding more features can only increase the number of NN^k but the precise relationship is difficult to divine. An idea of the relationship can in principle be obtained by computing for a set of queries the number of NN^k as we increase the number of features. If the set of features is augmented in the same way for each query, the result will reflect not only the general scaling property we seek to elucidate, but also the independence relationships, or the lack thereof, between features. For example, if the first group of features by which we augment our feature set are only very weakly independent of each other, we expect the number of NN^k to increase only mildly. As we begin to add features that are independent of those already in the set, we expect the number of NN^k to increase more markedly. The converse would result if we began with a set of highly independent features. This sensitivity to the order in which features are added suggest a different method whereby we do not prescribe any particular order of features but randomise the order for each query. Once the order is fixed, we compute the number of NN^k for a particular query and an increasing set of features. The result is shown in the left graph of Figure 3.8 for a particular collection. For each query we vary the number of features between one and eleven. Although the number of NN^k is clearly bounded above by the size of the collection, at least for the first eleven features their numbers increase in a roughly linear manner.

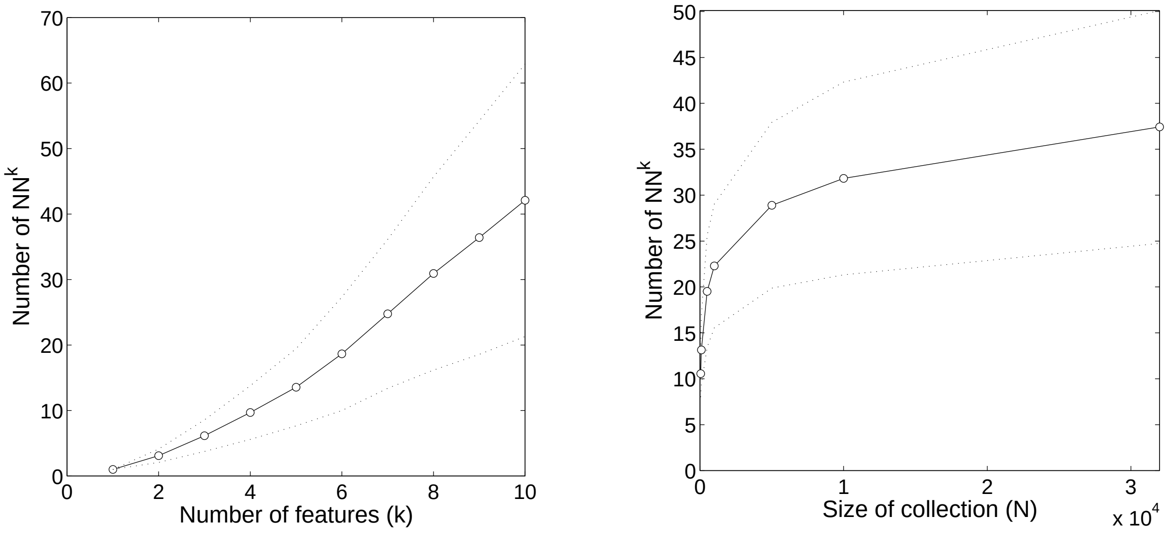


Figure 3.8: Scaling of the number of NN^k with the number of features for one particular collection (left), and with the size of the image collection. The dotted lines mark one standard deviation.

To see how the number of NN^k scale with the collection size, note that the larger the collection, the lower the probability that a new data point may extend the existing convex hull. Again, we expect the number of NN^k to level off although the precise nature of the relationship is hard to divine. The actual relationship is depicted on the right graph of Figure 3.8.

3.3 Optimisation

As we have seen in Section 3.2.2, a simplicial mesh with fixed resolution n and k features requires the evaluation of N distances at $\binom{n+k-1}{n}$ vertices. In this section we will consider a number of techniques that can be used to help render the search for NN^k more efficient. First of all note that since NN^k determination must be understood as an approximation, the computational effort spent depends to a large extent on how certain we want to be that all NN^k have indeed been identified. We shall subsequently refer to the proportion of NN^k found as the *coverage* under a particular resolution. There are two strands along which we can optimise. First, we can adjust the resolution of a fixed grid such that we achieve an acceptable trade-off between computational cost and coverage. Secondly, we can exploit geometric properties and ignore some areas of the weight space altogether.

3.3.1 Selection of grid size

Since the distance vectors of the NN^k form a subset of the convex hull of the entire point set, their number can generally be assumed to be considerably smaller than the number of images in the collection. Further, since the NN^k partition the $(k-1)$-dimensional subspace of permissible weight vectors, it appears reasonable to assume that it should be relatively easy to find a large proportion of them even with a grid of low resolution. This is indeed the case. To evaluate the coverage under different resolutions, we need to know the true number of NN^k, or a sufficiently good lower bound thereof. The former can be obtained by finding those extreme points of the convex hull of the set of distance vectors that lie in the orthant containing the origin, while the latter can be obtained simply by determining the number of NN^k under a very fine grid. For any other resolution we may then determine the coverage as the ratio of the number of NN^k at that resolution to our lower bound. We have computed the coverage for 100 randomly selected images and a fixed number of features.

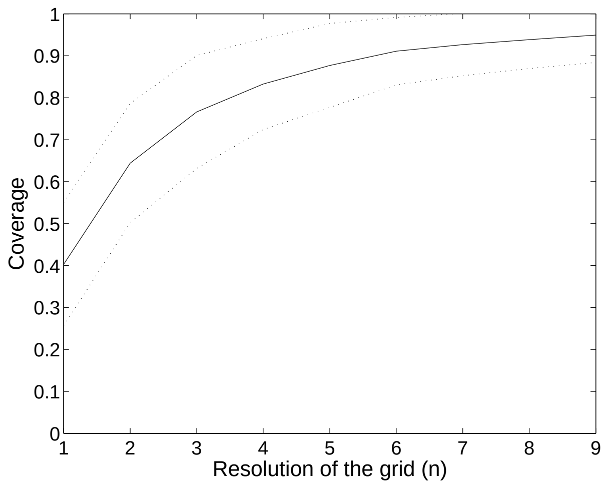


Figure 3.9: Convergence of the number of NN^k as the grid resolution increases. The dotted lines mark one standard deviation.

Figure 3.9 displays a summary of the results for seven features and 6192 images. Coverage increases rapidly for low resolutions and converges relatively quickly towards one. We can be reasonably confident, therefore, that in spite of its approximate nature, the simplicial mesh approach captures most of the neighbours of an image. Also note that the probability of an NN^k escaping the simplicial mesh is inversely proportional to the relative weight space that the NN^k occupies. Images that are missed by the grid are therefore those that have the lowest support.

Theorem 1 tells us that the number of vertices in the grid depends in a simple, and nearly symmetrical way on the resolution n and the number of features k. Knowledge of this precise relationship allows us to control the computational cost of the grid search. Given an upper bound U on the computational cost and a fixed set of features k, we can choose the largest n such that

$$c(n,k) = \frac{(n+k-1)!}{(k-1)!\, n!} \leq U. \tag{3.4}$$

We can approximate the factorials using Stirling's formula: $x! = x^x e^{-x}\sqrt{2\pi x} + O(1/N)$, where the final term indicates that x times the magnitude of the error is bounded in the limit as $x \to \infty$. In particular, one can show that the condition is well approximated by

$$\frac{(n+k-1)^{n+k-1}}{n^n} \leq \xi, \tag{3.5}$$

where $\xi = U(k-1)!e^{k-1}$. Figure 3.10 shows the relationship between n, k and U as given by Equation 3.5 for the special case of equality.

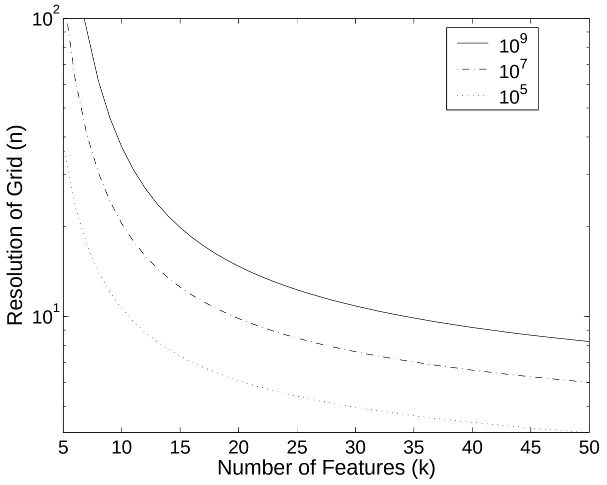


Figure 3.10: Relationship between the number of features and the resolution of the grid when fixing the total number of grid points (10^9, 10^7 and 10^5).

3.3.2 Difference computation

In addition to controlling the number of vertices in the grid at which to determine the nearest neighbour, we may also simplify the computation involved at each such vertex. Recall that each vertex corresponds to a particular weight vector w. The nearest neighbour of the focal image for that weight vector is the image that minimises the weighted sum of distance scores,

$$D = \sum_{f=1}^{k} w_f d_f.$$

As we recurse through the grid, most pairs of consecutive vertices differ in only two components of the weight vector. Instead of computing D according to the above equation involving k multiplications and $k-1$ additions, we can exploit the fact that in most cases $k-2$ of the terms of the product have already been computed at the previous vertex. In such cases we can obtain the distance at the new vertex by subtracting a correction term from the distance at the previous vertex. Let F be the set of indices whose weights differ between the two vertices then the update is

$$D^j = D^{j-1} + \sum_{f:w_f \in F} (w_f^j - w_f^{j-1}) \cdot d_f,$$

where j indexes the step at which the vertex is visited during the recursion. With the recursion code from Section 3.2, we have $w_i^j - w_i^{j-1} = \pm\frac{1}{n}$ where n is the resolution of the grid. Imagine, for example, that we have $w^j = (w_1, w_2, w_3, w_4, w_5)^T$ and $w^{j+1} = (w_1, w_2, w_3, w_4 + \frac{1}{n}, w_5 - \frac{1}{n})^T$. Instead of computing $D^{j+1} = w_1 d_1 + w_2 d_2 + w_3 d_3 + w_4 d_4 + w_5 d_5$ we compute $D^{j+1} = D^j + \frac{1}{n}(d_4 - d_5)$. To be able to apply the method, we need to store the previously computed distance and have a way of detecting the weights that have changed and by which amount. It turns out that a necessary and sufficient condition for the update method to be applicable to a grid point is that $i > 1$ and dim $= 0$ in the inner for loop of the `enumeratePi` method. The modification required is a branch condition after line 11 that causes the execution of the update method if $i > 1$. For a collection of 8000 images we have plotted the percentage gain in CPU time that can be achieved with this method (Figure 3.11). The performance benefits are quite appreciable and become greater as we increase the resolution of the grid. With eight features and a grid size of six the update method speeds up performance by almost 40%.

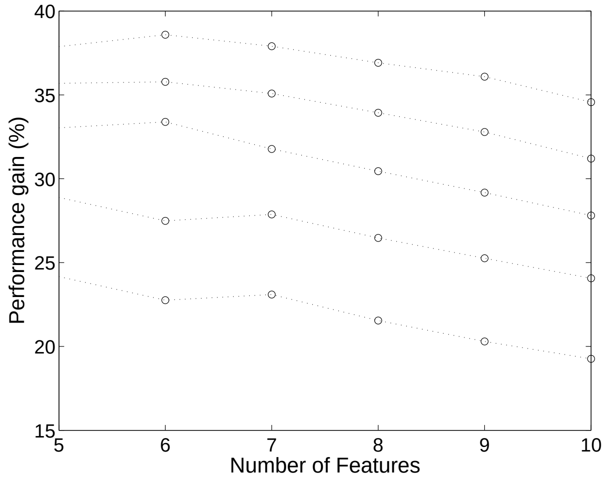


Figure 3.11: Speed-up achieved through difference computation.

3.3.3 Exploiting convexity

As we have seen in Section 3.2.4, each of the NN^k regions are convex. If the NN^k under two different weight vectors are the same, we can therefore deduce that the same NN^k must be produced for all intermediate weight vectors. This provides a way to reduce the number of function evaluations.

3.4 Conclusions

This chapter has introduced the notion of the set of NN^k of some focal image. These are defined as all those images that are closest to the focal image under at least one of the infinitely many instantiations of a parametrised distance metric. We have argued that the NN^k may be viewed as a pictorial exemplification of all those semantic facets of the focal image that lie within the representational scope of the features. Because no single feature is systematically favoured over others, the NN^k offer a maximally unbiased interpretation of the focal image.

We have described a method for approximating the set of NN^k by discretising the space of permissible weight sets. A recursive scan of the finite grid thus constructed achieves a high coverage even for low grid resolutions and a large number of features. We prove a number of important theorems some of which can be used for reducing the computational complexity of NN^k computation.

The number of NN^k increases logarithmically with the number of images in a collection and has empirically been found to lie around 30 for $10,000$ images and eight features, a number that can comfortably be accommodated on a screen. The relationship between the number of NN^k and the number of features appears nearly linear up to 11 features. For a large number of features, we expect the number of NN^k to level off.

We have sketched the relationship between the NN^k idea and the Skyline or Pareto operator from relational databases. The latter extracts all those tuples for which there does not exist any other tuple that is better for all attributes. If we view the distance scores between an image and a focal image under k features as a k dimensional vector, the NN^k lie on the convex hull of the point set. The Skyline, meanwhile, corresponds to points from the convex hull and an additional number of points from the contour.

This chapter has introduced a number of terms describing properties of an NN^k, such as its support and its iso-NN^k region. These terms will resurface throughout the rest of the thesis and for quick reference are collectively described in the glossary.

As a first significant application of the ideas presented here we will in the next chapter develop a powerful method of relevance feedback in the context of image retrieval.

Chapter 4

NN^k Search

4.1 Introduction

Recall from Chapter 2 that the overall distance between a query q and an image p from the collection is commonly taken to be the weighted sum over the feature-specific distances

$$D(p, q) = \sum_{i=1}^{k} w_i d_i(p, q), \tag{4.1}$$

and that it is the goal of many a relevance feedback technique to iteratively fine-tune the weight vector w. As we noted in the same chapter, a widely overlooked problem in this context is that of parameter initialisation. The limitation of resorting to default parameters does not become apparent when the collection size is small and queries are kept simple as a sufficient number of relevant images are retrieved even with crude initialisation. As collections continue to grow, however, this ceases to be the case and default parameters will generally not be good enough to supply users with positive examples for effective relevance feedback. Existing relevance feedback techniques successfully address the problem of fine-tuning weights when the initial retrieval result is already of reasonable quality but fare badly if it is not and thus are inadequate on their own in more realistic settings.

A simple method for initialising parameters prior to any relevance feedback is based on the observation that some features are generally better suited for most queries than others, so that one can learn feature weights that maximise average performance on a set of training queries (Yavlinsky et al., 2004). The method assumes that the distribution of queries is somewhat similar to that of the training set which might not always hold. More importantly, by not taking into account that the optimal w may vary considerably with the query it leaves unexploited much of the potential performance gain. How the optimal w can vary between different query images is illustrated in Figure 4.1 with three images from the Corel collection. As in Chapter 3 each point within the simplex is associated with a three-dimensional weight vector w whose components are proportional to the distance between that point and the respective side of the triangle. For a set of such points we have measured the average precision that is obtained when retrieving with the corresponding weight set from a collection of 10,000 images. The lines shown are lines of equal performance intrapolated from the finite sample, the weight set for which performance is maximised is marked by a white dot. It is clear that the performance surfaces exhibit notable differences.

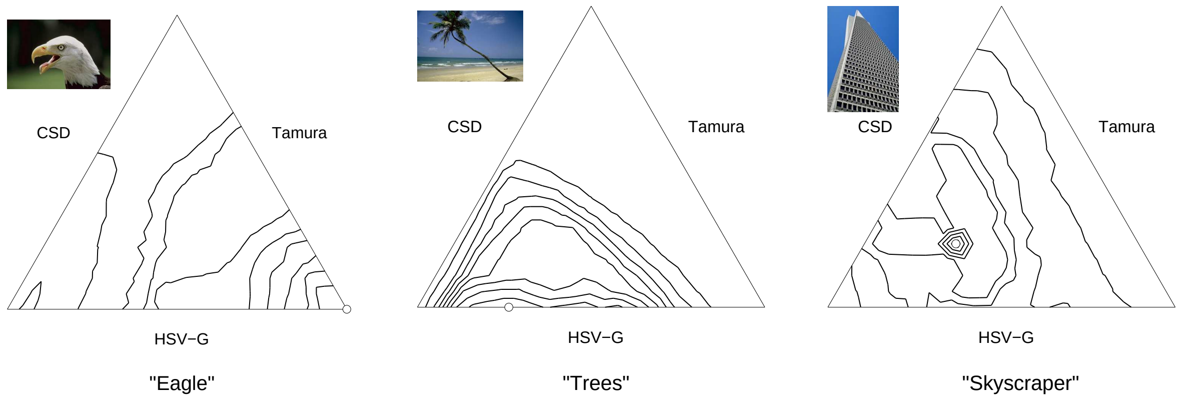


Figure 4.1: Performance landscapes for three different queries.

If it were true that similar images share similar optimal weights, it would be conceivable to learn a mapping between the feature representation of an image and the set of optimal weights on a training collection of queries. Given a new query image, we could then estimate its optimal weight set based on its visual characteristics. The problem of learning such a mapping is greatly compounded by the observation that images admit to multiple, equally valid interpretations and that the optimal weight sets may vary considerably not only between queries but between different interpretations of the same query. The extent to which this is the case is illustrated in Figure 4.2. The query is a photograph of an open landscape and is annotated with the words 'evergreen tree', 'moss' and 'rain'. We consider an image relevant, if it contains the respective term in its annotation and, as before, measure performance in terms of average precision. Even for terms which we might associate with visually similar image classes, such as 'moss' and 'evergreen tree', the optimal w can be quite different. Both Figure 4.1 and Figure 4.2 vindicate our earlier claim that a uniform w, corresponding to the centre of the simplex, can only be expected to produce suboptimal results. In large collections, the optimal weights is likely to remain unattainable with the scantiness of positive examples from the first retrieval.

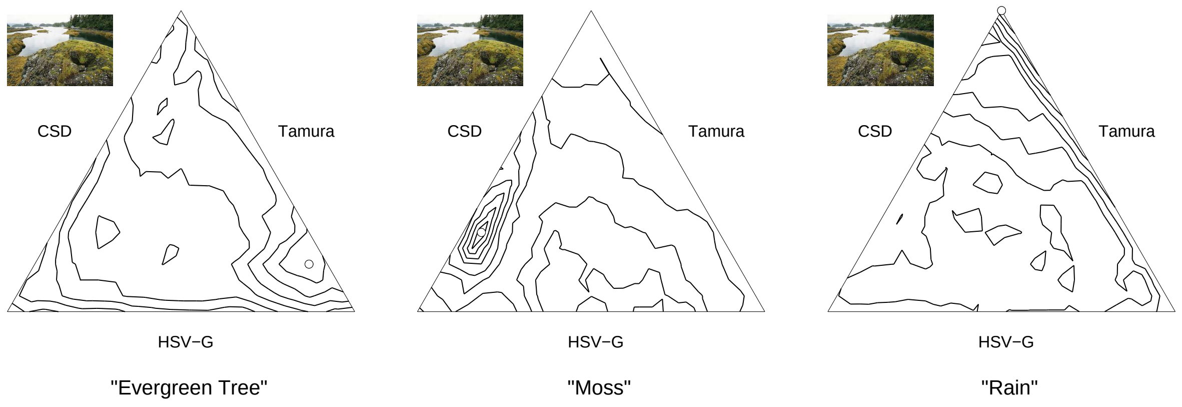


Figure 4.2: Performance landscapes for one image with three possible meanings.

The only notable attempt to achieve parameter initialisation prior to the first retrieval is due to Aggarwal et al. (2002), which we referred to in Chapter 2. Relevance feedback is given on a set of images that are created on the fly by modifying segments of the query image in various ways. Modifications include changes in shape, size and colour. Each modification can be thought of as a particular parameter setting under which the modified query is close to the original query. After providing positive relevance feedback, the optimal feature weights are learned similar to Rui et al. (1998). The problem of parameter initialisation is thus solved by not initialising parameters in the first place but by showing the user a set of images representing different parameter settings. The major limitation of the method stems from its dependence on good segmentation results and the arbitrariness of the modifications applied to the segments. In many cases, and in particular when the images cannot meaningfully be segmented, it may be a difficult task for the user to judge relevance. Also, the first set of images with which the user interacts do not come from the collection and are by themselves of no interest to the user.

The NN^k idea developed in the previous chapter suggests an interesting alternative. The method, which we shall call NN^k search, offers a principled way of optimising feature weights for individual queries. It involves little user interaction and its initial step allows the user a first glimpse on the content of the collection. In spirit it is very similar to the method of Aggarwal et al. (2002) but it manages to stay clear of its major shortcomings. The method involves two steps and works as follows: we take an agnostic position in the first step and avoid any parameter initialisation by determining the set of images which have minimal distance to the query under at least one of the infinitely many possible combinations of feature weights, in other words, the set of NN^k of the query. Each NN^k can be associated with a characteristic weight set (see Section 3.2.6 in Chapter 3) for which it is closer to the query than any other image. In the second step we gather relevance information on the set of NN^k and choose the characteristic weight sets of the selected images for subsequent retrieval.

We will describe the two steps in greater detail in Section 4.2 and illustrate the method with three particular example queries in Section 4.3. Section 4.4 is the longest section of this chapter. It presents a large-scale evaluation of the proposed method by comparing it with another well-known relevance feedback technique (Rui and Huang, 2000).

4.2 NN^k search

4.2.1 Rationale

By showing the set of NN^k, i.e. the union of top-ranked images over all combinations of feature weights, the first step aims to expose the multitude of possible image meanings that can be represented with the given feature set. If a semantic facet lies within the representational scope of the features, we expect representative images to appear among the NN^k.

Once the NN^k have been determined and characteristic weight sets have been associated with them, users provide relevance feedback by selecting relevant images from among the NN^k. For each selected image we recover its characteristic weight set and compute the overall distances of all images to the original query under that weight set before aggregating these to form a final ranked list. The second step is motivated by the intuition that weight sets under which the selected images are ranked top are likely to retrieve other such images at lower ranks. Each weight set which the user implicitly selects can be thought of as providing access into different semantic territories of the collection with the NN^k from the first step being the signposts.

Although we should be interested in how many of the semantic facets the set of NN^k actually cover, this figure is not easily obtainable experimentally. The total number of interpretations an image may admit to is only bounded by the number of users interpreting the image. Moreover, it should often not be clear whether two facets are sufficiently distinct to warrant their separate existence rendering any such figure somewhat arbitrary. To describe the semantic facet of the focal image a particular NN^k exemplifies means to put a singular interpretation on the NN^k. This again should not always be possible as the image may share several aspects with the focal image at once. The term semantic facet therefore proves rather elusive once we attempt to capture it quantitatively. But it should be clear from, for example, Figure 3.1 in Chapter 3 that it is more than merely a metaphorical aid.

NN^k search takes seriously the observation that the meaning of an image is contextual and that in many cases it can only be revealed by grouping the image with others. The semantic atoms in the NN^k framework cease to be individual images and become image pairs consisting of a focal image (the query) and one of its NN^k. By choosing one of the NN^k we fix the semantics of the focal image, a process which we shall refer to as the *semantic collapse*. The idea of adding meaning by letting the user form explicit associations between images ultimately underpins all relevance feedback techniques. What distinguishes NN^k search from other such approaches is its semantic unbiasedness in the first step and the particular way of associating optimal weight sets with the set of images on which relevance feedback is given.

4.2.2 Merging of ranked lists

If users select only one image from among the NN^k, the one ranked list induced by the corresponding weight set constitutes our final ranked list. If more than one image is selected, we need to merge the ranked lists obtained from each of the weight sets. Merging ranked lists is a problem that has received considerable attention and to which we already alluded to in chapter 2, where it presented itself in the context of multi-point queries. Merging ranked lists intelligently is also of critical importance in distributed information retrieval where search results from multiple collections are to be integrated in one final ranked list and where the choice of the merging method can have a profound effect on performance (Si and Callan, 2002, 2003). Depending on whether only ranks are considered or the individual scores, we can distinguish two broad classes of merging methods.

Score-based merging

Arguably the most basic method of score-based merging is to rank the images according to the highest score they achieve in any of the individual lists. A simple algorithm for achieving this is the l-way merge (Knuth, 1973). It involves a pointer for each of the l lists that initially point at the top-ranked object. The object with the smallest distance is added to the final list and the pointer of the corresponding list is advanced by one entry. Information about the other scores is discarded which might not always be reasonable. If an image is consistently ranked high, this could be taken as additional evidence that it is indeed relevant to a query.

Another shortcoming of this merging method is the need for the scores to have similar distributions. To see why this should be so, consider two features with mean-normalised Gaussian distributions D_1 and D_2 with the same mean and different standard deviations, $\sigma_2 > \sigma_1$. The initial ranks will predominantly be occupied by images from the distribution with larger variance, D_2, while the final ranks are more likely to be from D_1. This is shown in Figure 4.3 for which we have chosen $\sigma_2 = 1.5\sigma_1$.

Such distributional differences do indeed exist among feature distances and they clearly do not vanish upon mean-normalising the data. An example is given in Figure 4.4 which depicts the distance scores for the $8,201$ most similar images to a query under four different features (using the Getty collection described shortly).

Instead of ranking according to the smallest distance, we may consider only the largest distance or the average of the scores. The three methods are respectively called *CombMin*, *CombMax* and *CombSum* (e.g. Fox and Shaw, 1994; Lee, 1997), with the last method being arguably most tolerant to distributional differences.

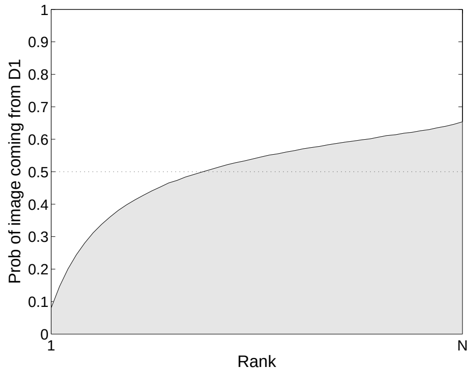


Figure 4.3: Problem with score-based data fusion.

Slightly more sophisticated approaches formulate the problem as a supervised learning task. Given that we know the ideal ranking for a set of queries, we can estimate parameters of a general linear model to find monotonically increasing functions that optimally map the raw scores for each list to a new set of scores that can then be combined in a final rank list (Lin and Hauptmann, 2004).

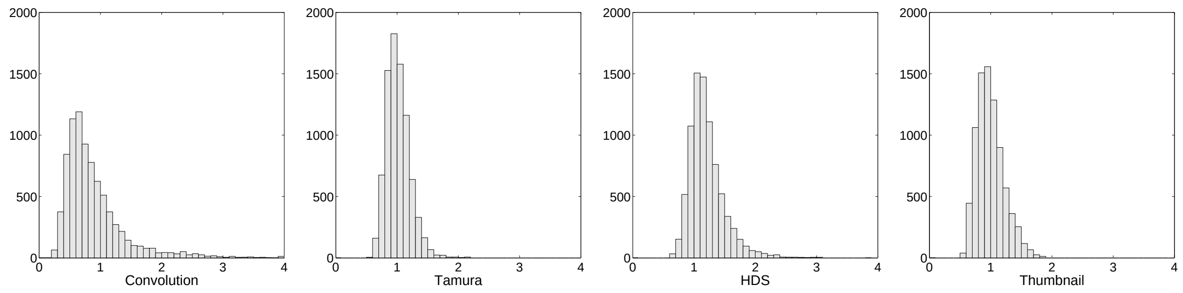


Figure 4.4: Histogram of normalised distance scores for four features.

Rank-based merging

A second class of merging methods ignore scores and work with the ranks instead. Rank-based methods can be implemented very efficiently and easily outperform score-based methods when different lists have widely differing distributions. The simplest such method interleaves the ranked lists according to a Round Robin scheme. Advancing downwards starting from the first rank, we add all of the l entries at the current rank to the final list that have not yet been added. The rank of an image in the final list thus depends only on the highest individual rank of itself and of all the other images.

A popular rank-based merging method is Borda-Fuse (Aslam and Montague, 2001). It assigns points to each entry in each of the lists as follows: given c objects to be ranked (only a subset of which need to appear in individual lists), we assign c points to the top-ranked object in each list, $c - 1$ points to the second-ranked object and so forth. Any candidates not appearing in a list are given an equal share of the remaining points. The final ranking is obtained by summing for each object the points it has been assigned in the individual lists. If we have reason to assume that the ranks in some lists are more informative, it is natural to aggregate the scores in a weighted sum. The weights then capture our prior knowledge about the usefulness of different features, or in the context of meta-search, of different retrieval systems.

Although the search for optimal merging methods is not at the heart of our evaluation of NN^k search in Section 4.4, the literature clearly suggest that merging offers substantial scope for performance optimisation and it should therefore not be ignored. Moreover, merging has been studied largely in the somewhat different context of meta-search in distributed environments where systems typically access different collections with the result that the returned lists exhibit only partial overlap. When objects are returned and ranked only by a subset of search systems, one should reasonably assume that discarding scores altogether might be disadvantageous since the rank an object has been assigned in one list may critically depend on what other objects are also present in that particular list. Score-based merging methods offer some immunity against the biassing effect of partial evidence and should prove more adequate. The present context is different: each feature-specific ranking comprises the same, complete set of images thus removing one of the principal weaknesses of rank-based approaches. We shall see that it is indeed a rank-based method that turns out optimal for our purpose.

4.3 Three examples

Before presenting a quantitative evaluation of the proposed technique, we shall demonstrate its effectiveness anecdotically with example queries from the Corel image collection. Both the collection and the choice of features will be described in greater detail in Section 4.4. In all three query scenarios, we show the results of the first step (the set of NN^k of the query) and the second step (ranked list of images obtained by retrieving with the selected weight sets and merging individual lists). We include the query image at the top left of each group and present images such that their ranks decrease from left to right and from top to bottom. The set of NN^k are ranked according to their support, i.e. the top ranked image to the right of the query is closest to the query under most feature combinations. For presentation purposes we have truncated the results at 19 and we should note that in many cases the number of NN^k on the left from which the user may select relevant images exceeds this number (the average number of NN^k is 46).

The first query image is a photograph of a mountain peak. The set of its NN^k are rather diverse differing widely in colour although blue is predominant and a large proportion in all photographs is coloured fairly uniformly. The images selected for relevance feedback are those at positions (4,4) and (3,2) of the 4×5 grid. The characteristic weight sets of the two images favour a subset of the colour features with weights for all other features being negligible. Merging the two lists induced by the two weight sets yields the ranked list on the right. By definition, the selected images turn up at the top (position 2 and 3) since we retrieve with their characteristic weight sets. These are followed by a number of more mountain pictures. It is evident from the right group that retrieval with a limited set of weights produces images that are visually considerably more uniform than the original set of NN^k.

The second query is a photograph of a road sign which once again produces a set of visually distinct and semantically unrelated NN^k of which only a few may be considered relevant. The selected images include the two road signs from the left group plus one other image from the remaining NN^k. The three images are retrieved top in the right group with five more road signs appearing in the top 19.

The third query image depicts a glass with a transparent liquid and the search is for examples of the 'beverage' class. Selecting the two relevant images found in the left group at positions (1,5) and (4,5) produces five more relevant images after the second step.

4.4 Evaluation

This section investigates the effects of NN^k search on retrieval performance. As is customary for this purpose, we employ manually annotated image collections and automate the feedback process so that an evaluation can be carried out on a large scale.

The standard measure of retrieval performance in information retrieval continues to be mean average precision (Clough et al., 2005; Vorhees and Buckland, 2005). When evaluating relevance feedback techniques, however, arguments based solely on mean average precision must be taken with caution. When the set of relevant images is low, even a small shift of relevant images in the ranked list may result in substantial gains in average precision without necessarily leading to an increase in the number of relevant images displayed to the user. To take an extreme example, consider there being only two images relevant to a query. If these are ranked at positions 3 and 4 by one system and at positions 2 and 3 by another, the average precisions of the two systems are 50% and 67%, and thus, on the face of it, we would have a relative gain of 34%. This figure is clearly a hugely inflated estimate of the difference between the systems. In fact, for the user the difference is unlikely to be of any practical significance. This effect is known as the ranking effect (as opposed to the feedback effect) and in document retrieval a number of techniques such as residual ranking and rank freezing have been proposed to control for it (Chang et al., 1971; Ruthven and Lalmas, 2003). We address the problem by employing precision at rank 50 as an additional performance measure. We denote this by Pr-50 and, to avoid confusion, should emphasise that this is not precision at 50% recall. Assuming that a screen can comfortably accommodate 50 images, changes in Pr-50 give us an estimate of how the number of relevant images changes among the set with which the user interacts. We focus on precision rather than recall as we believe that for large image collections, users are more interested in the former than the latter. In the context of Web document retrieval, for example, this is a commonly made assumption (Hawking et al., 1999).

4.4.1 Image collections and queries

Corel

Our evaluation is based on two collections. The first one is a subset of the Corel 380,000 Photo Gallery comprising 31,997 images. The Corel collection enjoys great popularity among researchers not only because of the good quality of the photographs. The pre-assignment of images to categories by means of their position in the directory hierarchy greatly facilitates automatic evaluation: an image is considered relevant to a query if it is from the same directory (Heesch and Rüger, 2003; Urban and Jose, 2004a). By not allowing membership to more than one class, however, the Corel classification

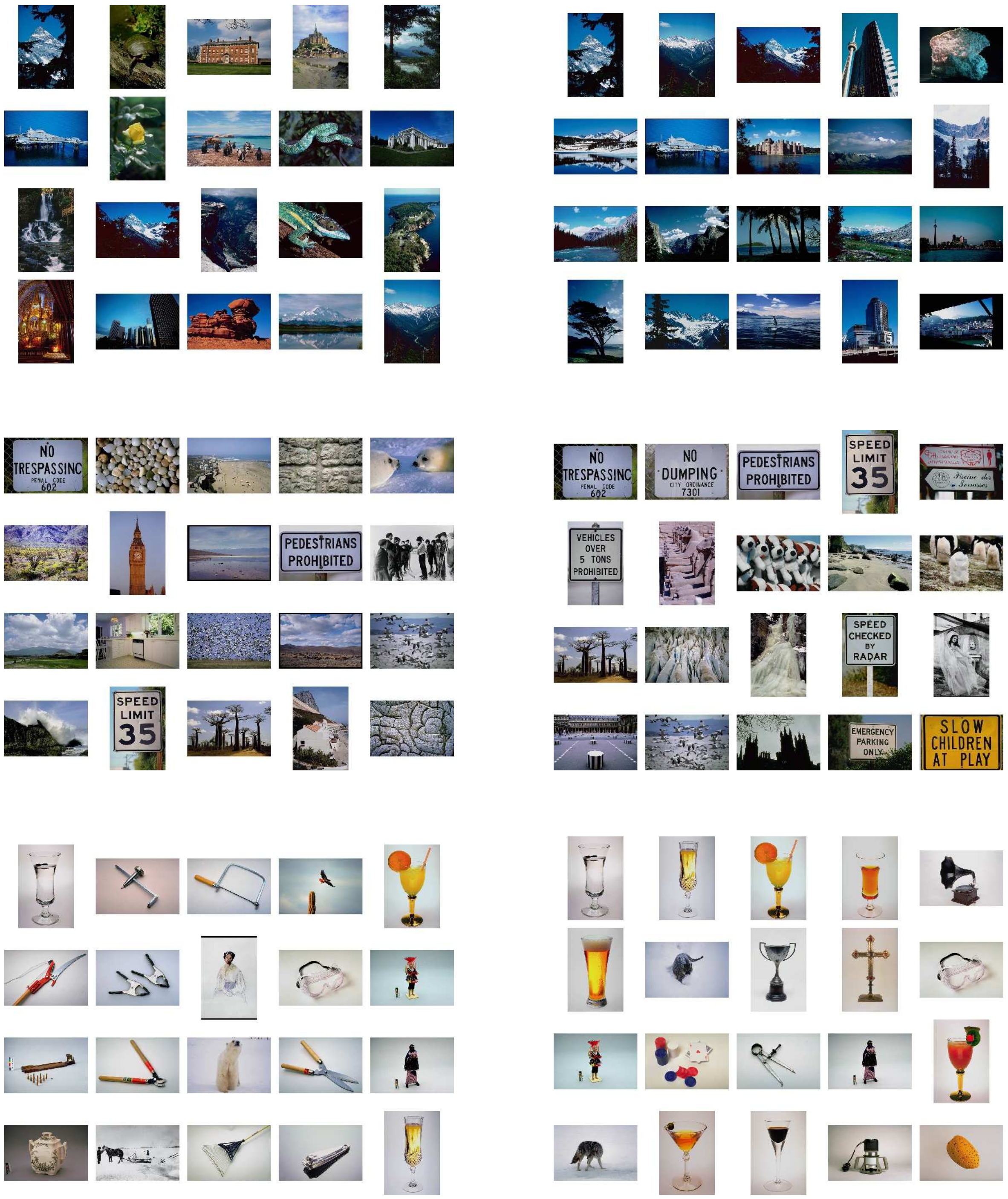


Figure 4.5: Results for the first (left) and second step (right) of NN^k search for three queries.

fails to model image polysemy. To overcome this limitation we make use of the textual annotation that accompanies each image as this can form the basis of an alternative classification with overlapping classes. The joint vocabulary of the 31,997 images comprises 20,250 terms of which we only keep those that are associated with at least 20 and at most 100 images. Of the resulting 530 classes, we discard those with too little visual coherence leaving us with a final set of 191 classes with an average of 49 images and ranging in size from 20 to 99. Example classes beginning with 'S' include 'Sandstone', 'Santorini', 'Sculptures', 'Seals', 'Seashore', 'Sepals', 'Shells', 'Ship', 'Shrimp', 'Skyscraper'.

Getty

We built a second collection of a more diverse kind with richer and more consistent annotation than Corel. We downloaded medium-resolution thumbnails of photographs from the Getty Image Archive website (http://www.gettyimages.com) along with the annotations assigned by the Getty staff. The selection of photographs was obtained by submitting the following query to the Getty website: "photography, image, not composite, not enhancement, not 'studio setting', not people", with the additional search option to exclude illustrations. With this query we hoped to obtain a random selection of photographs that would exclude pictures with no photographic content, digitally composed or enhanced photos and any photos taken in a studio setting. Because the resulting dataset contains pictures from a number of different photo vendors we hope to reduce the chance of unrealistic correlations between images.

Our Getty collection contains 8,202 images from which we construct classes as before. The joint vocabulary of 8,551 terms is reduced by retaining only those terms that are associated with at least 20 and at most 100 images. We again discard terms that we consider either too difficult for visual retrieval such as 'freshness' or too easy such as 'blue'. After these pruning steps the vocabulary has shrunk to 100 terms which we regard as class labels. Examples of classes for the Getty collection beginning with 'S' are: 'Salad', 'Sand Dune', 'Sea Life', 'Seascape', 'Skyline', 'Snowcapped', 'Spotted', 'Steam', 'Storm' and 'Sunrise'.

With images belonging to several relevance classes, a query consists of an (image id, image class) pair. The same image can participate in more than one query. Because of the extensive reduction of the vocabulary, only a subset of the images will be annotated with one of the remaining class labels and it is only these that we can employ as queries. Of the 3533 queries for Corel and the 4479 queries for Getty, we choose a random subset of 500 for evaluation purposes. A summary of various collection parameters is shown in Table 4.1.

	Corel	Getty
Size	31,997	8,202
Number of classes	191	100
Class size (avg)	49	44
Class size (range)	20–99	20–99
Avg polysemy	1.1	1.4
Max polysemy	5	7

Table 4.1: Summary of properties for the two collections used in the evaluation

4.4.2 Visual features

We have chosen a set of eight colour and texture features that we believe provide a good coverage of visually relevant image aspects. Some have proven successful individually in other retrieval experiments or form part of the MPEG-7 standard. In some cases we divide images into tiles and compute features for each tile.

HSV global colour histogram:

HSV is a cylindrical colour space with H (hue) being the angular, S (saturation) the radial and V (brightness) the height component. We choose a linear subdivision into 10 hues, 5 saturation values and 5 brightness values yielding a 205-dimensional feature vector: Since hue is singular along the achromatic axis we merge all pie-shaped three-dimensional HSV bins touching the achromatic axis. The HSV colour histogram is normalised so that the components add up to 1.

HSV focal colour histogram:

This is the same as the global colour histogram except that it is applied only to the central 25% of the image.

Colour structure descriptor:

This feature is defined in the HMMD (hue, min, max, diff) colour space and is part of the MPEG-7 standard (Manjunath et al., 2001). The HMMD space is derived from the HSV and RGB spaces. The hue component is the same as in the HSV space, max and min denote, respectively, the maximum and minimum among the R, G, and B values, and the diff component is the difference between max and min. The colour space is quantised non-uniformly into 184 bins with the three dimensions being hue, sum (defined as $(\max + \min)/2$) and diff.

The colour structure descriptor is obtained by sliding a 8×8 structuring window and count for each HMMD bin the number of positions for which the window contains at least one pixel from that bin. This descriptor is capable of discriminating between images that have the same global colour distribution but different local colour structures (see the example at the beginning of Chapter 3). The 184 bin values are normalised by dividing by the number of locations of the structuring window so that each of the bin values falls in the range $[0, 1]$ but the sum of the bin values can take any value up to 64.

Thumbnail feature:

This feature is obtained by scaling down the original image to 44×27 pixels and then recording the gray value of each of the pixels leaving us with a feature vector of size 1,188. It is suited to identify groups of near-identical images.

Tieu's convolution features

These features have been proposed by Tieu and Viola (2000) in conjunction with their boosting method. It is a set of highly selective features generated by applying 25 primitive convolution filters first to the initial gray-level image and then to the outcome of the filtering process. After applying the convolution three times, we obtain $25^3 = 15625$ feature maps each of which is then represented by the sum of its pixel values. The features take large values typically only for a small fraction of the image collection.

Tamura features

We compute the three Tamura features coarseness, contrast and directionality as proposed in Tamura et al. (1978) for each of 9×9 non-overlapping tiles. For each pixel we compute the values of each of the three features and compute for each tile a three-dimensional histogram. More details regarding this feature can be found in Howarth and Rüger (2004).

Co-occurrence feature

The co-occurrence feature is based on the notion of a grey-level co-occurrence matrix P. Given a quantisation of the pixel intensity into k grey-levels and some vector d, each entry p_{ij} gives the number of times that two pixels separated by d belong to respective grey-level bins i and j. By varying d the resulting $k \times k$ co-occurrence matrices can capture different texture characteristics. Based on the evaluation in Howarth and Rüger (2004) we choose to split the images into 9×9 non-overlapping tiles. We quantise the colour space into 32 grey levels and determine 16 co-occurrence matrices for each tile by considering four different lengths (1-4) and four different orientations (0-$3\pi/2$ in steps of $\pi/2$) of d. For each co-occurrence matrix P we calculate a homogeneity feature H as

$$H = \sum_i \sum_j \frac{p_{ij}}{1 + |i - j|}.$$

Gabor feature

Gabor filters are amongst the most popular methods for texture extraction. Our implementation of the Gabor filter is based on Manjunath and Ma (1996) and Howarth and Rüger (2004). Each image is passed through a bank of four orientation-sensitive filters and two scale-sensitive filters according to

$$W(x, y) = \int I(k, s) g^*(x - k, y - s) \, dk \, ds.$$

where g^* is the complex conjugate of a normalised Gabor wavelet. The feature vector consists of the mean and standard deviation of the modulus of W for 7×4 tiles. Since different filters produce outputs in different ranges, we normalise both the mean and the standard deviation for each filter.

4.4.3 Reference performance

Retrieval performance is exceedingly sensitive to experimental conditions and thus absolute performance figures need to be interpreted with great care. The size and nature of the image collection,

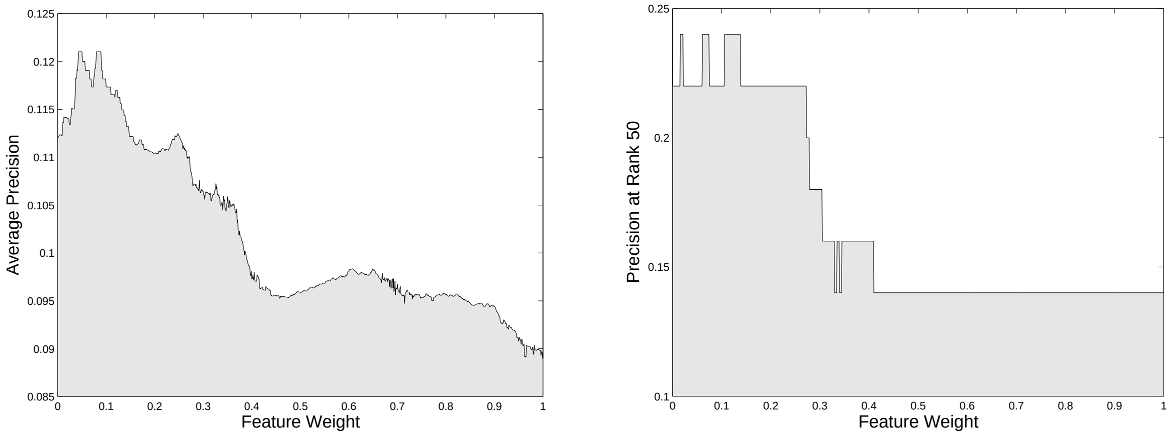


Figure 4.6: Variation in retrieval performance as we vary one feature weight.

the set of query images and the choice of visual features all have a potentially profound effect on performance. Any system evaluation ought to be carried out, therefore, in direct comparison with other systems. We compare NN^k search with three other methods. The first method weighs all features equally and provides us with a baseline. The second method, an oracle, weighs features according to an optimal weight set that has been determined empirically for each individual query. No system that retrieves with only one weight set can do better than this. The third method is an implementation of the well-known weight update technique proposed by Rui and Huang (2000), a relevance feedback technique that aims to optimises such a single weight set. We describe each method in turn.

Baseline performance

The baseline performance for our experiments is that achieved by giving all features an equal weight. Without any information about the relative performance of different features, such a default set is our best guess and it is almost universally used as initialisation strategy when evaluating relevance feedback techniques. The third method, the weight update technique by Rui and Huang (2000), will use the results from a baseline run as the first set on which relevance feedback is performed.

Oracle performance

We are not only interested by how much the technique improves retrieval performance beyond the baseline, but also to what extent we are able to approach the performance that could be attained if we knew for each query the best single weight set. This is an important benchmark since a large number of relevance feedback techniques are concerned with finding such a single weight set (e.g. Rui et al., 1998; Rui and Huang, 2000; Heesch and Rüger, 2003). Should it turn out that NN^k search surpasses this optimal performance, we may legitimately claim, therefore, that the method does better than all these relevance feedback techniques without having to test them one by one.

In our endeavour to determine the best set of weights for individual queries, we first observe that owing to the discrete nature of ranks, an infinitesimal change in the weights may bring about a finite change in performance. As a result the scalar field which maps weights to average precision or Pr(50) is discontinuous and not differentiable. One way to optimise performance under such conditions is to provide a stochastic formulation of the objective function thereby rendering it differentiable, see for example Goldberger et al. (2005). Alternatively, we can carry out a grid search of the weight space as proposed in Chapter 3 for the purpose of NN^k determination. The performance value thus found is unlikely to be a local let alone a global maximum but the grid search can be coupled with a subsequent gradient ascent search which should at least lift us to a local maximum. This is the strategy adopted here.

Figure 4.6 depicts how retrieval performance for a particular query varies as we change the weight of a single feature. The discontinuity is evident when performance is measured in terms of Pr(50) but much less so when measured in terms of average precision. The reason for the difference is that average precision takes into account the entirety of the ranked list and any small change in the distribution of relevant images is likely to have a correspondingly small impact on performance. We can also see from the average precision graph that we should be able to obtain an approximation of the slope for the purpose of our gradient ascent step. Let j index the feature whose weight we vary by a small amount h, then an estimate of the jth component of the gradient is

$$\frac{P(w) - P(w + \Delta w)}{h},$$

where Δw is a vector whose jth element is h. Because the weights are sum-normalised, any increase in one feature weight must be accompanied by a corresponding decrease in the sum of all other weights. Strictly speaking, therefore, increasing the jth weight by h requires us to multiply all others by some common factor that ensures that they again sum to 1. We find that the other components of Δw are given by

$$\Delta w_i = \left(\frac{h}{w_j - 1} \right) w_i.$$

After identifying the weight and the direction (the sign of h) along which performance increases most, we perform a search along the line given by the set

$$\left\{ w \;\middle|\; w_i = 1 - \frac{t w_i^0}{1 - w_j^0}, \;\; w_j = w_j^0 + t, \;\; t \in [-w_j^0, 1 - w_j^0] \right\},$$

where j indicates the weight we vary, and w^0 denotes the starting weight. With three features, the line corresponds to the straight line on the two-dimensional weight simplex that crosses the point corresponding to w^0 and the vertex opposite the side that corresponds to the feature whose weight we vary.

The w that maximises performance locally is found by moving along the line in small intervals Δw starting at w^0 until performance begins to decrease. In Figure 4.6 the line search begins at $w_j^0 = 0$ and moves at intervals of 0.01 to the right. As expected the line search terminates no later then the first peak at $(0.05, 0.121)$, which happens to be the global maximum in that direction. The next iteration finds the new direction along which performance can be increased further, and so on until performance decreases in every feature direction. In the above example, performance can be increased to above 0.14 using this method. Although the method is not guaranteed to find the global optimum, it generally comes very close to it as suggested by grid searches with very high resolution.

At the risk of belabouring the point, we wish to stress once more that the performance thus obtained does *not* place a limit on the performance of the NN^k method as the latter allows retrieval with multiple weight sets whenever more than one NN^k has been chosen. The oracle provides an upper performance bound for any relevance feedback method that attempts to optimise and retrieve with a single weight set.

In case that NN^k search does not exceed the performance of the oracle, it is important to have a direct comparison with alternative relevance feedback methods. Below we remind ourselves of one of the most popular methods that was introduced earlier in Chapter 2 and which has been taken up by other groups (Urban and Jose, 2004a).

Weight update according to Rui and Huang (2000)

Recall from Chapter 2 that Rui and Huang (2000) derive an optimal solution for the feature weights w when the objective is to minimise the summed distances between relevant images and the query with the distance of each relevant image being weighted by its relevance score v_i. Given N positive examples, they find that the optimal w satisfies

$$w_j \propto \frac{1}{\sqrt{\sum_{i=1}^{N} v_i d(p_{ij}, q_j)}}, \tag{4.2}$$

where the denominator is the sum of the weighted distances between the query and all relevant images under feature j. In our experiments we choose at random a maximum of ten relevant images and set their relevance scores to one. Note that since the aggregation formula for the feature-specific distances is linear in the feature weights, the proportionality constant in Equation 4.2 can be chosen arbitrarily. The first retrieval is with a default weight set after which we allow three rounds of relevance feedback before measuring average precision and Pr-50.

4.4.4 Performance after the first step of NN^k search

It is one of our hopes that by covering all possible feature combinations in the first step, the NN^k are more likely to contain at least one relevant image than were we to retrieve with only one default weight set. Because we do not impose a particular interpretation on the query image, we should be more likely to gather at least one representative of the initially unknown semantic facet the user is interested in, although, and for the same reason, we might not gather many of them. This would place a definite advantage on NN^k search as it increases the scope for positive relevance feedback.

The results for the entire set of 3,533 queries built from the Corel collection provides circumstantial evidence that this is indeed the case. Let n_{NN^k} (n_{base}) be the number of times that we find at least one relevant image among the NN^k (baseline) but not among the baseline (NN^k). We find that indeed $n_{\text{NN}^k} > n_{\text{base}}$ although with $n_{\text{NN}^k} = 755$ and $n_{\text{base}} = 734$ the difference is not statistically significant as suggested by a χ^2-test with one degree of freedom. We also find that while the set of NN^k is more likely to contain at least one relevant image, the average number of relevant images is greater for the baseline search (albeit only by 0.29).

4.4.5 Performance after the second step of NN^k search

Although the total number of relevant images returned from the first step of NN^k search is smaller than that for a baseline search, we hope that the few relevant images are nonetheless sufficient to outperform the baseline search in the second step, and possibly to exceed the performance that can be attained when retrieving with the best possible weight set. The latter could be achieved for individual queries whenever there are at least two relevant images among the set of NN^k so that the second step has more than one weight set at its disposal. We also hope to beat if not the optimal performance so

	Corel		Getty	
	MAP (%)	Pr-50(%)	MAP (%)	Pr-50 (%)
Baseline	2.29	1.58	2.95	1.64
Oracle	5.26	2.81	6.41	2.92
RUI	2.49	1.66	3.17	1.71
NN^k Round Robin	4.45	2.12	5.76	2.28
NN^k Borda Fuse	3.43	1.83	4.55	2.02
NN^k CombSum	3.79	1.95	4.91	2.14

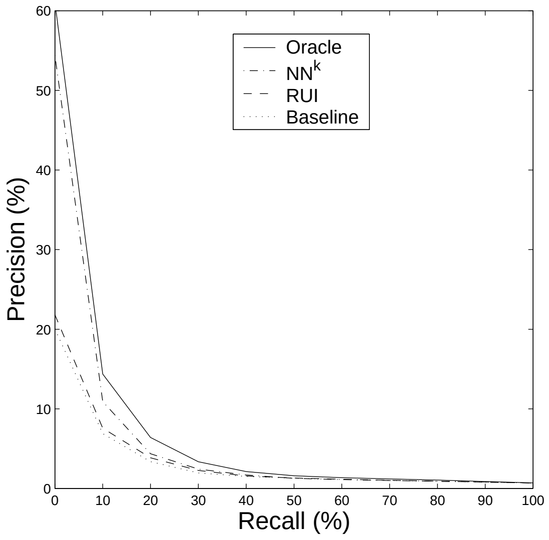
Table 4.2: Performance comparison between NN^k search with three different fusion techniques.

at least the alternative update method by Rui and Huang (2000). The main results are compiled in Table 4.2. It displays the absolute performance averaged over 500 queries for the baseline search, the optimised search, Rui's method, and NN^k search with three of the merging methods introduced in Section 4.2.2. It seems that all our claims are borne out by the experiments. We can make a number of specific observations:

1. Performance figures are strikingly similar for the two collections. This should at first surprise as the Corel collection contains more than three times as many images with relevance classes that are only marginally larger than for Getty. This alone should render the retrieval task more difficult. It appears, therefore, that the greater variability of images within the relevance classes of the Getty collection makes up for its smaller size and that it does indeed constitute a more challenging collection for retrieval.

2. From among the three merging techniques, the Round Robin method consistently achieves the greatest performance gains. The reason for the superior performance of Round Robin can be seen in the fact that, unlike Borda Fuse and CombSum, the final rank assigned by Round Robin to an object is only very weakly correlated with the average rank it occupies in the different lists to be merged. Averaging ranks or scores, which is the effect of CombSum and Borda Fuse, appears sensible in situations where we do not want extreme individual ranks to dominate the final ranking. That they are less helpful in the context of image retrieval may result from the observation that relevance classes are composed of several groups of visually similar images with little visual coherence between groups (e.g. Kim and Chung, 2003). If the selected NN^k happen to come from different groups, we expect the different weight sets to be particularly good at retrieving more relevant images from the respective group. Relevant images, therefore, are expected to be ranked high for some weight sets and low for others. The Round Robin scheme ensures that such images still receive a high overall rank. Its good performance may be construed as circumstantial evidence in favour of our assumption about the visual structure of relevance classes.

3. Irrespective of the fusion technique, the performance of NN^k search lies markedly above the baseline and reaches close to the optimal performance for the best fusion technique. The differences to the baseline is statistically significant, but so is its difference to the optimal performance (for both we find $p < 0.001$ under a paired t-test). That NN^k search does not succeed in closing the gap fully might result from the fact that for many queries the set of NN^k includes but one relevant image so that the best single weight set performance is indeed the best we can do.



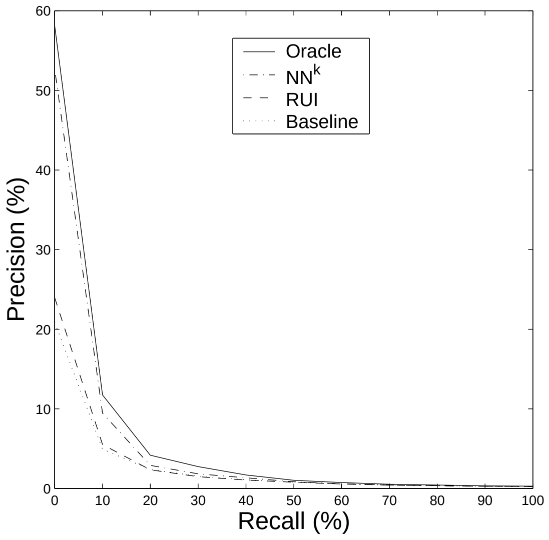


Figure 4.7: Precision-recall graphs averaged over 500 queries for Getty (left) and Corel (right).

Nevertheless, the difference is rather small and the scope for single weight set relevance feedback techniques to outperform NN^k search is negligible. This is therefore strong evidence in favour of the NN^k search methodology.

4. As further confirmation of the previous observation, we note that the performance figures for the well-known method by Rui and Huang (2000) are dwarfed by those of NN^k search. In fact, the weight update according to Equation 4.2 does not seem to lift performance much above the baseline and, indeed, with $p = 0.066$ the difference in Pr-50 is not significant under a paired t-test.

Visual confirmation of these claims comes from the precision against recall curves in Figure 4.7 obtained by averaging over the 500 queries. Again, the two collections afford the same picture. In both cases the performance of NN^k search comes tantalisingly close to the optimal performance and leaves little room for further improvement by alternative weight update methods.

4.4.6 The computational cost of NN^k search

The above measures of retrieval performance, mean average precision and Pr-50, give little indication on their own of the rate at which relevant images are encountered during the search process. Depending on the application, users may not be willing to wait longer unless the results are considerably better. The rate depends on retrieval performance as well as the computational cost. The latter, therefore, becomes an important additional axis along which performance ought to be measured. We have kept the two separate because computational cost is much less fixed and can be reduced by optimisation, approximation, parallelisation and advances in hardware technology.

If we assume that all features and distances have been precomputed (i.e. the query image is part of the collection), the first step of NN^k search is computationally the most expensive part. If we apply the optimisation techniques proposed in Chapter 3, NN^k determination takes slightly more than five seconds for $31,997$ images), a grid resolution of four and eight features (in which case there are 715 grid points that need to be visited). Given the substantial gain in performance, we believe that this cost is acceptable. Note also that the way the NN^k are computed offers the possibility of returning them to the user whilst the grid search is being performed. Contrast this to k-nearest neighbour search where the final list can be determined only after the entire collection has been scanned.

For each grid point, NN^k determination involves applying Equation 4.1 to each of the N images in the collection and sorting the resulting set of distances. The complexity is therefore of order $O(N \log N)$. We will see in Chapter 7, that the great majority of an image's NN^k are ranked high by at least one feature. This means that we can greatly reduce the complexity by only considering the union of highly-ranked images. This requires us to sort the feature-specific distances for each feature once prior to the grid search. Since the union of highly-ranked images has a size that is virtually independent of the size of the image collection, grid search acquires constant-time complexity.

If we want to cater for queries from outside the collection, we will have to perform feature extraction and distance computation on the fly. For large collections and many high-dimensional features, distance computation incurs a serious overhead but one which relevance feedback techniques in general are affected by equally. These observations are summarised in Table 4.3 for a 3GHz processor.

Step	Time (sec)	Notes
Feature computation	1.45	
Distance computation	1,601	L_1 metric between 4,457 floats per image
NN^k determination	5.52	715 grid points
Ranked lists computation	0.016	per positive example
Ranked lists merging	0.153	per positive example

Table 4.3: Computational cost of NN^k search for 31,997 images and eight features.

4.5 Conclusions

This chapter has seen the first application of the NN^k idea to content-based image search. The proposed method provides an effective solution to the problem of combining distance scores from different features. The standard approach involves some form of relevance feedback given on a set of images that have been retrieved with a default weight set. Effective relevance feedback relies on the presence of relevant examples among the first retrieval result. With crude initialisation, however, this is only the case as long as the collections are sufficiently small and queries sufficiently simple.

NN^k search addresses the problem by essentially avoiding parameter initialisation altogether. Instead of retrieving with one default weight set, the user is shown all the images that are most similar to the query under some combination of features. These are just the NN^k introduced in Chapter 3. The set of NN^k of an image provide an overview of the possible semantic facets that a user may wish to search for. In addition to covering a wider range of possible image interpretations, and thus being more likely to capture the semantic facet that is important to the user, the first step of NN^k search makes it possible to associate each of the NN^k with a characteristic weight set, which is the average weight set under which that NN^k is closest to the query. Therein lies the principal strength of the method: by selecting relevant images from among the set of NN^k, users implicitly select weight sets which can subsequently be used for retrieval. Since the weight sets have retrieved the selected images at the top, we should hope that it retrieves more images of the same kind further down the list. This intuition is amply vindicated by our experiments on two large image collections. For both collections retrieval performance gets close to the best single weight set performance and consistently outperforms the well-known method for updating feature weights proposed in Rui and Huang (2000).

The computational cost of the first step of NN^k search is significant but certainly an acceptable price to pay for the enhanced performance. One problem with the proposed method is that it does not offer much help when the first retrieval result does not include relevant images; the best we can do in those circumstances is to provide the user with a ranked list induced by a default weight set. In the chapters to come we will therefore advance beyond the framework of search by example in general and NN^k search in particular, and construct image networks that can be browsed. The structure that we will propose and begin to explore in the next chapter is firmly rooted in the NN^k idea and naturally suggests itself as a powerful continuation of NN^k search.

Chapter 5

NN^k Networks

5.1 Introduction

The previous chapter introduced NN^k search as a two-step relevance feedback method for learning feature weights given a query image. The key idea was not to impose a single metric when retrieving the first set of images but to display instead the NN^k of the query as introduced in Chapter 3. These can be thought as the nearest neighbours to the query image in different feature directions and thus provide a concise pictorial representation of the different semantic facets that users may be interested in and that the chosen feature set can represent. The additional advantage of computing neighbours in this manner is that we can associate each NN^k with an optimal metric, which is the average metric for which that image is the NN^k. Selecting relevant images in the second step thus yields a set of weights that can subsequently be used for retrieval. The first step provides an overview of the possible semantic facets of the query image while the second step precipitates what we called a semantic collapse.

The method proves effective whenever there is at least one relevant image turning up in the first step. But for very large collections or difficult queries with no relevant image being among the NN^k, the method provides little scope for improving search results. Another limitation, which the method shares with most relevance feedback methods, results from its objective to find a global metric. When a relevance class forms a complex distribution in feature space, any such metric will often be locally suboptimal and will not model the metric structure of the class as a whole. NN^k search is an exciting first application of our NN^k idea but in clear need of extension.

In this chapter we will propose a structure that addresses the principal limitations of NN^k search. It is motivated by the observation that if the set of NN^k of a query image does not include any relevant objects, it will contain images that are at least visually closer to the target. Instead of triggering a semantic collapse by selecting a particular NN^k, we may instead compute a new set of NN^k for the selected image and repeat this process as often as the user wishes. Although there is no methodological difference between computing the NN^k for a query image and computing them for an image from the collection, the latter can be precomputed so that we avoid the computational burden of NN^k search at run-time. By determining the set of NN^k for every image from a collection we effectively generate a hyperlinked network in which all images are connected to their NN^k. Image search then takes the form of navigating through a static network structure. We call these structures NN^k Networks and shall in this chapter begin to examine some of their properties. We will delve more deeply into network analysis in the four ensuing chapters.

This chapter is organised as follows. In Section 5.2 we describe the process of computing NN^k Networks given an image collection and a set of image features. Section 5.3 sets the structure apart from the other browsing structures that we encountered in Chapter 2. Section 5.4 begins a first characterisation of NN^k Networks in terms of global properties. In Section 5.5 we describe the basic mode of interaction with NN^k Networks and illustrate with one example how quickly NN^k Networks allow navigation across a collection.

5.2 NN^k Network construction

Given a collection of images and a set of image features, we construct an NN^k Network by identifying vertices of the graph with individual images and by establishing arcs between every image and all their NN^k. NN^k Network construction thus involves image-wise NN^k computation as the central, computationally most expensive step. Recall from Chapter 3 that an image p is an NN^k of q if and only if there exists at least one convex combination of feature-specific distances $d(p, q)$ for which p has minimal distance to q. We refer to the image whose NN^k we have computed as the focal image and write $q \rightarrow p$ to indicate that q is a focal image of p. The NN^k relationship is not generally reflexive so that $q \rightarrow p$ does not imply $p \rightarrow q$. The resulting graph is therefore directed.

The connectedness of individual vertices is only part of the information we obtain from NN^k computation. For each NN^k we also determine the fraction of weight combinations for which it is closer to the focal image than is any other image from the collection. We call this the support of the respective NN^k. The support may be viewed as a measure of similarity between the focal image and its neighbours: The more feature combinations support the NN^k relationship, the more similar is the NN^k to the focal image. The support can be incorporated into the NN^k Network structure by letting it play the role of arc weights. NN^k Networks are thus completely specified by storing for each image the indices of all its NN^k, and for each of the latter its support. By storing the NN^k in decreasing order of support we are able to quickly retrieve the set of most similar NN^k.

The storage requirements are moderate. As an example consider a collection of 100,000 images with an average of 100 NN^k per image. The corresponding NN^k Network requires $10^5 \times 50 \times 8$ bytes, that is 40 MB of storage. By limiting the number of NN^k that are stored for each image, storage costs increase linearly with the number of images, otherwise the relationship would be log-linear according to Figure 3.8 in Chapter 3.

We will find it useful for Chapter 8 to also store for each arc the characteristic weight set of the respective NN^k. Recall that this weight centroid was the key to finding feature weights in the second step of NN^k search. This adds another 4 bytes for every feature so that for ten features our example collection now claims $10^5 \times 50 \times (8 + 4 \times 10)$ bytes or 240 MB.

The time complexity of NN^k Network construction is $O(N^2)$ where N is the number of images in the collection. The quadratic relationship is imposed by the requirement to compute the pair-wise distances between any two images. The largest collection we have indexed comprises 100,000 images. This can be achieved in around a week for 8 features on a single 3 GHz processor. Both distance computation and NN^k computation can readily be parallelised which should provide a practical solution for collections of a few million images. For much larger collections, however, the quadratic complexity will need to be tamed. For low dimensional spaces, k-nearest neighbour search can be accelerated through the use of hierarchical index structures such as R trees (Guttman, 1984) but for the dimensionality typically encountered in image retrieval applications, performance of such space partitioning methods degrade rapidly (Weber et al., 1998). An alternative approach that is currently attracting much attention is to approximate the k-nearest neighbours. The Vector Approximation File (VAFile) by Müller and Henrich (2004), for example, each feature dimension is quantised thus allowing a fast initial filter step. In a second step, the complete feature vectors of the filtered set are used to refine the answer. The similar BOND system of de Vries et al. (2002) employs a branch-and-bound algorithm so that data in later dimensions can be discarded. The iGrid of Aggarwal and Yu (2000), the bitmap index of (Cha, 2003) and the local similarity function method by Howarth and Rüger (2005) all work with vertically decomposed features where only those parts of each dimension are considered that are close to the query point. This brief survey only serves to illustrate that notable progress is currently being made to speed up k-nearest neighbour search so that the complexity of NN^k Network construction may be reduced to $O(N \log N)$ or even $O(\log N)$ if k-nearest neighbour search could be achieved in constant time.

5.3 NN^k Networks in context

In this section we shall consider some of the browsing structures that we first encountered in Chapter 1 and work out precisely how these differ from NN^k Networks.

The closest in structure are arguably Cox's nearest neighbour networks (Cox, 1992, 1995). Cox defines a nearest neighbour network of order k as a directed graph such that each object is linked to its k most similar objects under some fixed distance metric. Note that for $k > 2$ most of the connections are between objects that are only near to each other, not nearest. Similarity is measured by applying the cosine similarity measure to a vectorial representation of objects. Browsing takes place by users choosing any of the k neighbours of a node which retrieves the set of neighbours of the newly selected object. NN^k Networks differ significantly from Cox's nearest neighbour networks by explicitly addressing the problem that motivates relevance feedback techniques, in general, and the NN^k idea in particular, namely the dependence of the optimal distance metric on both the user and the query task. Interestingly, Cox notes in his final chapter that "there is no reason why the measures used to find nearest neighbours need be exactly the same in different parts of the network." and suggests that "when we are inserting an object we use the measures associated with the objects that are in the database, but we allow the person doing the insertion to disregard the computer suggestions." (Cox, 1995, pp. 140). Allowing the metric to vary with the object does, however, not solve the problem arising from the empirical fact that even for the same object the metric may vary considerably between different users (see Figure 4.2 of Chapter 4). Moreover, locally optimising metrics through explicit user input becomes impractical for large collections. NN^k Networks address both issues by determining for each image nearest neighbours for a large set of metrics. If we assume the metric to be fixed, the lines of equal distance are the same for every image and, unless we intentionally introduce a bias, they have the same extension in every feature direction. The neighbourhoods of k-nearest neighbour networks are therefore locally more confined than those covered in NN^k Networks. By looking at different feature combinations, NN^k Networks can draw on points that may be quite far by unweighted Euclidean standards. In addition to being unbiased with respect to the underlying features, NN^k Networks thus offer a greater chance to quickly traverse an image collection.

Rubner et al. (1997)'s browsing model displays the neighbourhood of an image by applying multi-dimensional scaling to a distance matrix derived from colour information. While providing a well-motivated principle for displaying image neighbourhoods, the approach does not address the problem of finding object- and user-specific metrics, but nor does it claim to do so. It assumes that colour is generally the best feature for retrieval purposes and places its emphasis instead on image layout and distance computation. Another notable difference of Rubner's model to both Cox's nearest neighbour networks and NN^k Networks is the computational cost of navigating through the structure. Browsing takes the form of iterated query by example. Selecting an image from the retrieved set initiates a new query involving the computation of a new set of k-nearest neighbours followed by layout optimisation. The application of MDS to entire collections might only be meaningfully employed to browse small collections of a few hundred images.

Santini et al. (2001)'s model supports sophisticated techniques for adjusting feature weights so that the distances computed by the system match better those communicated by the user. The process is one of continuous refinement and appears ill-suited to quickly navigate through a large collection of images. Despite appearance the model is in fact more akin to traditional relevance feedback techniques and does not concern itself with the question of how to initialise the distance metric in the first place. Its computationally expensive optimisation step after each feedback iteration might lead to unacceptable response times for large collections. We shall see that NN^k Networks support both fast undirected browsing without a well-defined information need as well as directed target search without paying the price of increased computational complexity.

The ostensive browsing model as conceived by Campbell (2000) and adapted to content-based image retrieval by Urban et al. (2003) lets the user follow paths in a dynamically growing tree structure. The objective is to establish the best query vector given the history of past images on the browsing path. This makes for a more robust method than were we to use only the selected image as the query. More recent evidence is assigned greater weight so that the model is flexible enough to track changing information needs. By incorporating information about the user's immediate browsing history, the model exhibits the interesting property that the same image may lead to different search results. Like the models of Rubner et al. (1997) and Santini et al. (2001), the ostensive browsing model involves the execution of a new query at each step and does not aim to extract different semantic facets of the current image. Although NN^k Networks are precomputed and inherently static, an analysis of the browsing history could help to arrive at more robust estimates of the information need of a user. This in turn could be exploited for automated query formulation and weight estimation, providing a possible synergy between NN^k Network browsing and query by example. This touches on the general question how to exploit implicit relevance feedback provided through browsing in an integrated retrieval framework, an issue that is of great interest but that lies outside the scope of the present thesis.

A number of hierarchical browsing models have been proposed for image retrieval (Krishnamachari and Abdel-Mottaleb, 1999; Swets and Weng, 1999; Chen et al., 2000; Pečenović et al., 2000; Barnard and Forsyth, 2001). Characteristically, the distance metric employed for hierarchical clustering is fixed so that once again the problem of image polysemy is left unaddressed. Browsing takes place not across networks but along tree structures, which are arguably less suited for undirected exploration of information spaces.

NN^k Networks are similar to conventional nearest neighbour graphs, which consist of only those edges that connect a point to its nearest neighbour. Because there is typically exactly one nearest neighbour, these graphs are extremely sparse. NN^k Networks connect each point to all points nearest under any one feature or a linear combination thereof, so NN^k Networks can be viewed as a superposition of k-nearest-neighbour graphs to which a few edges are added by also taking into account feature combinations. We note from Figure 3.8 of Chapter 3 that most of the NN^k are in fact induced by feature combinations. For $k = 8$ and $10,000$ images the number of NN^k is around 30, meanwhile the maximum number of nearest neighbours that would result from a superposition of nearest neighbour graphs is eight and may be less since an image can be the nearest neighbour to another under more than one feature. The individual nearest neighbour graphs therefore contribute only a minority of edges to the rich connectivity structure of NN^k Networks.

5.4 A global view of NN^k Networks

This section is concerned with contrasting NN^k Networks in terms of global properties with random networks on the one hand, and k-NN networks on the other hand. These investigations will provide us with first insights regarding the structural characteristics of NN^k Networks.

5.4.1 Layout optimisation

NN^k Networks represent the local neighbourhood of an image and we therefore expect the graphs to be in some sense structured. For example, if two images p and q share the same neighbour, they are more likely to be connected themselves than in a random network because p and q occupy a similar region in feature space. As argued above, we expect the equivalent k-NN networks to exhibit more structure still because the neighbourhood is spatially even more confined.

We seek to validate this hypothesis by simply visualising the networks. Gaining a visual impression of the structure of NN^k Networks is possible for all but very small collections. Even if every vertex has only a small number of neighbours, the graphs quickly become unwieldy to display. For the purpose of illustration, we have constructed the NN^k Network for 100 images from the Corel collection described in Chapter 4. For each vertex we only record the two NN^k with the largest support. In addition, we construct a random network where every image is connected to two randomly chosen images. A 2-NN network is constructed by finding for each image the two nearest neighbours under an unweighted Euclidean metric.

The graphs are displayed by randomly assigning vertices to regularly spaced positions along the circumference of a circle. To keep the plot simple we omit any indication of the direction of the edges. The results are shown in the top row of Figure 5.1 (left: random network, middle: NN^k Network, right: k-NN network).

The three graphs look essentially the same with any differences in detail easily attributable to chance. But it should readily be seen that the random arrangement of vertices to positions introduces a considerable amount of noise that would certainly help to hide any significant differences if they existed. We can control for this effect by subsequently repositioning vertices with the goal of optimising the layout. If networks preserved neighbourhood structure than we ought to be able to flatten the network in a manner analogous to that of unwrapping a manifold in a vector space. We can attempt such linearisation by repeatedly excising a segment of variable size from the circumference and inserting it at another position. If the total edge length is thereby reduced, we keep the new configuration and continue. The results are shown in the second row of Figure 5.1 in the same order as before. With the noise thus reduced, notable differences have now surfaced. Both the NN^k Network and the k-NN network tend to concentrate a substantial proportion of edges very close to the circumference resulting in a much smaller outer shell than for the random network. Also, the number of long-distance edges is visibly larger in the latter. We also find confirmation of what we have repeatedly been emphasising, namely that the neighbourhood represented in k-NN networks is somewhat tighter than that of NN^k Networks. The latter appear to lie structurally somewhere between random graphs and highly structured k-NN networks.

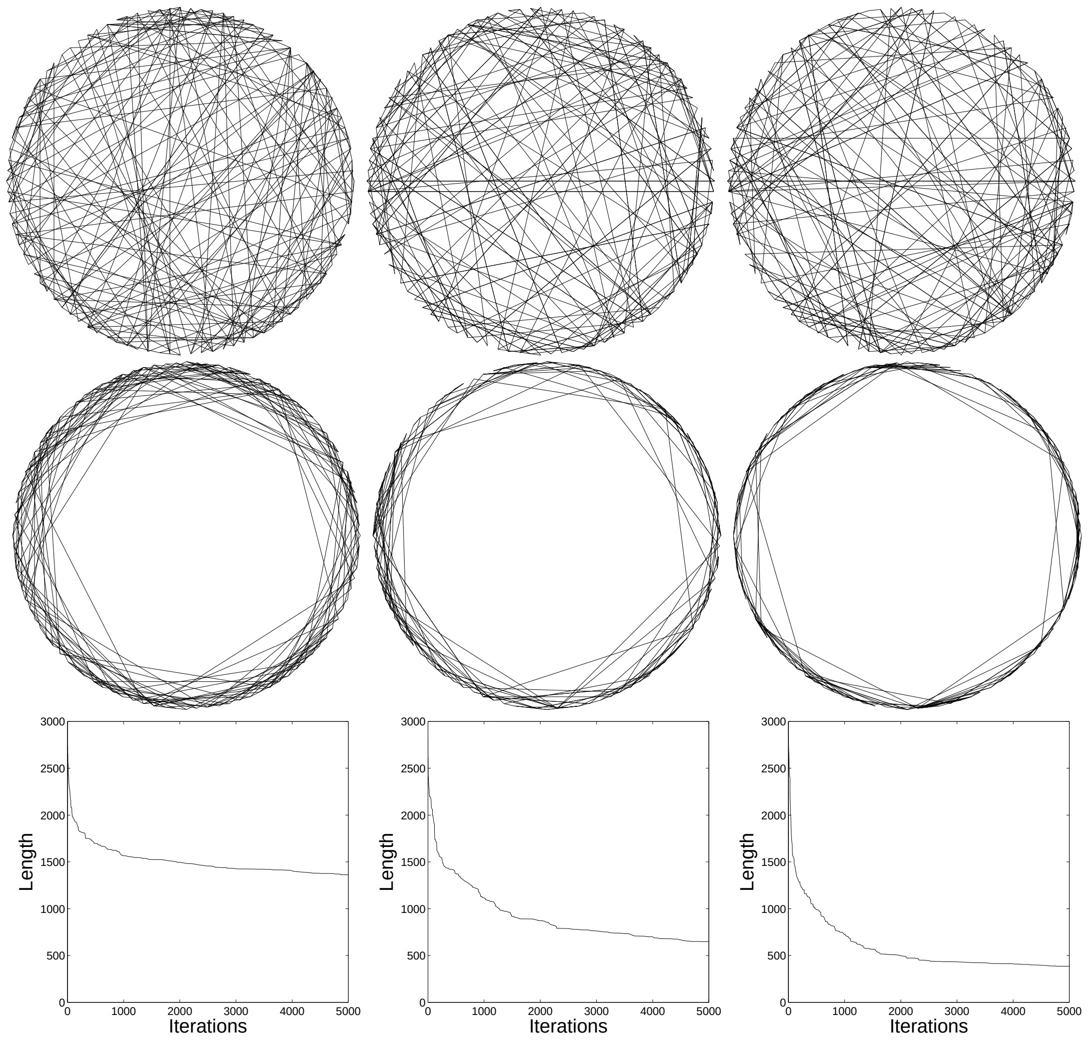


Figure 5.1: Optimised layout for a random network (left), an NN^k Network (centre) and a k-NN network (right).

5.4.2 Bandwidth minimisation

A network with n vertices is completely specified by its $n \times n$ adjacency matrix where $a_{ij} = 1$ if there is an arc from vertex i to vertex j and $a_{ij} = 0$ otherwise. In general, we expect to be able to represent more structured graphs by adjacency matrices with a larger number of non-zeros entries near the diagonal. In the extreme case of a cycle graph, for example, we can order the vertices such that the adjacency matrix is zero except for the sub- and superdiagonal. In Figure 5.1 ordering vertices according to their positions along the circle in the optimised layout would give a similar effect. Changing positions of vertices on the circle corresponds to permuting rows and columns of the adjacency matrix. Such permutations can be achieved in a more systematic way than described above. The reverse Cuthill-McKee algorithm (Cuthill and McKee, 1969; George and Liu, 1981) is an iterative, greedy procedure that reduces the profile or bandwidth of a matrix. The bandwidth is defined as the maximum number of entries in any row that separate the first from the last non-zero entry where the diagonal entries are by convention treated as non-zero. The algorithm is not guaranteed to find the smallest possible bandwidth but it usually does. The result is a matrix in which adjacent vertices of the graph tend to correspond to nearby rows. We have applied this technique of bandwidth minimisation to the three types of networks considered in the previous section. Each one now contains 1,000 images from the Corel collection and the networks are generated such that each image is connected to at most five vertices with the number of arcs for a particular image being the same across all three networks. The results are shown in Figure 5.2. The top row shows the original adjacency matrices of a random network (left), the NN^k Network (centre) and the k-NN network (right). The bottom row shows the optimised adjacency matrices after application of the reverse Cuthill McKee algorithm.

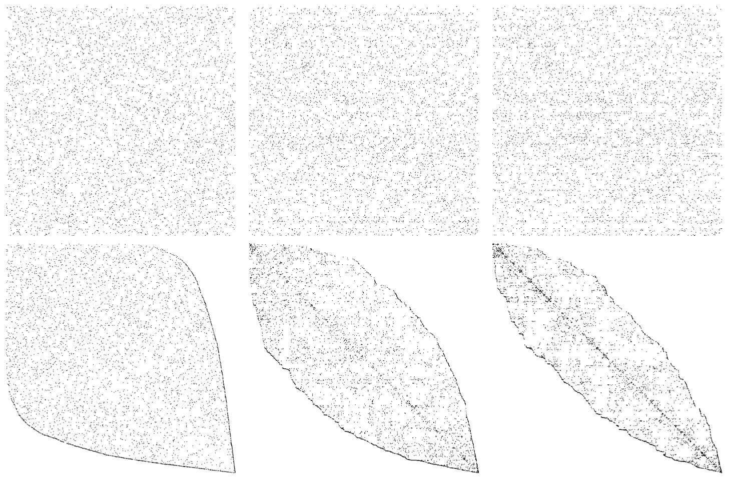


Figure 5.2: Bandwidth minimisation for a random network (left), an NN^k Network (centre) and a k-NN network (right).

Despite appearance, the matrices are exceedingly sparse with only 5,000 of the 1,000 $\times$ 1,000 entries being non-zero, that is 0.005 %. That this does not show in the plot results from the fact that the resolution of the plot is lower than what the size of the matrix demands (if this were not so, little could be discerned). The respective bandwidths for the three graphs are 940, 768 and 591. Even without knowledge of the precise figures, it is evident from the plots that the random network lends itself much less to bandwidth reduction than the other two. It confirms our observation from Figure 5.1 that NN^k Networks are somewhat less structured than their close counterpart.

Incidentally, we also note that compressing the byte representation of the adjacency matrices using the popular Lempel-Ziv encoding algorithm as implemented in gzip further underlines this observation. The three matrices can be compressed from 1MB to 11,489 (k-NN), 12,498 (NN^k) and 13,694 (random) bytes, respectively.

5.5 Interaction with NN^k Networks

Variation in conjunction with non-random selection is indispensable for progress. It forms a cornerstone of Darwin's theory of natural selection taking up the full first two chapters of his book. It provides cultural evolution with its building blocks and provides the key to understanding the effectiveness of the mammalian immune system. Unsurprisingly, variation plays a pivotal role in the theory of genetic algorithms where a solution is progressively improved on by examining and possibly selecting variations thereof. It should therefore appear natural to formulate the problem of image retrieval in terms that are more amenable to the operation of variation and selection.

NN^k Networks can be viewed as one possible solution. We can think of them as structure on which the principles of variation and selection are continuously operative. Variation is inherent in the network while selection takes place by users choosing neighbours. The fact that variation is always with respect to the current focal image is what makes cumulative improvement possible. Unlike genetic algorithms where a well-defined objective function allows automating the selection step, search across NN^k Networks relies critically on user interaction. The criterion by which users are expected to select neighbours is their degree of visual similarity to the target image. This may be an image that either forms part of the collection or exists merely in the user's mind. We believe that this selection criterion comes natural to most users in spite of the fact that it may not customarily be applied in other contexts.

We will take a closer look in Chapter 7 at the effectiveness of NN^k Networks for directed search. In the same chapter, we will also consider the question of how to provide initial access to the networks. For now we shall content ourselves with illustrating with one example how quickly users may navigate through NN^k Networks on the basis of visual cues starting at some initial image. The network from which the paths in Figure 5.3 were obtained is based on 1,000 images from the Corel collection indexed with 8 features. The initial image displayed topmost is the photograph of a wading bird. From it radiate a number of different paths, any one of which is a realisation of a possible browsing sequence. The first five images around the top-most image are a subset of the latter's NN^k. Each of the displayed images are nearest to the initial picture under some weighted combination of features-specific distances. The four images on the right contain some element of water and a fair proportion of blue but they differ markedly in composition and the relative frequencies of colours. The leftmost image is very different in colour from the initial picture and is likely to be among the NN^k on account of its textural resemblance. We note that the first-degree neighbours of the initial image already form a rather diverse set. The second-degree neighbours lead us even further into the network. Each of the images is one of the NN^k of the image to which it is connected above. Thus, selecting the sandy beach would produce the skyline among the next set of images. For the central three images we allow the browsing path to continue one more step.

Three observations can be made. First, we observe that successive images along any one browsing path differ only relatively little. They form a natural sequence and we can imagine how users may home in on their target through a sequence of small and relatively intuitive steps. Secondly, the images at which we arrive after only two to three steps along the different browsing paths could hardly be more different. In fact, in this particular example five steps are sufficient to reach any image in the collection. Three steps cover 92% of the collection. Thirdly, as we move along a particular browsing path, the importance of different features may change. As an example, the sunset scene on the leftmost path owes its NN^k status to texture similarity, meanwhile the rose one step further into the network

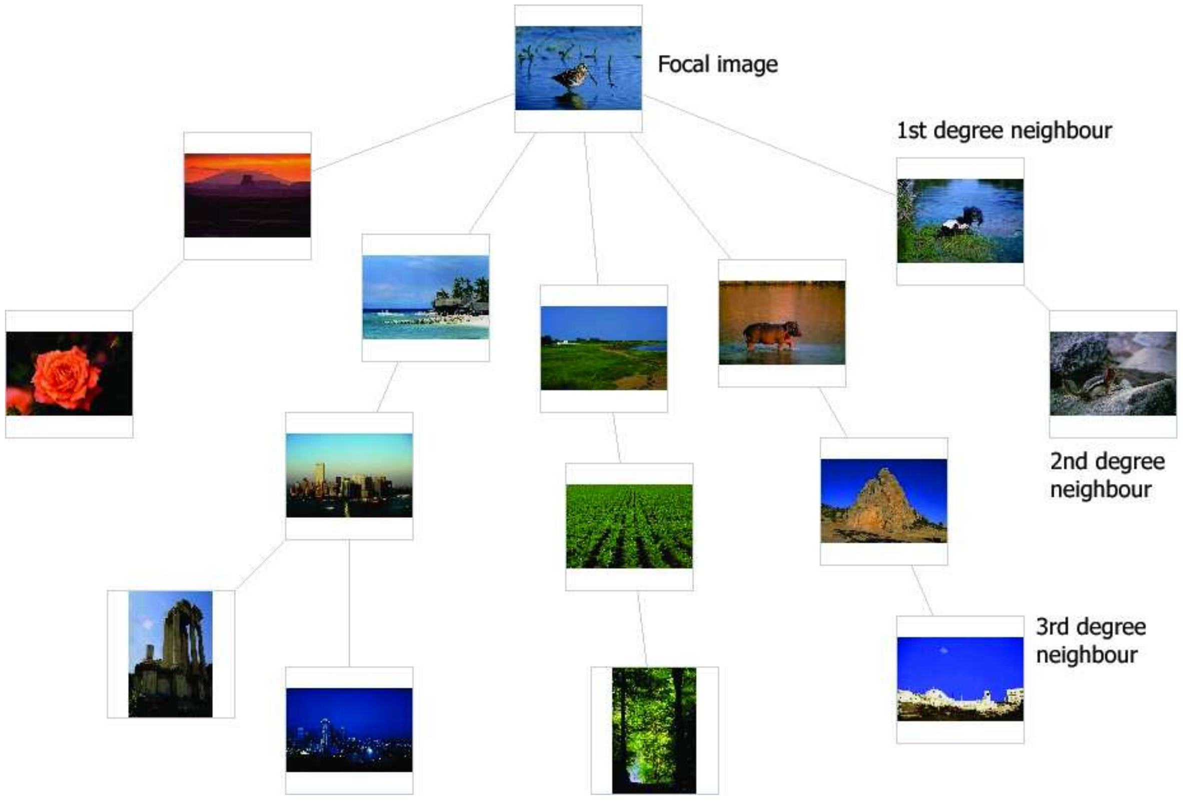


Figure 5.3: A subset of the first, second and third degree neighbours of an image (top)

is connected by virtue of its similar distribution of hues. Almost the opposite can be observed along the rightmost path. The first-degree neighbour is very similar in many respects to the initial picture while the second-degree neighbour has little colour in common with its predecessor but all the more texture. This is of course the defining characteristics of NN^k: they aim to cover the feature space in all directions and ensure that users can always move quickly between different regions in feature space.

5.6 Conclusions

Building on many of the ideas developed in chapter 4, this chapter has introduced NN^k Networks as a novel browsing structure for interactive image retrieval. We have emphasised how NN^k browsing can be viewed as a natural extension of NN^k search. By not imposing a particular metric NN^k Networks capture in a very compact way the multitude of possible meanings a user may want to associate with the images. We have distinguished our work from other recent approaches to image browsing, most notably the work by Cox (1995), Santini and Jain (2000), Rubner et al. (1998), Campbell (2000) and Urban et al. (2003). We believe that there are at least three major advantages of NN^k Networks. (i) Unlike other browsing solutions (Cox (1995) being a notable exception), NN^k Networks are entirely precomputed. NN^k Network browsing is not iterated search in disguise and can therefore be carried out very fast irrespective of the collection size. (ii) With NN^k Networks we adopt the same agnostic view that underpins NN^k search. The networks provide an unbiased view of an image by connecting it to all images that are most similar to it under any fixed set of features. If the features are any good, some of the images it is connected to will correspond, or correspond more closely, to the image the user has in mind. NN^k Networks therefore increase the chance that a particular semantic facet is found among the neighbours of an image. (iii) Because of the diverse nature of the set of NN^k, NN^k Networks allow users with no particular information need to quickly navigate through the collection.

We have taken a brief look at toy networks generated for small collections and examined simple global properties using layout optimisation and bandwidth minimisation. Both measures suggest that NN^k Networks lie structurally between random graphs and k-NN networks. The chapter has given rise to a number of questions: How well connected are NN^k Networks? How many edges separate two images on average? How are images from the same class distributed across the network? By answering these questions in the next chapter, we shall hope to substantiate many of the claims this chapter has made.

Chapter 6

NN^k Network Analysis

6.1 Introduction

Networks can exhibit a wide variety of structure. Sometimes their structural peculiarities offer us deep insights into some of the causal aspects involved in their genesis and can help us understand the behaviour of the physical systems which they represent. Indeed, notable success has been achieved in recent years in modelling seemingly complex networks in terms of relatively simple generative rules (Bornholdt and Schuster, 2003; Newman, 2003; Wolfram, 2004) with the promise of being able to predict the effects of perturbations on network structure and function. NN^k Networks provide us with a structure where the basic rules are already known prior to the analysis: the construction principle of NN^k Networks is encapsulated in Equation 3.1 of Chapter 3 and it is therefore not the rules of growth we seek to understand but the structural characteristics of the resulting networks insofar as these are of functional significance.

We take two complementary approaches in our dissection of NN^k Networks. We call these, respectively, the formal and the semantic analysis. The formal analysis abstracts from the meaning that we attach to vertices and edges. It is blind to the relevance relationships that a user would see between images and exposes the topological characteristics of the networks. The semantic analysis incorporates the notion of relevance as an important additional axis of analysis. It studies how semantically related images are distributed across the network and to what extent the distribution differs from that of a random subset of images. Both types of analysis provide important evidence in support of our conviction that NN^k networks organise images in ways that lend themselves well to image browsing and directed search. In particular, we set out to test the following three hypotheses:

1. NN^k Networks are highly connected so that from any vertex users can browse to all others.
2. NN^k Networks have a small diameter so that any two vertices are separated only by a few edges.
3. NN^k Networks arrange relevant images in connected subgraphs so that users easily find more relevant items if they have found one.

As our analysis makes extensive use of terminology from graph theory, it is convenient to begin the chapter by reviewing some elementary graph-theoretic concepts and notations.

6.2 Preliminaries

A graph G is a set of vertices $V = V(G)$ and a set of edges $E = E(G)$ connecting any two elements in V. An edge joining vertices x and y is denoted by xy if it is undirected, and by $x \rightarrow y$ if it is directed. Directed edges are also referred to as arcs. If $xy \in E(G)$ then x and y are adjacent or neighbouring vertices of G. We assume that a graph does not contain multiple edges or loops. The order of G is the number of vertices and is denoted by $|G|$. The size of G is the number of edges and is denoted by $e(G)$. $G(n, m)$ denotes an arbitrary graph of order n and size m. The set of vertices adjacent to some vertex $v \in G$ is denoted by $\Gamma(v)$. The degree of x is $d(x) = |\Gamma(x)|$. The density of a graph is the proportion of edges relative to the maximum number of possible edges, i.e. $m/\binom{n}{2}$. $G' = (V', E')$ is a subgraph of $G = (V, E)$ if $V' \subset V$ and $E' \subset E$. In this case, we write $G' \subset G$. If G' contains all edges of G that join two vertices in V' then G' is said to be the subgraph induced or spanned by V' and is denoted by $G[V']$. A tree spans a graph $G$and is referred to as a spanning graph if it includes all the vertices of G.

A path is a graph P of the form

$$V(P) = x_0, x_1, \dots, x_l, \quad E(P) = x_0x_1, x_1x_2, \dots, x_{l-1}x_l$$

and is usually denoted by $x_0x_1 \dots x_l$ where x_0 and x_l are the end vertices of P and $l = e(P)$ is the length of P. We say that P is a path from x_0 to x_l. Given vertices x and y, their distance $d(x, y)$ is the minimum length of an x, y path. If there is no such path then $d(x, y) = \infty$. For undirected graphs we have $d(x, y) = d(y, x)$. This does not generally hold for directed graphs. The diameter of a graph is the largest distance between any two vertices.

A graph is connected if there exists a path between every pair of vertices. Components of a graph are maximal connected subgraphs, that is, they cannot be augmented by any additional vertex without disconnecting the subgraph. A graph that does not contain any cycles is an acyclic graph. Acyclic graphs that are connected are trees, and forests if they are not.

Strongly connected components in a directed graph can be thought of as a partitioning of V into equivalence classes V_i, $1 \leq i \leq r$ such that vertices v and w are equivalent if and only if there is a path from v to w. Let E_i, $1 \leq i \leq r$ be the set of arcs with head and tail in V_i. The graphs $G_i = (V_i, E_i)$ are called the strongly connected components of G.

Random graphs can be defined in terms of two closely related probabilistic models. Starting with a fixed set of distinguished vertices n, we can choose every edge with some probability $0 < p < 1$, independently of the choices of other edges, or else we take all graphs of order n and size M and consider them as points of a probability space having equal probability. We shall only make use of the first definition and write $G(n, P(\text{edge}) = p)$ for the corresponding probability space. We shall also use the idea of random subgraphs. Given a graph $G = (V, E)$, a random subgraph of G is a graph induced by a random subset of V.

6.3 Formal analysis

Graphs are conceptually simple and yet they can differ widely in their structure. In this section we are principally interested in the general connectivity properties of the network and confine our investigations to a select few that we deem to be of particular functional significance in the context of interactive browsing.

6.3.1 Connectivity

Undirected graphs have a simple topological structure. They are either connected or they are not. In the former case every vertex can be reached from any other, in the latter some vertices are separated from others. The directionality of the arcs in directed graphs like NN^k Networks or the World Wide Web (subsequently referred to as simply the Web) allows for a much richer structure.

The five continents of directed graphs

It is instructive to think of a directed graph as being made up of five distinct regions or subgraphs.

- Central core: Unless the graph is very sparse, it is likely to have what has somewhat tautologically been called a central core, a large set of vertices such that each vertex is reachable from any other. The central core is the largest strongly connected component of the graph and is also referred to as the giant component if it is significantly larger than the others. Vertices that do not belong to the central core belong to either of four regions depending on the extent to which they are connected with the central core.
- In-continent: a subgraph containing all vertices from which one can reach the central core through one or more arcs but which cannot be reached from the central core
- Out-continent: a subgraph containing vertices that are reachable from the central core through one or more arcs but do not lead back to the central core.
- Tendrils: a subgraph containing vertices that may either be reached from the in-continent or reach into the out-continent but not both
- Islands: a subgraph containing all the remaining vertices

Within each of the four peripheral classes, vertices may themselves form other strongly connected components. This division is motivated by Barabási (2004) and illustrated in Figure 6.1.

The navigability of a directed graph depends on the relative size of these regions. Evidently of greatest importance is the relative size of the central core. Ideally, we would like to be able to reach any vertex of the graph irrespective of the starting location and would therefore wish the central core to encompass the entire graph. In practice, the central core can be remarkably small. In the Web, for example, it accommodates a mere quarter of all the sites (Broder et al., 2000). It is still the largest strongly connected component and thus qualifies as the central core. The remaining 75% are fairly equally distributed across the four peripheral classes. A staggering 25% of all the sites are estimated to exist as small islands disconnected from the rest of the Web and invisible to search engines. The Web is highly fragmented and we should certainly hope that NN^k Networks fare better. The high degree of fragmentation is ultimately a result of the heterogeneity of the community that is weaving the Web and, in particular, the presence of closed communities where links are placed highly non-randomly. No such fragmentation would be observed in a random graph of the same size and order as the Web.

In search for the central core

Let $G = (V, E)$ be the directed graph corresponding to an NN^k Network. For reasons of navigability we consider it a premium that G is strongly connected, or at least contains a strongly connected

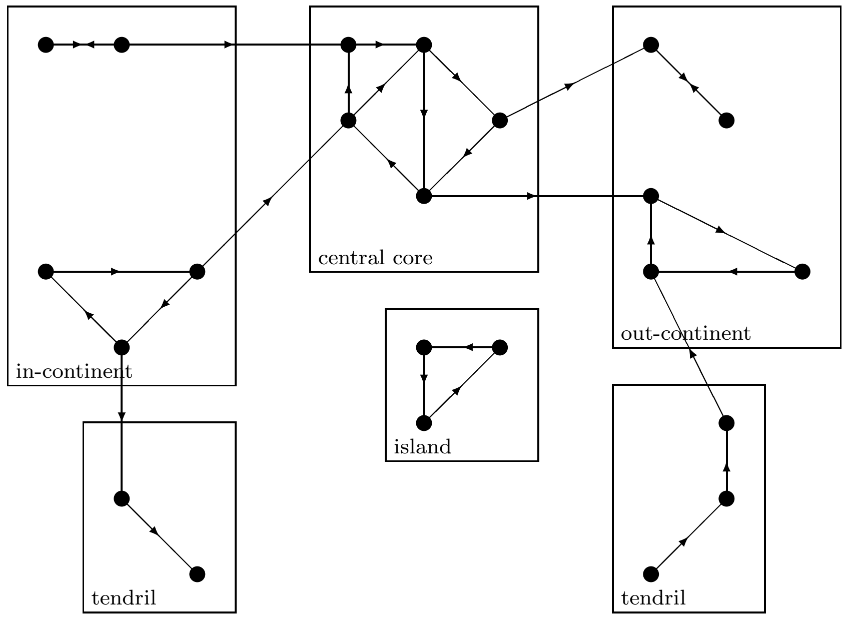


Figure 6.1: The five components of directed graphs.

component that comprises a large proportion of V (the central core). A number of efficient algorithms have been devised to find strongly connected components. The algorithm reported by Aho et al. (1983) involves two depth-first searches of the graph and the determination of the transpose of the adjacency matrix of G. The depth-first search may be written in terms of the following pseudocode

```
1  function dfs(int v) {
2    visited[v] = true;
3    foreach w in NNk[v] {
4      if( visited[v] = false ) {
5        dfs(w)
6      }
7    }
8  }
```

In practice, for large graphs involving several thousand vertices, an iterative version of depth-first search is preferable to the recursive version to avoid stack overflows. The algorithm for finding strongly connected components is rather quite elegant and involves the following steps:

1. Perform a depth-first search on G and number the vertices in order of completion of the recursive calls (after line 5 of the pseudocode)
2. Construct a new directed graph G_r by reversing the direction of every arc in G.
3. Perform a depth-first search on G_r starting the search from the highest-numbered vertex according to the numbering assigned at step (1). If the depth-first search does not reach all vertices, start the next depth-first search from the highest-numbered remaining vertex.
4. Each tree in the resulting spanning forest is a strongly connected component of G.

The central core in NN^k Networks

We have identified the size of the central core for a variety of NN^k Networks. Without exception, the central core contains the vast majority of vertices (Table 6.1). NN^k Networks hence appear to be highly navigable. This is in striking contrast to the equivalent figures obtained for the Web. There are

Collection	Size	Coverage of Core	Number of SCC	Order of 2nd largest SCC
Corel	500	1	1	0
Corel	1,000	0.999	2	2
Corel	5,000	0.999	2	2
Corel	10,000	0.998	13	2
Corel	20,000	0.997	11	2
Corel	31,997	0.996	76	2
TVID2004	31,019	0.959	204	62
Getty	8,202	0.999	7	2

Table 6.1: Strongly connected components in NN^k Networks.

two reasons for the difference. First, by construction of the network, every image must be connected to at least one other. This condition eliminates many fragmented networks from the outset. For example, the directed graphs displayed in the bottom row of Figure 6.2 are ruled out by construction. Of the 16 topologically distinct networks that can be generated with three vertices, only the top five networks in Figure 6.2 satisfy the connectivity constraint imposed by NN^k Networks. It turns out that it is only these five that are also strongly connected. The fact that every image connects to some other ensures that in NN^k Networks there are no dead ends, even though eternal cycles there may be. The second reason lies in the very simplicity of NN^k Network construction. Links are established based solely on low-level feature similarity and do not depend on the social structure of a Web community with their natural fragmentation into interest groups. The good connectivity properties give us an indication that NN^k Networks may be somewhat less structured than the Web.

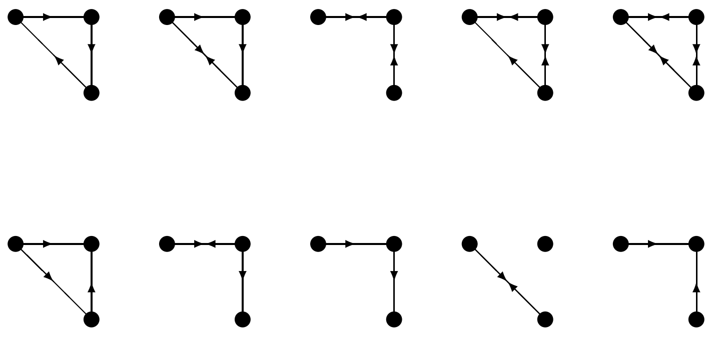


Figure 6.2: Permissible (top) and non-permissible (bottom) NN^k Networks of order three.

We note from Table 6.1 that all strongly connected components that are not part of the central core are typically very small for most collections. Often they are made up of two or more near-identical images that link to each other and to no other vertex. The video collection is unique in exhibiting a comparatively high degree of fragmentation with 204 components in addition to the central core. The reason lies in the large number of repeated sequences (e.g. adverts, news). In addition to small clusters of highly similar images, a component may also consist of only one image. Because each image has outlinks, the reason for forming a component on its own is that no other image links to it. These are images that deviate markedly and singularly from the rest of the images. For a user these disconnected islands are invisible since they are unreachable. The islands made up of two images present themselves differently however: they can be reached but allow no escape. In Chapter 7 we will suggest possible ways to modify the network to avoid these islands altogether.

The birth of the giant component

When edges are added at random to n initially disconnected points, a remarkable transition occurs when the number of edges becomes approximately $\frac{n}{2}$. Erdös and Rényi (1960) studied random graphs with n vertices and $\frac{n}{2}(1+\mu)$ edges as $n \to \infty$ and discovered that such graphs almost surely have the following properties: if $\mu < 0$, only small trees and unicyclic components are present, where a unicyclic component is a tree with one additional edge. Moreover, the size of the largest tree component is $[\mu - \ln(1+\mu)]^{-1} \ln n + O(\ln \ln n)$. If $\mu = 0$, however, the largest component has size of order $n^{2/3}$. And if $\mu > 0$, there is a unique giant component whose size is of order n. To simulate the evolution of a random graph, we simply vary the probability p in $G(n, P(\text{edge}) = p)$ and thus the average degree of a vertex. In NN^k Networks we can control the average vertex degree by varying the number of features k: as more features are added, the average vertex degree goes up (see Figure 3.8 of Chapter 3). For a collection of 20,000 images, we have constructed different NN^k Networks with k varying from two to eight and determined for each network the relative size of the largest component and the average vertex degree. The relationship is depicted in Figure 6.3 where we have indicated the number of features next to each data point. As we discovered in the previous section, for eight features the relative size of the giant component is comfortably close to one. But even for much fewer features this holds true. For $k \geq 3$ the largest component comprises more than 90% of the network. This assures us that the good connectivity properties of NN^k Networks do not rely on a large number of features. In the light of these findings, our first hypothesis seems amply vindicated.

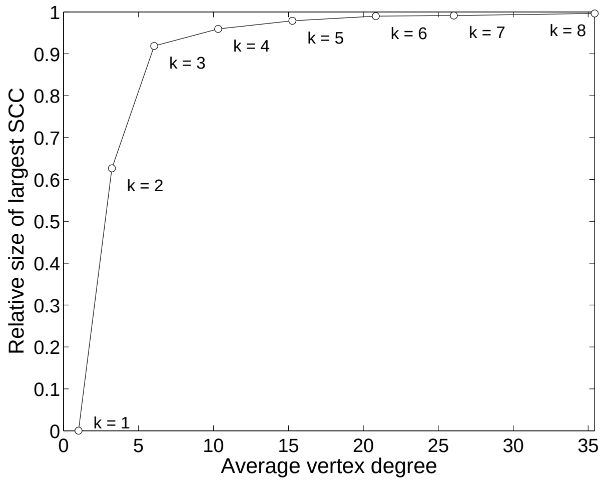


Figure 6.3: The giant component forms at a connectivity somewhere between 2 and 3.

6.3.2 Small-world properties

Watts and Strogatz (1998) explored how the connection topology of an initially highly regular graph changes as more and more of its edges get rewired at random. An example of such a regular graph can be constructed by arranging vertices in a circle and connecting each vertex with each of its k-nearest neighbours. The procedure of rewiring the original graph involves taking each vertex in turn and for each of its edges choosing a different end-vertex with probability p. As p varies between one and zero, the graph topology changes from regularity to randomness.

The authors were particularly interested in two properties: the average distance between two vertices and the average clustering coefficient. The latter property was first motivated in that same paper. It quantifies the extent to which neighbours of a vertex are themselves neighbours and thus provides a measure of what me may intuitively recognise as the cliquishness of a graph. Formally, given a graph G without loops and multiple edges and a vertex v, the local clustering coefficient at v is given by the ratio of the number of actual edges between neighbours of v and the maximum number of such edges. The maximum number is given by $[\Gamma(v)(\Gamma(v)-1)]/2$ which is attained if every neighbour of v is linked to every other of v's neighbours. The average clustering coefficient is obtained by averaging the local clustering coefficient over all vertices.

Both the average distance L and the average clustering coefficient C take highest values for very regular graphs and are minimised in random graphs. The particular way in which these properties change as we vary the probability p of rewiring an edge is significant and depicted in Figure 6.4. The graph corresponds to 1,000 vertices arranged in a circle each being connected to its ten nearest neighbours (as in the original paper). Measurements of C and L are averaged over ten random realisations of the rewiring process. As we increase p, the average distance drops relatively suddenly while the clustering coefficient remains high for an appreciable range of p values. There exists, therefore, a range of p values for which the resulting graphs are highly clustered like regular lattices, yet have a small average distance similar to that of random graphs.

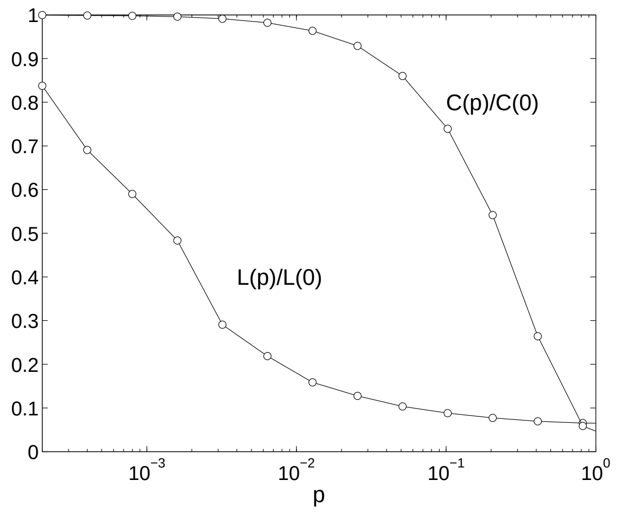


Figure 6.4: Change of clustering coefficient C and average distance L as a regular graph is randomly rewired according to parameter p. C and L are normalised by the maximum values they take ($p = 0$).

Note from Figure 6.4 that the range of p values for which the clustering coefficient remains high is actually very small. It appears inflated only as a result of the logarithmic scaling. The combination of a high clustering coefficient and a small average distance emerges therefore as characteristic topological properties of graphs in a narrow regime between the two extremes of complete order and complete randomness. These properties are now commonly referred to as small-world properties. They have subsequently been found in many biological, social and physical networks (Bornholdt and Schuster, 2003).

For a random graph $G(n, P(\text{edge}))$ we can obtain estimates of these properties. The expected clustering coefficient is given by $C_{\text{ran}} = z/n = p$ where z is the average vertex degree. The average distance in a random graph can be approximated by $L_{\text{ran}} = \log(n)/\log(z)$ (Bollobás, 1985). Table 6.2 shows the corresponding figures for three different collections and the expected values for random graphs having the same number of vertices and the same average vertex degree. We observe that the average distance between vertices in the networks is remarkably low and close to that found in the corresponding random graphs. Moreover, the values do not vary much with the size of the collection. Even for collections comprising more than 30,000 images the average number of edges that need to be traversed to reach a given vertex from any other vertex is just above three. The low average distance between vertices is mathematically expressed by the logarithmic increase of the average diameter of the network with the total number of vertices (Song et al., 2005). This scaling property is very desirable. Even for collections comprising several million images, the small-world topology ensures that any one vertex is never more than a few edges away. This is just what our second hypothesis claims. The analysis cannot reveal, of course, whether the shortest path between two vertices is also a plausible path to users in the sense that they would follow it to solve a search task. But an average distance that comes very close to that of random graphs is as good a lower bound on the length of such plausible paths as we could wish for.

Collection (size)	C_{nnk}	C_{ran}	L_{nnk}	L_{ran}
Corel (7,000)	0.047	0.0040	3.22	2.73
Video (31,284)	0.140	0.0012	3.33	2.83
Getty (8,202)	0.049	0.0060	3.15	2.29

Table 6.2: Small-world properties in NN^k Networks.

A more detailed view on the distribution of distances is given in Figure 6.5. The left histogram shows the distribution of distances between any two vertices in the Corel 5,000 collection. The distribution is roughly Gaussian with a large proportion of images being separated by three edges. Note, as an aside, that the number of image pairs with distance one is just the number of edges in the graph.

By definition, the distance between a focal image and its NN^k is one. Since the NN^k relationship is not reflexive, the length of the minimum path from the NN^k back to its focal image may be larger. The exact distribution for Corel 5,000 collection is decidedly non-Gaussian and shown on the right in Figure 6.5. We find in particular that in about one third of the cases, the NN^k relationship is actually reciprocated, that is, if q is the NN^k of p, p is likely to be the NN^k of q. This observation will assume particular significance in Chapter 7 where it will help us to approximate the set of images that are likely to link to a new image. Very similar histograms are obtained for other collections.

Returning to Table 6.2, let us compare these figures with those found in other small-world networks. The network representing interactions of 488 proteins in *Drosophila* has an average distance of 3.3 (Stanyon et al., 2004). The distance (or degree of separation) between any two people on the planet

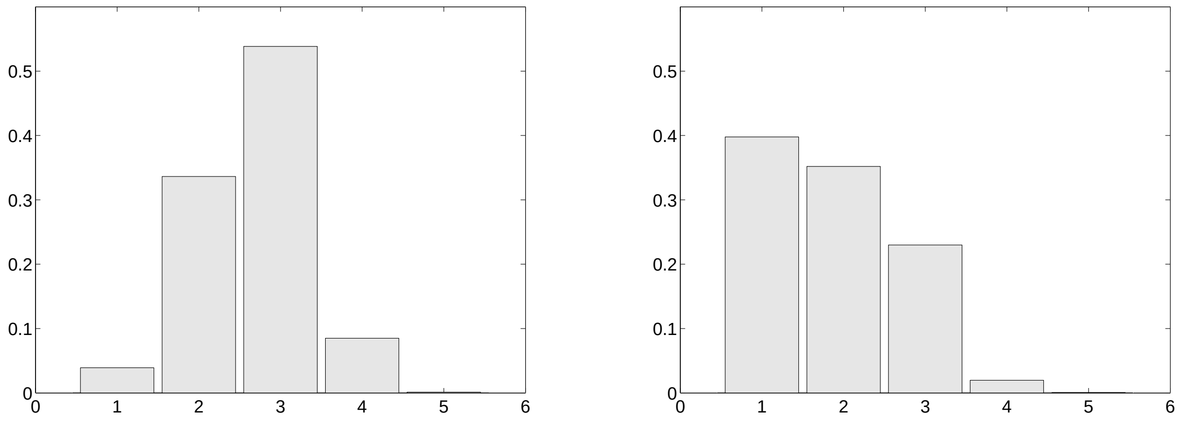


Figure 6.5: Distribution of distances between all images (left), and between NN^k and their focal images (right).

has been estimated to be around 6. Note that unlike NN^k Networks the last two are undirected networks. One of the networks with a rather large average distance is the Web, a directed network of sites and hyperlinks, with an estimated degree of separation between sites of 18.59 in a collection of 10×10^8 sites (Albert et al., 1999).

Figure 6.6 shows how the small-world properties change with the order of the graph in the case of ten features. The figures are obtained by computing the average distance and the average clustering coefficient for a number of subsets of the Corel collection. The left plot shows the logarithmic scaling of the distance with the order of the graph, the right plot shows how the clustering coefficient decreases as the collection grows in size (with the standard deviation indicated by dotted lines). This is a result of the rapid increase in the number of potential edges between a vertex v's neighbours, $[\Gamma(v)(\Gamma(v)-1)]/2$, as the vertex degree, $\Gamma(v)$, increases.

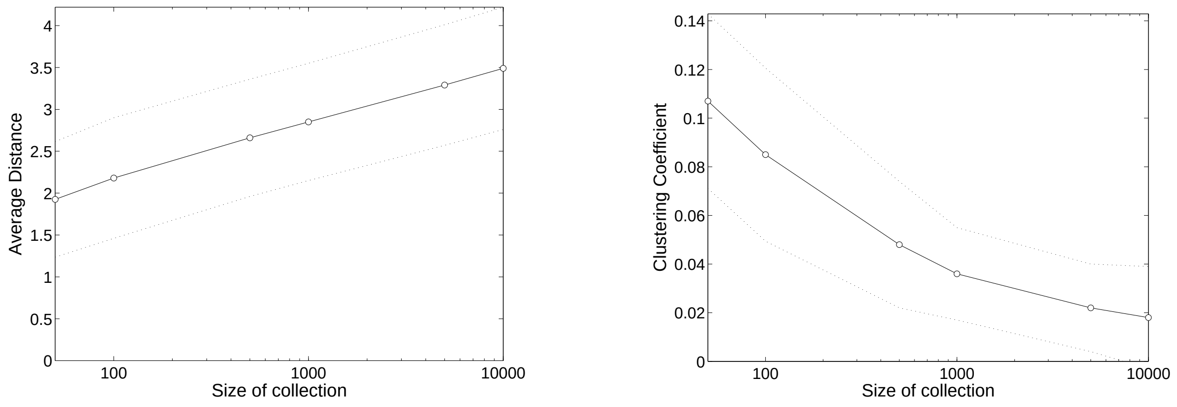
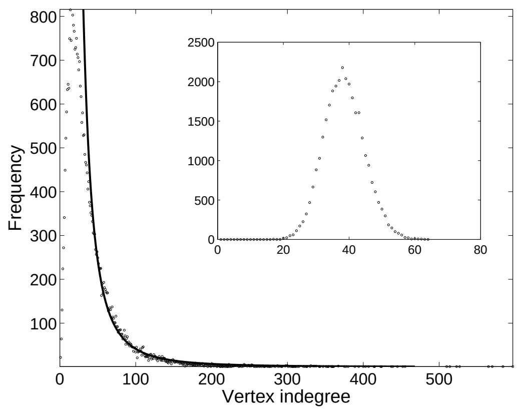


Figure 6.6: Effect of collection size on average distance (left) and clustering coefficient (right)

6.3.3 Degree distribution

Another property that has received considerable attention is the distribution of the vertex degrees (or simply the degree distribution). It has been found that in a number of real-world networks the degree distribution, $P(k)$, can be represented by a power-law $P(k) \propto k^{-\gamma}$ with a degree exponent γ that is usually in the range $2 < \gamma < 3$ (Albert et al., 1999; Albert and Barabási, 2002; Dorogovtsev and

Mendes, 2003; Newman, 2003; Song et al., 2005). The power-law implies that the network contains few vertices with a large number of links (commonly referred to as hubs) and a large number of vertices with a small number of links. Such a power-law arises quite naturally if the networks are allowed to grow such that a vertex is more likely to acquire new links the more links it already has (Barabási et al., 2003). In NN^k Networks no such preferential attachment takes place and we may expect to find a degree distribution that is not scale-free and characterised by the presence of a distinct mode. Since NN^k Networks are directed, we need to distinguish between indegrees and outdegrees. The outdegree of an image is just the number of its NN^k, the indegree is the number of images for which that image is itself an NN^k. Figure 6.7 shows the in- and outdegree distributions for a number of different networks. The difference between the two distributions is quite remarkable. The outdegrees follow a normal distribution with a comparatively narrow range (1-76) and a mode at around 40. Most of the probability mass is concentrated between 20 and 60. The distribution of the indegrees is distinctly non-normal with a long tail extending to the right and without a distinct mode. Some of



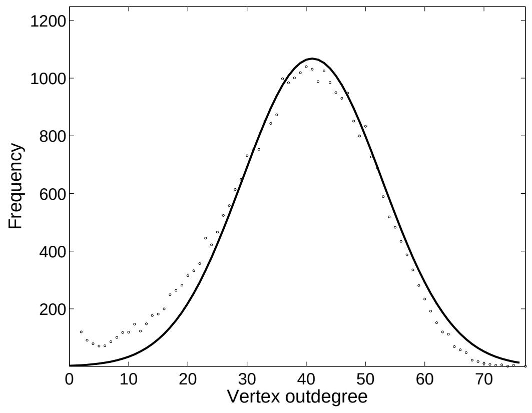


Figure 6.7: In and outdegree distribution for the Corel 32,000 collection.

the images have almost 600 incoming links, a 50th of the number of vertices in the collection and 150 times greater than the average number of outlinks. Both an exponential distribution, $P(k) = \lambda e^{-\lambda k}$, and a power-law distribution, $P(k) = ak^{-\gamma}$, provide reasonable approximations of the data although an exponential distribution does not account particularly well for the probability mass in the right tail. The fitted power-law distribution is shown in Figure 6.7 with $\gamma = 2.55$.

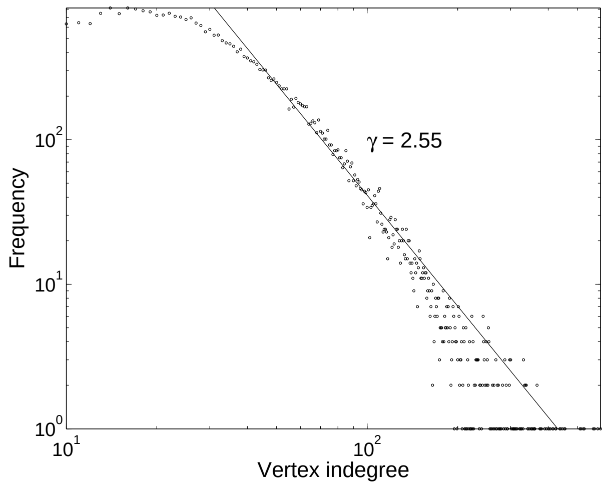


Figure 6.8: Log-log plot of indegree distribution.

Such distributions have the convenient property of appearing as a line with slope $-\gamma$ when plotted on a log-log scale as shown in Figure 6.8. The value of 2.55 for γ is remarkably close to the values found in other networks: 2.6 for the Web, 2.2 and 2.1 for the protein interaction networks of *E. coli* and *H. Sapiens*, respectively (Song et al., 2005). This difference is noteworthy for it is in marked contrast to what would be expected from a random graph with e edges and v vertices where an edge links to a particular vertex with a probability $1/(v-1)$ independently of other edges. The resulting distribution is binomial with mean e/v in the limit of large e and large v. The distribution for an equivalent random graph is shown in the inset of Figure 6.7.

It is instructive to take a look at the images with high indegree and high outdegree, that is, at the hubs and authorities of the NN^k Network. Figure 6.9 shows from top to bottom the five vertices with the highest outdegree, lowest outdegree, highest indegree, and lowest indegree. Some of the groups have quite distinct visual characteristics which may in parts account for the their connectivity properties. Vertices that are most highly linked to (third row) are remarkably homogenous in terms of texture and colour with very little depth while vertices that are only poorly linked to (fourth row) tend to display strong colour and brightness contrast. Based on this one example, it is tempting to conclude that low indegree images have strong visual characteristics while high indegree images have intermediate feature characteristics.

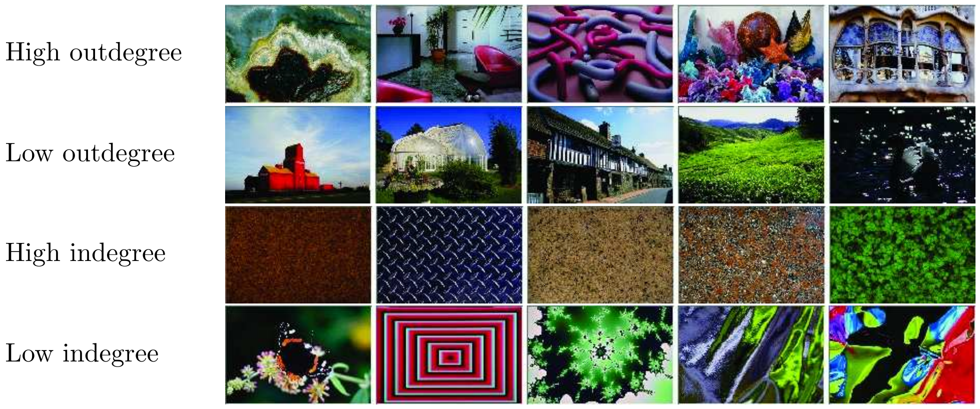


Figure 6.9: Images from Corel 32,000 with extreme in- and outdegrees.

A synthetic data set confirms this conclusion (Figure 6.10, left). The data set consists of 5,000 two-dimensional data points drawn randomly from a multi-variate normal distribution. Each axis can be thought of as corresponding to a different image feature. For each data point we determine its set of NN^k and record the number of in and outlinks. The left plot shows the high and low indegree points. A clear pattern can be observed: low-indegree points are fairly uniformly distributed across the data cloud while high-indegree points tend to concentrate in the centre of the cloud.

The latter observation should not come as a surprise for data points in the centre can be linked to from all directions in the feature space while peripheral points can only be linked to from some directions because of the lack of data in others. In fact virtually all of the points of the convex hull of the data set turn out to be low-indegree points. The uniform distribution of low-indegree points in feature space implies a greater likelihood for such points to lie in the periphery as the underlying data distribution is normal. That we observe such points not only in the periphery is likely to be a result of the fact that as we move away from regions of high density, inter-point distances increase. As a consequence, a point is more likely to have a nearest neighbour in the direction of the centre and is less likely to link to points further out in the periphery. These observations go some way towards explaining the non-normality of the indegree distribution.

A plot of high and low outdegree points is, by comparison, rather uninformative (Figure 6.10, right). Their distribution seems normal with points of both groups extending far into the periphery of the data set. And indeed, the first two rows in Figure 6.9 are visually remarkably similar. The number of outdegrees of an image seems to be governed chiefly by the local distribution of nearby images.

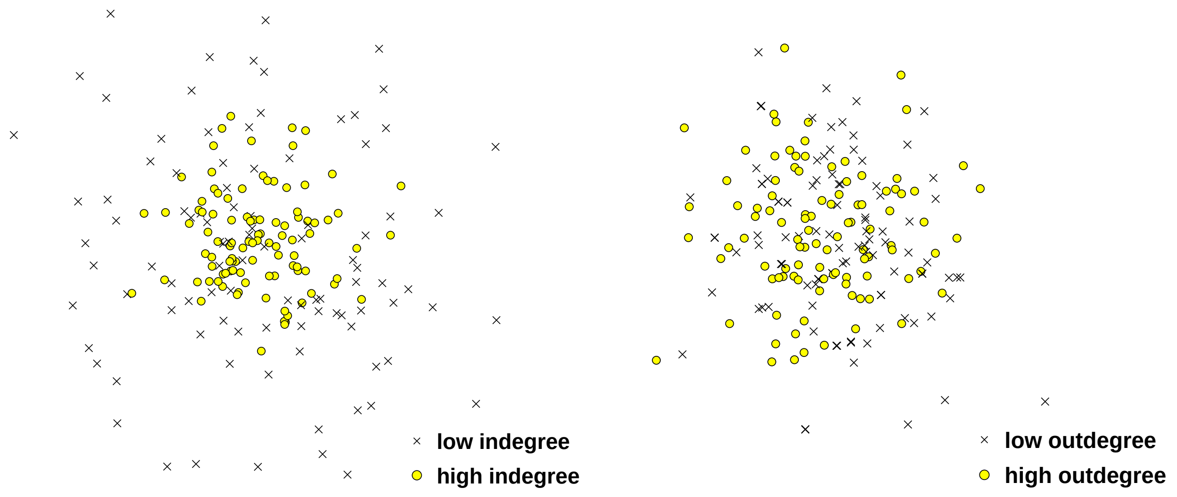


Figure 6.10: Points with extreme in- and outdegrees for synthetic data.

6.4 Semantic analysis

In the preceding sections, we have enquired into the topological structure of NN^k Networks without any reference to the meaning or relevance of images. We called this the formal analysis. It has brought to light a few interesting properties, namely small average distance between any two images, high clustering coefficient and the presence of a giant component even for a small number of features. What has been missing up to now is what may be called a semantic analysis of the browsing structure as has been carried out in Heesch and Rüger (2005). By this we mean an analysis of the structure in terms of the distribution of particular subsets of images. This is clearly essential to test the last of our three hypotheses. The subsets we are interested in are the relevance classes that associate with a particular information need. Our hope is that two images that belong to the same relevance class are likely to be close to each other. We shall denote by V_c the set of all those vertices in the network G that belong to the same relevance class c. $G[V_c]$ is the subgraph induced by V_c. We call this the relevance subgraph. We can view semantic analysis as the topological analysis of relevance subgraphs. Our hope is that these subgraphs are in some sense *compact*.

6.4.1 Compactness of subgraphs

In our endeavour to complement our formal analysis with a semantic analysis, the first problem we face is the question of how to measure compactness of subgraphs. Intuitively, a subgraph that is complete should be classified as highly compact, and as least compact a graph where no two vertices are connected. We will in the following motivate and describe three properties which we think capture this intuitive notion well. For each of the properties, we compare the observed values with those obtained on random subgraphs obtained by forming the graph induced by a random set of vertices of the network. Such comparison is necessary since the absolute compactness values depend not only on the structure of the relevance subgraph but crucially on the structure of the whole network.

6.4.2 Average distance

Perhaps the most intuitive property we would like relevance subgraphs to possess is that its constituent vertices have a small average distance (defined as the length of the shortest path between two vertices) in the original graph. It is important to note that we are not interested in the distance within the subgraph but within the original graph since the user is not confined to the subgraphs when navigating through the network. The lowest possible average distance that can be attained is one and requires the subgraph to be a complete graph where each vertex is connected to every other. As we have shown as part of our formal analysis, the average distance between any two vertices in NN^k Networks lies between three and four across a range of different collections. Given the lower bound of one we therefore do not expect any further reduction to be very substantial.

The average distance between any two vertices of the relevance subgraph turns out to be very similar to that for a random subgraph. The results are best displayed in the form of a boxplot (Figure 6.11). The upper and lower boundary of the box mark the 25th and 75th percentile of the data. The notches in the box are graphic confidence intervals about the medians. The average distance in the relevance subgraphs hovers around 2.8 for both collections, compared to 3 and 3.6 for random subgraphs. Although the magnitude of the difference is small, it is significant under a paired t-test ($p < 0.01$). The result tells us that the other members of one's class are on average only 2.8 vertices away. If we display during the browsing process the focal image and its nearest neighbours, as in Heesch and Rüger (2004), this means that after only two clicks the relevant image comes within sight.

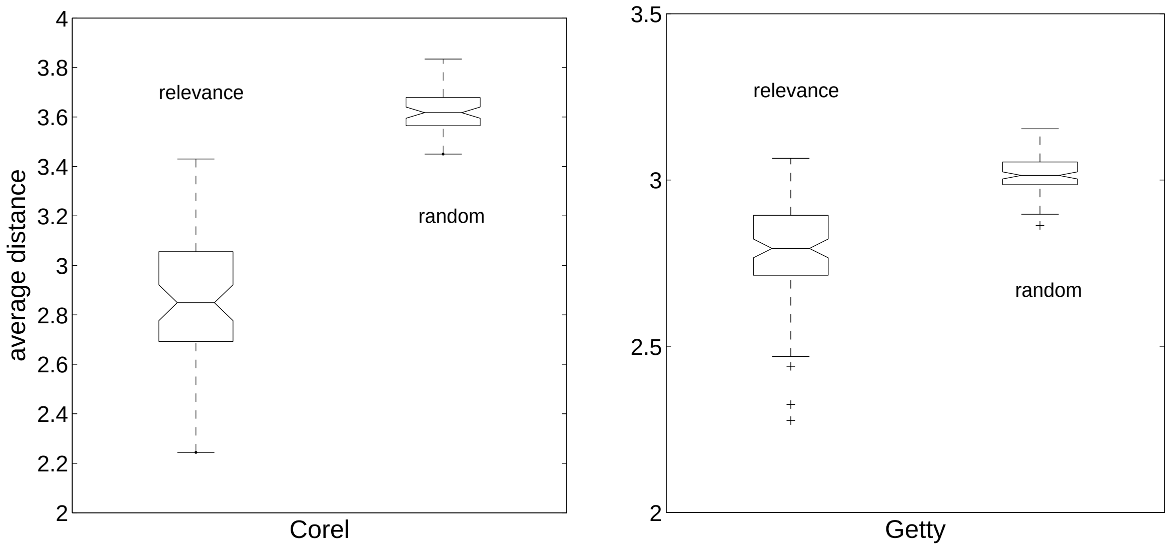


Figure 6.11: Average distance between vertices in subgraphs.

6.4.3 Average vertex degree

The average vertex degree of a vertex of G_c tells us what proportion of images within the same relevance class are directly adjacent. If vertices of one relevance class tend to be closer together, a larger proportion of these will be directly adjacent. If we let $G_c = (V_c, E_c)$ be the subgraph induced by V_c, then the average vertex degree of G_c should be larger than for a random subgraph. To take extremes, the average vertex degree of a complete subgraph of order $|G_c|$ is $|G_c| - 1$, since each vertex is connected to every other. For a graph induced by a randomly chosen vertex set of G the average vertex degree is to a good approximation equal to $|G_c| \times \bar{d}/|G|$ where $\bar{d}$ is the average vertex degree of G. This is typically much smaller than $|G_c| - 1$.

We have just seen that vertices of a relevance subgraph tend to be closer than vertices in a random subgraph. It seems likely, therefore, that in a relevance subgraph a larger proportion of vertices are directly adjacent, and hence that the average vertex degree is higher. This is indeed the case. Figure 6.12 plots the average vertex degree against the size of the relevance class (i.e. the order of the subgraph). Because there are typically many classes having the same number of images (there are, for example, 42 classes of Corel of size 100), we only plot their average such that for each class size there is at most one point. Note that the vertex degree increases with the order of the subgraph, both for the relevance subgraphs and, less conspicuously, for the random subgraphs. Indeed, if we were to increase the class size further, the average degree would converge towards the average vertex degree of G. More importantly, note that the average degree of the relevance subgraphs is considerably larger than that for random subgraphs. For a random subgraph every vertex is connected on average to less than one other vertex: most vertices have no neighbours. In relevance subgraphs, each vertex is linked to at least one other vertex (1.4 and 5.6 for Getty and Corel, respectively), suggesting that the structures do form connected wholes.

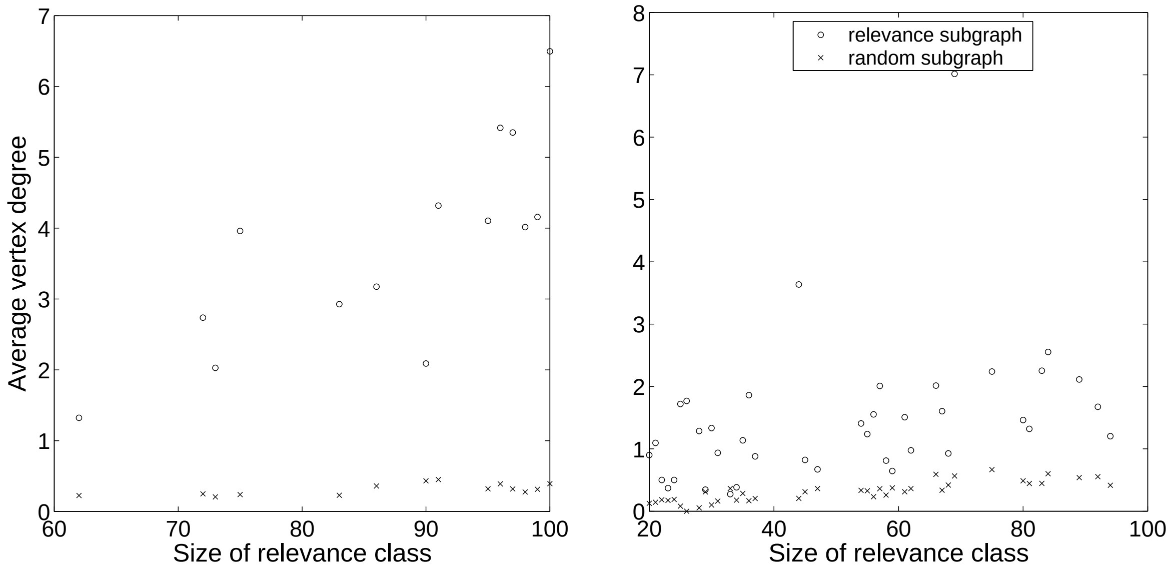


Figure 6.12: Average vertex degree for Corel (left) and Getty (right).

6.4.4 Connectivity

The last property which we shall consider is the order of the largest strongly connected component of the subgraphs. We hope that a relevance subgraph has the tendency to be strongly connected, or at least, that its largest strongly connected component comprises a large proportion of the vertex set. This property may seem at first to be the least obvious. Upon reflection, however, it turns out to be perhaps the most desirable. If relevance subgraphs had the tendency to be strongly connected, users who found a relevant vertex would be able to navigate to all other relevant images by following what might be called relevance paths, sequences of vertices each of which is relevant to their information needs. Connectivity is more important than average distance, for even if relevant vertices were only separated by a few clicks, the path connecting them would potentially lead through non-relevant territory.

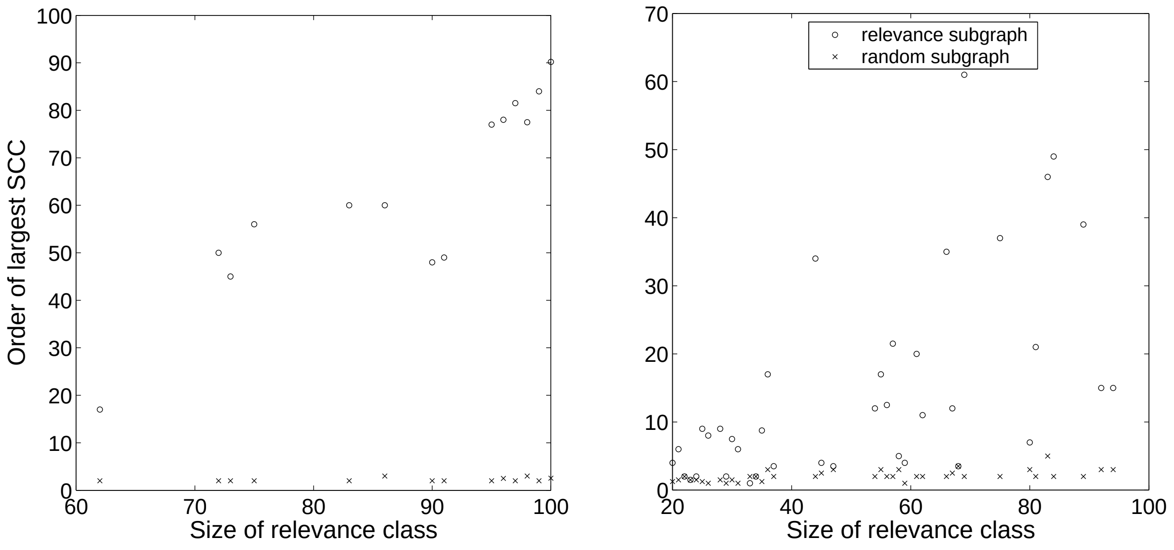


Figure 6.13: Order of largest strongly connected component for Corel (left) and Getty (right).

The results so far give us some indication of whether the subgraphs tend to be strongly connected. The high average vertex degree of 5 for the Corel relevance subgraphs is particularly suggestive. Figure 6.13 plots the order of the largest strongly connected component against the size of the relevance class. As before, we average over all classes of the same size. Unlike random subgraphs, relevance subgraphs have remarkably large strongly connected components. The relative proportion of images of a class contained in its largest component lies around 85% for Corel and 26% for Getty.

Even if a network is not strongly connected, some images that do not belong to the largest component can nevertheless be reached from it. Thus, the actual number of images that are accessible from any one image of the component typically exceeds the order of the component. Figure 6.14 illustrates this. The subgraph corresponds to the 'Flowers' class of Getty. Images that belong to the same strongly connected component are linked by black lines. Images without such lines form a component on their own. Gray lines are connections between components. The largest component is on the left with 14 of the 28 images. Of the remaining images, two can be reached from it via the gray lines.

Note that these three properties are only partly correlated. We can for example think of subgraphs with average distance of 2 with no path between any two vertices, or a subgraph with only one strongly connected component but with very low average vertex degree and large distance. A summary of the results is found in Table 6.3. The differences are significant in all cases with $p < 0.001$.

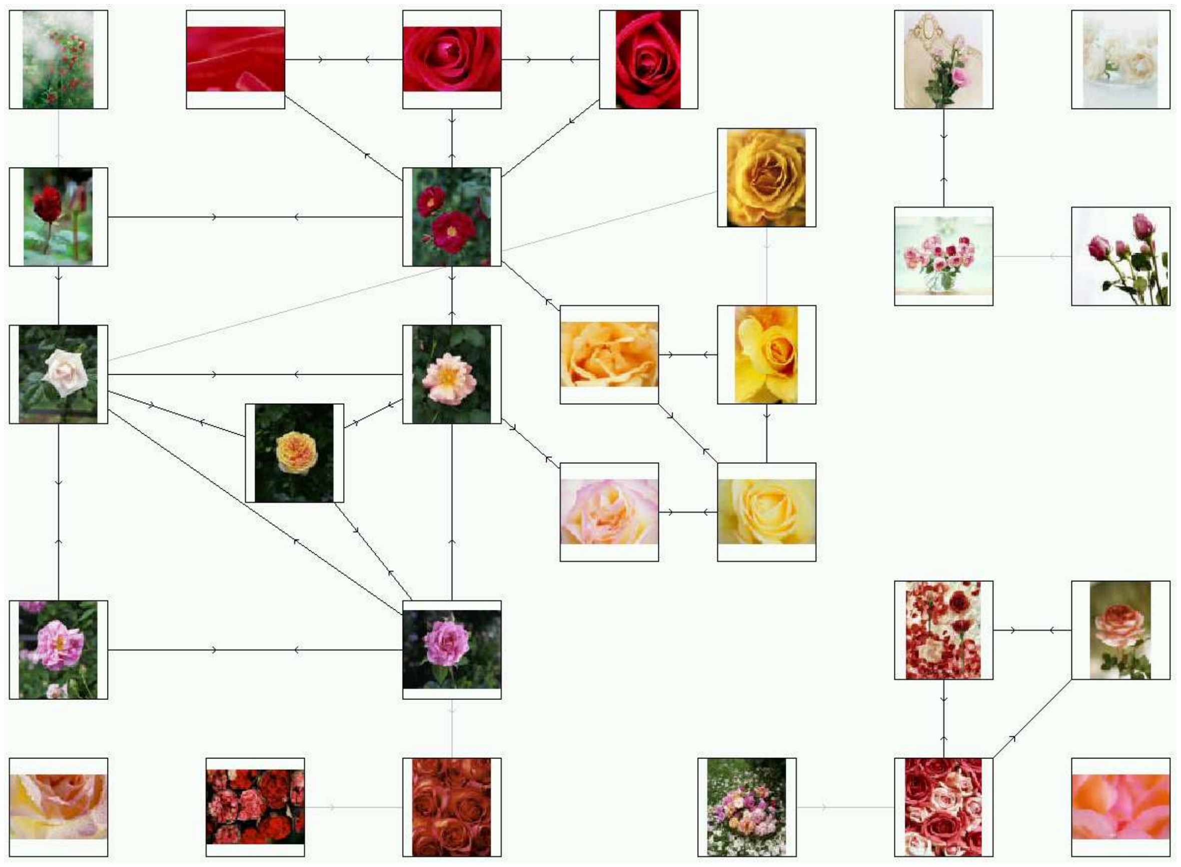


Figure 6.14: The subgraph corresponding to the ‘Flowers’ class of the Getty collection.

	Avg Dist	Avg Out	Max SCC
Getty 8,202	2.79 (3.02)	1.44 (0.27)	0.26 (0.05)
Corel 6,192	2.85 (3.62)	5.64 (0.39)	0.84 (0.025)

Table 6.3: Compactness figures on relevance subgraph and random subgraphs (brackets).

Our semantic analysis suggests that NN^k Networks do capture a considerable degree of the semantic structure of a collection. This might be expected if visual similarities are at all informative about semantic relationships. Note, however, that NN^k Networks connect images to a wide variety of neighbours that are only ever the top-ranked images under some feature combination. Semantic structure is preserved in spite of this diversity. Interestingly, the diversity of NN^k leaves its mark on the particular way semantic structure is represented: as suggested by the large average distance of images within relevance subgraphs, images of the same relevance class tend to be strung out along relevance trails and do not form tight, sphere-like clusters as one would expect if images were organised according to one particular metric. Thus, in accord with our third hypothesis, relevant images hang together and they do so in a peculiar way that reflects the construction principle of NN^k Networks.

6.5 Conclusions

We began this chapter by inspecting more closely the structural properties of NN^k networks. The analysis was motivated by the intuition that navigability of these networks depends crucially on their connectivity properties. For a number of different collections varying in size between 50 and 32,000, we found that vertices are on average separated by only 2-3 edges and that the average distance scales logarithmically with the size of the collection. In addition, vertices that share a third vertex are more likely to be themselves connected than would be expected for random graphs. The joint occurrence of these two properties, small average distance and high clustering, suggests that NN^k networks may be classified as small-world networks and thus are similar in structure to social networks, protein interaction networks and the Web. Unlike the latter, however, the NN^k networks we considered invariably exhibit the tendency to form giant components that typically encompass more than 90% of the vertices.

Analysing the networks from a semantic viewpoint reveals that the structural idiosyncrasies extend beyond those unravelled in our formal analysis. The distribution of images belonging to the same semantic class is highly non-random and remarkably compact as judged by three compactness measures that we propose: average distance, average vertex degree and degree of connectivity. Although the distance in the network between any two images of the same class is on average not much smaller than between any two randomly chosen images, the structure of the graph differs nonetheless substantially from random subgraphs. In particular, images of the same class tend to establish large strongly connected components that in the case of Corel contain an average of 85% of all class members. Hence, even though an image does not tend to be directly adjacent to a vast number of others of its class, most of these can be reached by following paths within the strongly connected component. The distribution of relevant images is thus best pictured not as a dense region with small diameter but as a reticular substructure that extends deep into the network whilst remaining largely connected.

It is natural to ask to what extent the structure which we have begun to lay bare in this chapter can be captured and leveraged to facilitate image search. Before turning our attention to this question in Chapter 8, we will first have to address fundamental practical issues pertaining to the basic browsing framework as described in this and the preceding chapter.

Chapter 7

NN^k Networks in Practice

7.1 Introduction

Chapter 6 has provided a formal and a semantic analysis of NN^k Networks. The formal analysis led to the discovery of small-world properties: images are separated by only a few links and they tend to cluster. The semantic analysis has revealed that the local structure captures the semantic relationships between images: relevant images tend to occur in compact subgraphs so that users who find one relevant image are likely to find more by following relevance trails through the network. Chapter 6 has also helped to identify a number of issues of practical significance that need to be resolved to add value to NN^k Networks and to which we shall turn our attention in this chapter.

We have seen that in spite of their great sparsity the degree of connectedness of NN^k Networks is remarkably high with all networks examined having a giant component that comprises more than 90% of the vertices. For the purpose of interactive search, the presence of a large giant component is a very encouraging structural property. Nevertheless, there exist a fair proportion of images not part of the giant component. Many of these vertices correspond to groups of near-identical images that are strongly interlinked but form no links with the giant component. Other images are not connected to at all and form unreachable islands. Both types of fragmentation have an adverse effect on the navigability of the networks. Clearly, we want every vertex to be reachable from any other which requires NN^k Networks to be strongly connected. In Section 7.2 we suggest modifications of the initial structure to ensure that this is indeed the case.

In Section 7.3 we consider the problem of updating NN^k Networks in dynamic environments where images are continuously added to or removed from a collection. We develop an approximative method that circumvents the need for recomputing the entire network structure and analyse the effect this approximation has on the topology of the resulting network.

Users search for images by choosing paths through the NN^k Network. Every such path has to start somewhere and for large collections the starting point may be an important determinant of success. The chance of finding what one is looking for should increase if we were to begin the browsing path in a region that is topologically not too distant from our target. One way to achieve this is by showing the results of an automated search given some initial query. This is the method of choice in a number of other browsing frameworks for both images and documents. The most notable example, of course, is the Web, where keyword searches place the user in areas of interest which can then be explored further through browsing. In image retrieval, queries are more difficult to formulate and to process, and it is indeed one of the strengths of browsing frameworks that queries need only exist in the user's mind. In Section 7.4 we try to leverage this advantage and propose an alternative query-free way of providing initial access to NN^k Networks.

When retrieval techniques rely heavily on user interaction, their automatic evaluation becomes a daunting task. That experiments with real users seem indispensable when gauging the value of NN^k Networks for image search results from the difficulty of formalising and thus automating the way users navigate through the network. The annually held video retrieval conference (TRECVID) provides an ideal opportunity for such experiments. Our participation in the search task of TRECVID will be described in Section 7.5.

7.2 Connecting SCC

Fragmentation of NN^k Networks into multiple strongly connected components has two causes. It may result from the presence of images that deviate so radically from all other images in their visual attributes that they are not the NN^k of any other image. These images have no inlinks, but like every image in NN^k Networks they have at least one outlink. Each such image forms a strongly connected component on its own.

Another cause of fragmentation is the presence of groups of very similar images. Each image in such a group then monopolises all outlinks of the other group members thus creating strongly connected components of two or more images. We shall refer to the first type of component as a source and to the second type as a sink. The terms are borrowed from the theory of network flow where they denote privileged vertices that represent the origin or destination of a flow of some commodity.

This fragmentation of NN^k Networks into a giant component, sources and sinks reveals potentially interesting structure about the image collection. For a collection of news video keyframes, for example, the number of sinks is astonishingly high. It turns out that the great majority of these sinks consist of near-identical keyframes from commercials. The fragmentation could thus be exploited to clear the network from unwanted images. If sinks and sources were retained in the network, however, they would limit the navigability of the networks. From the definition of a strongly connected component follows that if we were to leave one, we would be unable to return to it. Equivalently, if it were possible to return to it from a vertex outside of the strongly connected component, it would be impossible to reach that vertex in the first place. For example, users may reach strongly connected components consisting of near-identical images, i.e. sinks, but will be unable to leave them again as all links are within the strongly connected component itself. Similarly, visual outliers forming sources may link to the rest of the network but cannot be reached from anywhere else. They can only be visited if they are the source of a browsing path.

Sources are vertices with zero inlinks and correspond, therefore, to the anti-authorities which we met in Section 6.3.3 of Chapter 6. Indeed, each of the five images in the last row of Figure 6.9 forms a source in the corresponding NN^k Network. Their unusual visual characteristics provide immediate confirmation of our conjecture that sources are outliers with respect to their low-level feature representations. Figure 7.1 shows two examples of sinks found in the Corel collection. Each of the images points to at least one other from the same group and none of them links to an image outside the group. It is interesting to note that these sinks are nevertheless linked to quite extensively. The first sink receives a total of 20 links, the second sink 119.

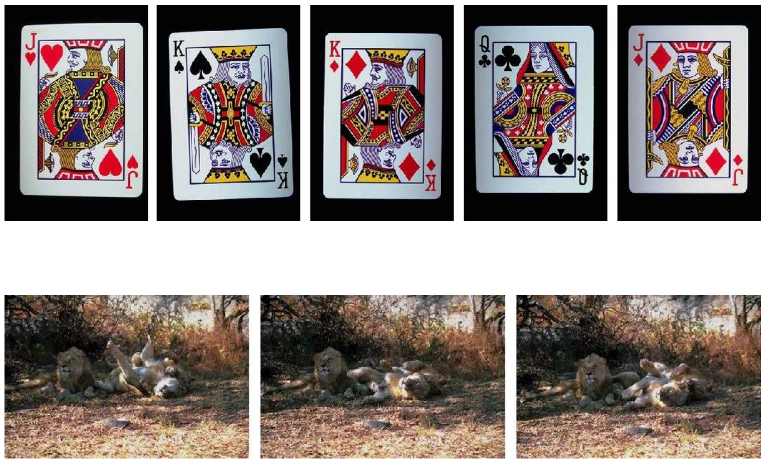


Figure 7.1: Two examples of sinks in NN^k Networks.

The degree of fragmentation depends on the collection as well as on the type and the number of features used. Figure 7.2 gives an indication of how the degree of fragmentation depends on the number of features for a collection of size 20,000. In the extreme case with only one feature being used to construct the network, each image links only to the image that is closest under that one feature.

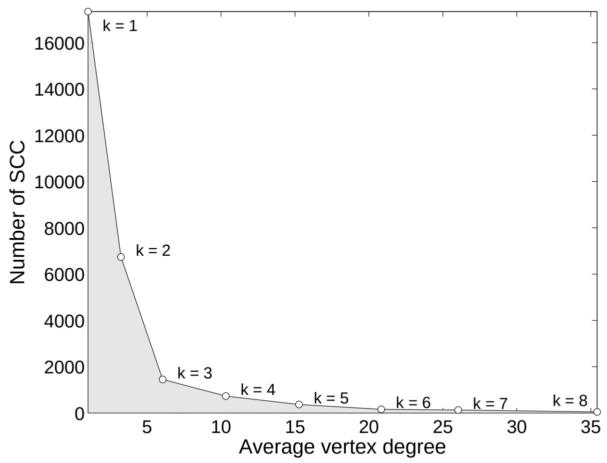


Figure 7.2: Fragmentation can be reduced by increasing the number of features.

The network, therefore, has as many edges as vertices and is as a result highly fragmented. In fact, each image forms a strongly connected component on its own unless it is linked to by its one and only NN^k, either directly or through a sequence of intermediate vertices forming a cycle. The number of strongly connected components for $k = 1$ is around 17,000. If the largest sink were of order two, we would have exactly $2 \times 17,000 - 20,000 = 14,000$ sinks of order one. Because sinks may also be of order greater than two, there are *at least* $14,000$ images forming strongly connected component by themselves. As we increase the number of features, more and more of these get inter-linked before finally being swallowed up by the giant component. The situation for sinks is different. Because images belonging to sinks are near-identical, they will remain nearest neighbours of one another irrespective of the underlying feature. As we increase the number of features, near-identical images will tend to remain inseparably tied together. As a result, the relative contribution of sinks to the overall degree of fragmentation increases with the number of features.

In Table 7.1 we have collected for a number of collections the proportion of images not in the giant component that form part of sinks and sources, respectively. For eight features (i.e. $k = 8$), the proportion of images forming sources lies in the range $0 - 0.1\%$, while that of sinks varies between $0.1\% - 0.32\%$. Even for the large and very diverse Web collection, which was compiled by downloading 1.14 million images from selected Canadian, American and UK universities, the proportion of images forming sources is remarkably small despite employing only five features. The contribution of sinks, meanwhile, is rather high. An interesting comparison is that between Getty (8,202) and Corel (10,000) as these are of similar size but differ in the number of features. The eleven features used for the Getty collection account for the better connectedness and the much lower incidence of sources (0.012 versus 0.06), but the proportion of sinks remains at the same level as for the Corel collection, providing some evidence for the claim we made at the end of the preceding paragraph.

Collection	Size	Number of Features (k)	Avg Degree	Sinks (%)	Sources (%)
Corel	100	8	13.1	0.00	0.000
Corel	500	8	19.5	0.00	0.000
Corel	1,000	8	22.3	0.00	0.100
Corel	5,000	8	28.9	0.20	0.100
Corel	10,000	8	31.8	0.15	0.060
Corel	20,000	8	35.4	0.26	0.095
Corel	31,997	8	37.4	0.32	0.075
Getty	8,202	11	51.4	0.12	0.012
Web	100,000	5	27.9	2.73	0.102

Table 7.1: Percentage of images in sinks and sources.

While the problem of visual outliers can be alleviated by choosing a diverse and sufficiently large set of features, the problem of sinks requires a different approach. We shall in the next section describe a principled method for removing both sinks and sources.

7.2.1 Sink elimination through constrained NN^k computation

Sinks are arguably the greater nuisance for while sources only mark out unreachable territory which simply remains invisible to the user, sinks form traps from where there is no escape. There are two approaches we can take to the sink problem. We may either remove the vertices and all associated arcs from the network, a procedure that is guaranteed to decrease the number of sinks by one and to add no additional source. Alternatively, we may add suitable arcs to the sink vertices with the effect of weaving the sink back into the giant component. The second approach is no doubt preferable and leaves only the question which arcs to add. Although a single arc linking any one sink vertex back to the giant component would be sufficient to establish the desired connectedness, for larger sinks users would have to look hard to find that one arc. We instead consider adding arcs to each of the sink vertices. A systematic method that is in closest agreement with the original construction principle of NN^k Networks involves recomputing for each sink vertex the set of its NN^k with the added constraint that for each weight combination we determine the most similar image from the complement of the set of sink vertices. Any such NN^k will provide a possible exit route. Since we do not want to destroy the inter-connectedness of the sink vertices, we add the new set of NN^k to the original set. Figure 7.3 gives an example how a strongly connected component consisting of three nodes (left) is thus linked up with the remaining network (right).

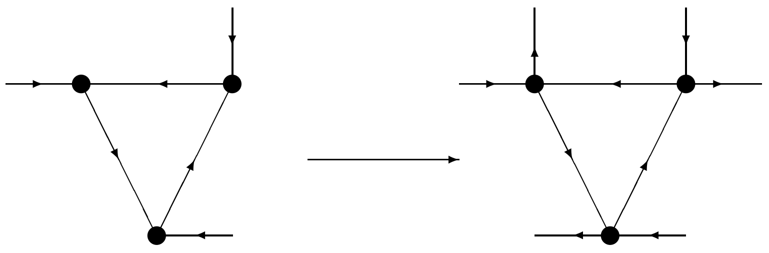


Figure 7.3: Weaving sinks into the giant component.

7.2.2 Source elimination through arc reciprocation

We would like to eliminate sources not by removing the associated vertices but by linking them to the giant component. Sources have no incoming arcs from the giant component. The addition of any one of such arcs would be sufficient to incorporate the source into the giant component. A reasonable strategy for choosing such arcs is to reciprocate all the outgoing arcs of the source vertex (Figure 7.4), so that a source acquires as many inlinks as it has outlinks. Even if the source is not the closest neighbour of any of its NN^k, we expect it to be much more similar to it than the great majority of other images in the collection and an artificial arc will not be entirely ill-placed. In fact, recall from Chapter 6 that in the NN^k Networks we studied, more than a third of the arcs are actually reciprocated.

If sinks only ever consisted of two vertices, we could detect sink vertices once we have determined their NN^k by observing that they have but one outlink. This would allow NN^k determination to remain a one-pass algorithm. Unfortunately, this assumption is not valid. Although for many collections the largest number of sinks consist of pairs, the majority of images come from larger sinks. Sink elimination, therefore, requires a second pass through the image collection once all the NN^k, and thus all the strongly connected components, have been determined.

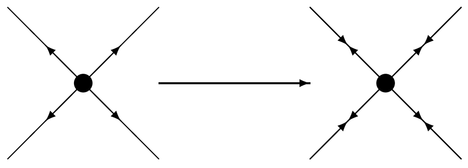


Figure 7.4: Eliminating sources by reciprocating their outgoing arcs.

7.2.3 Further considerations

Sink and source removal can be achieved efficiently with the above methods. One minor drawback of the proposed solutions is that they force us to accept a number of inconsistencies with regard to the additional information which we store for each image's NN^k. Recall from Chapter 5 that we keep the support of each NN^k (i.e. the relative size of its iso-NN^k region) as a measure of the similarity between it and its NN^k. The new neighbours computed during sink elimination (i.e. those not forming part of the sink) have iso-NN^k regions that overlap with those of the initial set of NN^k (i.e. those forming part of the sink). Both sets partition the weight space and the sum total of their supports is therefore 2 rather than 1 as for ordinary sets of NN^k. A method that can consistently be applied and that treats each set as being of equal importance involves rescaling the support by dividing by 2. Often sinks consist of no more than two images and occur at relatively low frequency in the collections we examined. The inconsistencies introduced are therefore rather slight and an acceptable price to pay for the improved connectedness of the network.

7.3 Incremental update of NN^k Networks

Few image collections of interest are static. Most are in constant transition as images are added to or removed from the collection. Although the construction principle underlying NN^k Networks is rather simple, we identified computational complexity as an important issue and addressed it in Chapter 3 by developing various approximations and optimisations. These methods seem sufficient if the collection is to be indexed only once. If the collection is in constant flux, however, recomputing the entire network every time a new image has been added is a prohibitive task. One solution to cope with dynamic environments consist in recomputing the network only every so often once a certain number of changes have accumulated. In the meantime, any changes that occur could be brought to the attention of the user by other means, for example in the form of a text caption that alerts users to the fact that an image is no longer available, or with a separate network for recently added images. Here we shall investigate in how far we can also achieve a truly incremental update whereby we add and remove images without having to compute the entire structure *de novo*. To understand how this could be done, we need to be clear about how NN^k Networks change upon the addition and removal of images.

7.3.1 Update scenarios

Adding a new image p to an existing NN^k Network can have a number of effects on the topology of the network. It involves, generally speaking, the addition of exactly one vertex and of at least one edge. Part of the integration of the new image consists in connecting it to all its NN^k. These can be determined by applying the grid method developed in Chapter 3 with p as the focal image. Thankfully, adding outlinks to p does not have any knock-on effects on the other edges of the network and can therefore be achieved efficiently. However, p can potentially be among the NN^k of any of the other images already in the collection. Moreover, by becoming an NN^k of some other image q, other outlinks of q may be broken because the new image p may completely displace some of the former's NN^k within their iso-NN^k regions. And even if it does not wholly displace them, it will have the effect of decreasing their relative support which would therefore need to be updated. Figure 7.5 illustrates the changes that may occur to the network due to the addition of a new image. The new vertex on the right acquires outlinks to two other vertices. One of these former vertices links to the new image and in turn disconnects from one of its previous NN^k. Note, as an aside, that although a new vertex will typically cause the number of edges to increase, one can conceive of situations where a new vertex is very similar to another and redirects all outgoing arcs of that vertex to itself leading to a graph of smaller size.

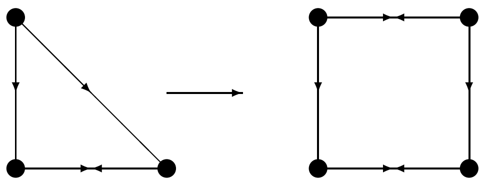


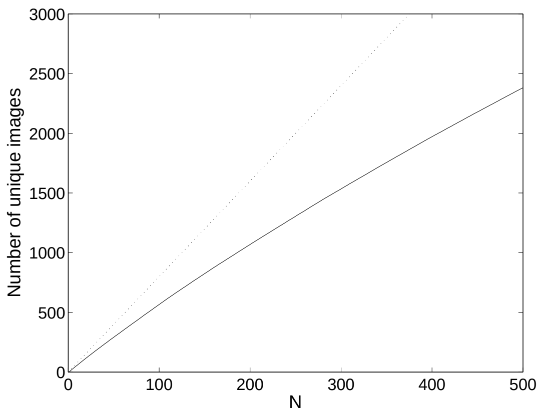
Figure 7.5: Insertion of a new node into an NN^k Network.

Removal of vertices from the NN^k Network is of equal importance as vertex addition for practical purposes but constitutes a much smaller challenge. In addition to removing the vertex and all its outlinks, we need to recompute the NN^k of only those vertices that have previously linked to it. As we have seen in Chapter 6, this number ranges between 0 and a few hundred for collections of several thousand images with the majority of vertices receiving inlinks from only a few dozen. If we store for each vertex both out *and* inlinks, vertex removal can be done efficiently. The principal challenge, therefore, lies in image addition and it boils down to the problem of finding the images that are focal with respect to the new image.

7.3.2 Subset selection

Ideally we only recompute the NN^k for images whose NN^k have changed upon the addition of an image p. Whether this is the case for an image q can only be ascertained, however, by actually computing the NN^k of q. It seems that we would have to carry out the computation first in order to find out whether it was necessary. Given that most images are focal to only a very small proportion of images, in the great majority of cases we would find that the new image is not an NN^k of q and that the computation was indeed wasted. The search, therefore, is for other cues that give some indication as to whether an image is focal with respect to the new image. We wish to compute the NN^k only for some set Q that is substantially smaller than the image collection. To see what such a cue could be, recall from Chapter 6 that although the NN^k relationship is not reflexive, the chance that $q \rightarrow p$ if $p \rightarrow q$ is around 35% for the networks considered (Figure 6.5). In other words, about a third of the focal images of the new image are found among its NN^k. Making Q the set of NN^k of the new image would be a simple method of finding a fair proportion of focal images. Q would typically not exceed 80 for collections of 30,000 images. To do better than this, we could add to Q the images that are not one but two links away from p. According to Figure 6.5 we can expect 60% of the focal images to be among this augmented set but the size of Q also becomes very large very quickly making this particular method unsuitable.

But the observation of partial reflexivity suggests another possible approach. We first note that whether an image is an NN^k of p depends in a generally non-trivial way on its feature-specific distances to p. An image that is more similar to p than any other under at least one feature is guaranteed to be an NN^k. If there is no such feature, it may nevertheless be an NN^k if it is sufficiently close under a set of features. Conversely, an image that is dissimilar with respect to all features is unlikely to be an NN^k. Imagine recording for each feature the N images that are most similar to an image p. This yields a set of kN images not all of which are unique. For some N, the size of the set of unique images U_N varies between N and kN (given that the latter does not exceed the collection size) and is generally larger for more orthogonal features. U_N contains all images that are among the N most similar images for at least one feature. For a collection of 10,000 images the relationship between U_N and N is depicted on the left of Figure 7.6. By definition U_1 is a subset of the NN^k of p. As we increase N we expect U_N to include more and more NN^k. The precise relationship is depicted on the right of Figure 7.6. We note that almost all of the NN^k are among the images that are highly ranked under at least one feature. For example, 95% of all the NN^k are contained in the set U_{50}. From the left graph we observe that the size of U_{50} is around 250. The partial reflexivity of the NN^k relationship now suggests that a similar observation can be made for focal images. That is, most of the focal images are expected to be found among the highly ranked images. This is indeed the case. The corresponding curve is shown on the right of Figure 7.6. The curve does not rise as steeply as that for the NN^k and shows slower convergence but clearly vindicates our hypothesis that focal images also tend to be highly ranked under at least one feature. In particular, 50% of the focal images are contained in the set U_{12} (75 images), 75% in the set U_{40} (252 images), 80% in U_{56} (350 images).



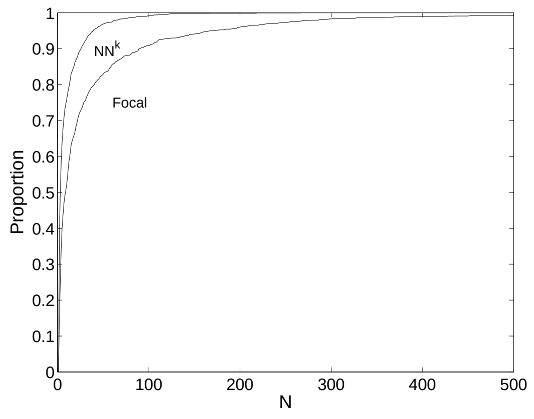


Figure 7.6: Number of unique images among the pool of top N (left). Proportion of NN^k and focal images among the pool of top N (right).

The observation suggests setting Q to U_N for some N whose value will determine the trade-off between accuracy and computational efficiency. To achieve an expected coverage of 80% using this method for a collection of 10,000 images, we would have to compute the NN^k of 350 images. This amounts to 3.5% of the computational effort that would be required were we to achieve an exact computation of all the inlinks. The method has three pleasant properties: firstly, it provides us with a good estimate of the expected coverage for a given value N; secondly, it does not add inlinks where there should be none; thirdly, the coverage for a previously added image p will tend to increase as more images are added to the collection. This is so because subsequent subsets Q will occasionally contain focal images of p that were not contained in p's target set Q. Previously added images will thus acquire more inlinks as the collection continues to grow. It is because of its approximative nature that the method fails to find all inlinks in the first place, and that it will find them eventually.

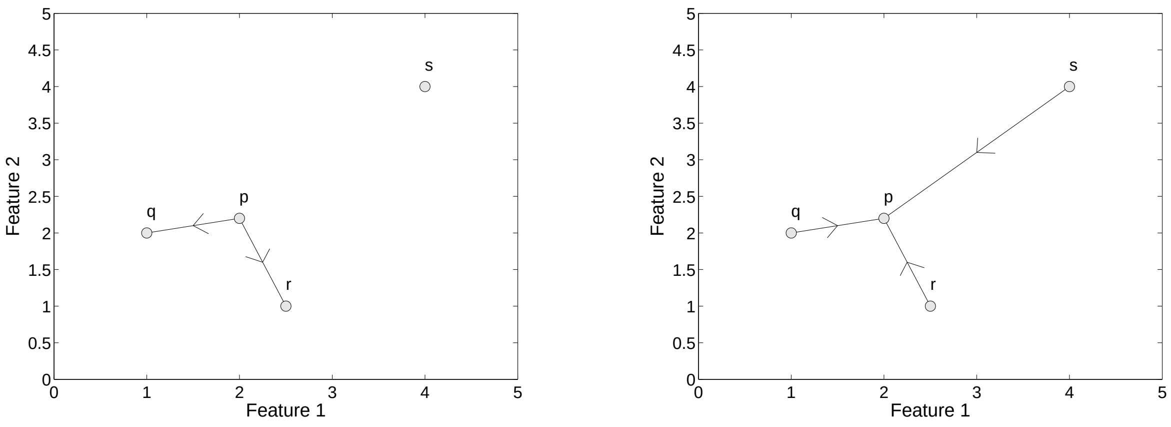


Figure 7.7: Effect of approximative update on network topology.

Much of the preceding section assumes that a single image is added to a collection. Perhaps a more likely scenario involves the addition of a group of images. In such a case we can further reduce the computational cost by noting that the sets U_N may exhibit some overlap since an image from the collection might be focal with respect to more than one of the new images. The set of images for which we want to compute the NN^k is therefore the union of the sets U_N. Because this union is typically larger than the largest U_N, the coverage for each added image will be larger than predicted by the right plot of Figure 7.6.

7.3.3 Effect of approximation on network structure

We can make predictions regarding the topological effect this approximation has on the resulting NN^k Networks. Firstly, since some of the inlinks will be missed, the network becomes sparser than the corresponding exact network. As a result, the average distance between vertices will be somewhat larger. Secondly, we expect the network to acquire more structure. To see why this should be the case note that the focal images we miss are images that are considerably more distant in feature space than many others ($|U_N|$ images to be precise) to the new image p. The fact that despite its large feature-specific distances p is nevertheless an NN^k of some focal image reflects the fact that p's distances to other images are even larger. The idea is illustrated in Figure 7.7. The collection initially consists of three images q, r and s to which a new image p is added. The outlinks of p are shown on the left diagram. Note that image s cannot be an NN^k of p since it is further away for both features than at least one other image. The inlinks of p are shown on the right. Both q and r are focal images but so is s since it does not have any image that is closer to itself under feature 2. Assume now that we only look for focal images in the set U_1, that is, in the set of images that are most similar ($N = 1$) to p under either feature. This set would include q and r but would exclude s, and hence, we would fail to discover the link between p and s. Setting $N = 1$ is clearly an extreme measure that reflects the simplicity of the chosen example. It only serves to illustrate a general point, namely that the approximative method tends to find focal images that are close in *absolute* terms. As a result, the method will tend to enhance any cluster structure that the images may exhibit in feature space.

We have applied the update method on a network of 10,000 images to which we have added 50 images one by one. For each of the 50 images we pool the top 20 images for each of 8 features (i.e. we use U_{20}). The accuracies of the method in terms of the inlinks that were found is illustrates in Figure 7.8.

The values vary between 52% and 93% with an average accuracy of 69%. Thus, already for U_{20} the method finds a fair proportion of inlinks. Greater accuracy could of course be obtained by considering more than the top 20 images.

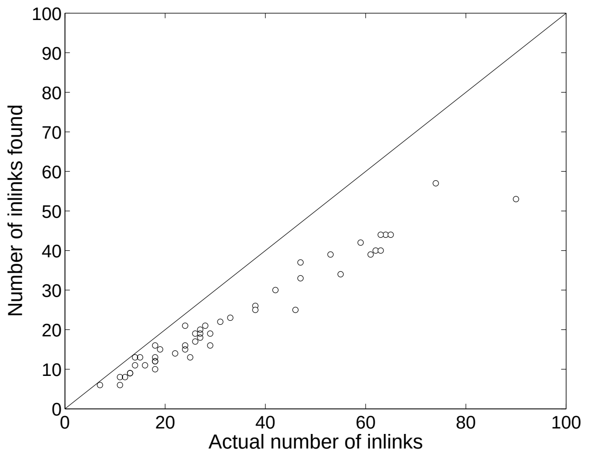


Figure 7.8: Accuracy of update method when considering the top 20 images for each feature.

An important question concerning the updated networks is whether they will remain strongly connected given that they were strongly connected before the update. It is easily seen that the removal of a single vertex may disconnect the network. It may leave behind both sinks and sources and we will therefore have to test for strong connectedness. The case of vertex addition is only slightly more complicated. If vertex addition only involved the addition of new edges, not, however, the rupture of existing edges, the new network could only become disconnected if the new vertex became a source (it could not become a sink because it will have to have at least one NN^k from the rest of the network). This we could easily detect when computing the inlinks of the new vertex. The addition of a vertex may also lead to the removal of any of the outgoing edges of its focal images. Imagine adding an image to a strongly connected network which is a copy of one already present. All the former outlinks of its identical twin will be removed and replaced by one single outlink connecting it to the new vertex. The new vertex in turn will only link to its copy. Vertex addition may thus lead to new sinks and, through the removal of edges, it may also generate new sources. In any case, network update requires the recomputation of the strongly connected components and, if necessary, strong connectedness needs to be restored using the method of Section 7.2.

7.4 Entry Points

7.4.1 Access through search results

In the browsing frameworks suggested by Rubner et al. (1997), Santini and Jain (2000) and Campbell (2000) users begin the browsing process with the set of images retrieved from a search. The initial search places the user in an area of interest which can then be explored further by browsing. This is clearly a possible option for NN^k Networks also. An example how this can work with a collection of Web images can be seen on http://beta.beholdsearch.com.

But as we have argued in Chapter 2 one of the advantages of browsing is that it can potentially free the user from the need of explicitly articulating a query. Unlike the above methods, NN^k Networks are precomputed and exist independently of any query. This suggests at least one way of accessing the network without formulating a query. The quest is for subsets of images that are valuable starting points under a broad range of conditions. Choosing such sets appears to be of particular importance for large collections as the number of images that can be displayed initially become an exceedingly small fraction of the entire collection.

7.4.2 Hubs

Hubs are nodes with a disproportionately large number of outlinks. If users were initially presented with a set of such hubs, they can expect a large choice of images in the subsequent step since each hub has a large number of neighbours. For collections of 30,000 images and eight features, hubs are images with around 70 outlinks (see Figure 6.7 in Chapter 6), a number that only slightly exceeds what we can comfortably display on the screen.

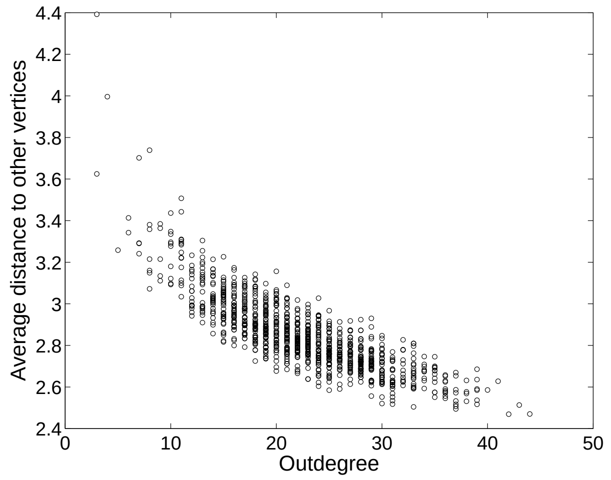


Figure 7.9: Relationship between average distance and outdegree.

It might be argued that instead of maximising the choice for the user, we ought to initiate the search with those images that have a small average distance to all other images in the network. Although we know from Chapter 6 that NN^k Networks have very good connectedness with a diameter rarely exceeding 7, the average distance between vertices can vary considerably as illustrated in Figure 6.6 of Chapter 6 and this is particularly so for small collections. It turns out that the two properties of a vertex, its average distance to others and its outdegree, are strongly correlated: vertices with large outdegrees tend to have small average distances. The scatter plot in Figure 7.9 gives an indication of the relationship. Hubs, therefore, do not merely offer a wide choice of possible entry routes into the collection but are also, as a result of this, closest on average to other vertices in the network. This provides additional support of our view that hubs are suitable starting points if users do not want to or cannot provide a query.

A shortcoming of this method is that it does not necessarily offer the user the best overview of the image collection as hubs need not form a very heterogenous set. Such an overview could be achieved, for example, by automatically identifying densely connected clusters of vertices. We will explore the profitability of this idea in Chapter 8.

7.5 TREC Video Retrieval Evaluation

TREC's Video Retrieval Evaluation (TRECVID), organised by the National Institute of Standards and Technology (NIST), provides an annual opportunity to test image retrieval methods on realistic search tasks and fairly large collections of keyframes from video sequences. The collaborative nature of the event makes it possible to construct relevance judgments and to provide metric performance evaluation which in turn allows a direct comparison between different systems. Although retrieval methods are free to vary, the prevailing method has undoubtedly been search by query image and it was therefore of particular interest to see how much ground could be gained by using NN^k Networks.

7.5.1 Collection, search topics and evaluation

The collection is obtained from about 130 hours of video sequences composed of 271 10–30 minutes programs of ABC World News Tonight, CNN Headline News, and C-SPAN, the latter including government committee meetings, discussions of public affairs, lectures, news conferences and public hearings. The sequences were sorted chronologically and then split into a development collection (the early half) and a test collection (the later half). For each shot (a set of consecutive frames obtained from one continuous camera movement) representative keyframes were extracted making it possible to treat the search for relevant shots as conventional image search. The test collection on which the search was to be carried out comprised 32,318 keyframes.

The information needs are formalised in the form of formatted multimedia statements or 'topics' generated at NIST. Initially, a set of text-only candidate topics were created by mining various news sources from the time period covered by the test collection and by analysing a log of actual queries provided by the BBC. If there exist sufficiently large numbers of relevant shots in the test collection for a particular candidate topic, it was accepted and enhanced with non-textual examples from the Web and from the development collection. Topics consist of a title, a brief textual description of the information need, and multimedia examples (video clips, images, audio). The verbal descriptions of all the 24 topics are shown in Table 7.2. The original topic identifiers are the four digit numbers listed in the table. Topic 120 was later withdrawn from the evaluation as it turned out to be rather trivial. For simplicity, we shall henceforth identify topics by the number formed by the two last digits.

Submission includes for each topic a ranked list of at most 1,000 shots from the test collection and the time elapsed during the search. The allowable maximum time for each topic was 15 minutes. Groups were allowed to submit the results for four different system variants or 'runs'.

With more than 30,000 keyframes, the set of relevant shots for each topic cannot sensibly be identified manually, in particular since for some topics (e.g. topic 104) it would not be sufficient to inspect a representative keyframe to determine the relevance of a shot. On the assumption that a relevant shot is likely to be returned towards the top by at least one of the submitted runs, we should obtain a fair proportion of the relevant shots by merging the top k results of different runs and inspecting the resulting ranked list up to some depth manually. This is the procedure adopted by TRECVID. With relevance judgments thus obtained, each submission is then evaluated to its full depth. The performance measures are average precision and elapsed time for each individual topic and mean average precision (MAP) for each run. Average precision is obtained by computing the precision for every retrieved relevant shot and averaging these over the number of retrieved relevant shots.

100	...with aerial views containing both one or more buildings and one or more roads.
101	...of a basket being made - the basketball passes down through the hoop and net.
102	...from behind the pitcher in a baseball game as he throws a ball that the batter swings at.
103	...of Yasser Arafat.
104	...of an airplane taking off.
105	...of a helicopter in flight or on the ground.
106	...of the Tomb of the Unknown Soldier at Arlington National Cemetery.
107	...of a rocket or missile taking off. Simulations are acceptable.
108	...of the Mercedes logo (star).
109	...of one or more tanks.
110	...of a person diving into some water.
111	...with a locomotive (and attached railroad cars if any) approaching the viewer.
112	...showing flames.
113	...with one or more snow-covered mountain peaks or ridges. Some sky must be visible behind them.
114	...of Osama Bin Laden.
115	...of one or more roads with lots of vehicles.
116	...of the Sphinx.
117	...of one or more groups of people walking in an urban environment, e.g. with streets, traffic, and/or buildings.
118	...of Congressman Mark Souder.
119	...of Morgan Freeman.
121	...of a mug or cup of coffee.
122	...of one or more cats with parts of both ears, both eyes, and the mouth visible. The body can be in any position.
123	...of Pope John Paul II
124	...of the front of the White House in the daytime with the fountain running

Table 7.2: Topics for the search task of TRECVID 2003.

7.5.2 System runs

The system we implemented for TRECVID allowed search by keyword and image query with manual feature selection (S), relevance feedback (R) as described in Heesch and Rüger (2003), and NN^k browsing (B). The four submitted runs differed in the extent to which these three functionalities were available to the users. Labelling the runs by the sequence of letters representing the functional modules that were activated we have the following four runs: SRB, SR, B, and SB. For the third run, which is our principal concern, users were not allowed to formulate a query but could see example images as well as the topic description to guide their search.

NN^k Network construction is based on six visual features and one text feature. We derived the latter from the supplied speech recognition transcripts by first stemming for each keyframe the accompanying annotation terms and then computing term weights according to the standard tf-idf formula. For more details of the visual features, some of which we already described in Chapter 4, please refer to Heesch et al. (2004). The NN^k Network for the test collection using the set of seven features had an average topological distance between vertices of 3.33, an average outdegree of 38.77 and a total of 1,253,074 edges.

Incorporation of this text feature introduces links between keyframes that share a comparatively large number of salient terms and are thus likely to be semantically although not necessarily visually related. Recall that browsing in NN^k Networks takes place following visual cues. At each step, users select the NN^k that are visually the most similar to their target category thus hoping to approach it over a sequence of such steps. It is constructive to analyse how text-induced edges, which do not necessarily involve any strong visual correspondence, fit into this search rationale. Clearly, once we have identified a relevant image, text-induced edges may lead us to other relevant but visually very dissimilar images which we would not otherwise have found. These images may in turn be the gateway to a whole new group of relevant images which it might be profitable to explore using visual cues. It is text-induced edges, therefore, that are well suited for linking up visually coherent relevance subgraphs and thus for widening our catchment area once we have managed to place ourselves in a graph neighbourhood of relevance. Interestingly, text-induced edges are much less useful for finding relevant images in the first place. To see why this is so, one only needs to observe that a necessary condition for images to be linked through a sequence of text-induced edges is that two adjacent images have at least one term in common. Finding an appropriate nearest neighbour at any one step amounts to the task of not only scrutinising the image annotations of the neighbours but also guessing which terms may lead through a chain of co-occurrences to the first relevant image.

For the other three runs involving automated search, the added value of the textual information is different and substantial. It can be exploited much more explicitly in the form of keyword searches making automated search quite possibly a more potent technique for tackling more difficult topics.

With four users participating in the search tasks, a total of 24 topics and four runs, the natural design choice for the search experiments was a 4×4 latin square with the topics being divided into groups of six and the rows and columns signifying, respectively, the set of four participants and the set of four topic blocks. To reduce the effect of learning over the course of several topics, we randomise for each participant the precise temporal sequence of topics within each of the four blocks.

An example of a possible beginning of a search session using NN^k Networks is shown in Figure 7.10 for the case of topic 102. Example images are displayed in the left panel and may guide the users in their search but are not used as queries. The first screen displays in no particular order the 50 most highly connected vertices in the network as motivated in Section 7.2. These images provide the initial access points for the user. Comparison with the set of examples suggests that the image at position (3,3) is visually perhaps the most similar. Clicking that image results in the screen on the right which displays on the main panel the set of NN^k of the selected image with their size being related to the relative support, i.e. the proportion of feature combinations for which that image is ranked closest to the selected image). Images are arranged so that those with large support tend to gather in the centre. The bottom panel shows the selected image in the centre with temporally adjacent keyframes of the same shot placed on either of it. Selecting the image at the bottom right which displays a fair amount of grass brings us to the screen on the bottom left with a number of sport-related images. We find that one of the images in the temporal browser is a relevant keyframe. Selecting it gets us to the fourth screen with relevant images aplenty. Note that the time needed to get from the first to the fourth screen depends only on how fast the user selects images. The set of neighbours are precomputed and need only be retrieved from the database. This makes it considerably faster than search by query image.

7.5.3 Results

The results for the four runs are shown in Table 7.3. The MAP figures for the top six rows are precision-recall values averaged over the 24 topics. The TRECVID mean is the MAP averaged over all submissions. The three runs involving automated search attain very similar performance levels and

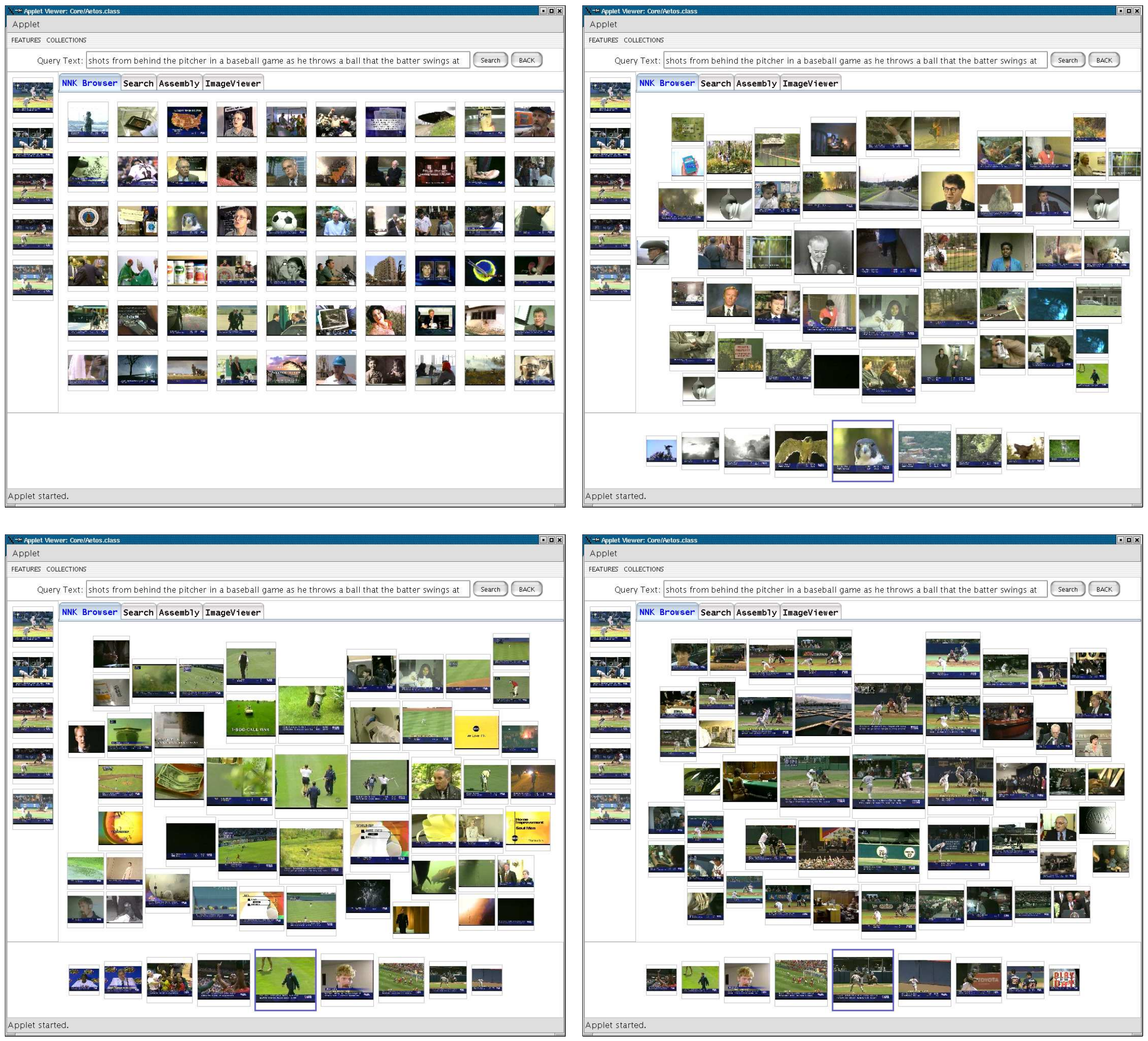

Applet Viewer: Core/Aetos.class
Applet
FEATURES COLLECTIONS
Query Text: shots from behind the pitcher in a baseball game as he throws a ball that the batter swings at
Search
BACK
NNK Browser
Search
Assembly
ImageViewer
Applet started.

Figure 7.10: A TRECVID search using NN^k Networks.

were among the top eight overall. The performance achieved by our browsing run is significantly lower than that of our other three runs and, on the face of it, does not seem to fare particularly well. This is not quite so. The difference can be explained in part by the fact that the search runs could take explicit advantage of the high-quality annotation available, a point which we made earlier. Indeed, for topics with little visual similarity between relevant keyframes, keyword search proved an indispensable tool and its exclusion from the browsing run rendered some topics particularly difficult. This is confirmed by a closer inspection of the performance for individual topics achieved in the browsing run (Figure 7.11). Average precision is particularly low relative to the TRECVID median for topics involving particular individuals, topic 103 (Arafat), topic 114 (Bin Laden) and topic 123 (Pope John Paul II). It is especially these topics that could be tackled effectively through text-based search.

We therefore did not expect performance to equal that of search runs. Nevertheless, the performance of the browsing run exceeds that of 25% of all the submitted runs which we deem quite considerable. For a number of topics with substantial visual coherence among relevant keyframes such as topic 100 (aerial view), topic 102 (baseball) and topic 112 (flames) browsing achieved excellent result that were far above the TRECVID mean for the respective topic. This agrees with our experience at the time of searching. We observed, in particular, that once a relevant keyframe had been found, it was easy to

Runs	MAP	rank out of 36
SRB	0.257 ± 0.219	5
SR	0.259 ± 0.210	4
SB	0.234 ± 0.240	8
B	0.132 ± 0.187	27
TRECVID Best	0.457 ± 0.276	1
TRECVID Worst	0.046 ± 0.051	36
TRECVID Mean	0.182 ± 0.088	

Table 7.3: Search task results for TRECVID 2003.

gather more by exploring the local graph neighbourhood, an observation that is in close accord with the outcome of our semantic analysis from Chapter 6.

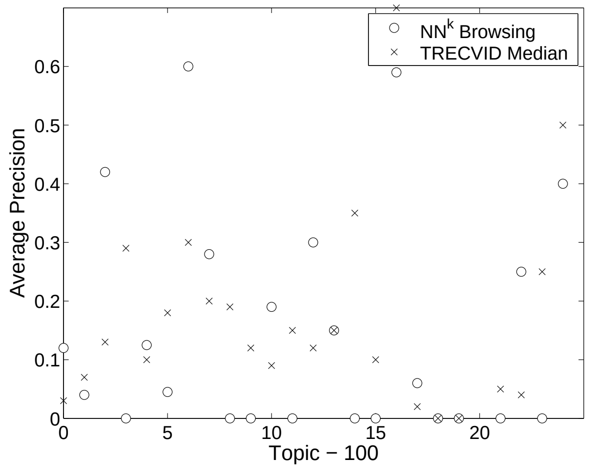


Figure 7.11: Performance according to topic.

7.6 Conclusions

In this chapter we have turned our attention away from theoretical considerations of NN^k Networks to important practical issues culminating in a large-scale performance evaluation of the network structure.

Sinks and sources are groups of vertices that form strongly connected components outside of the giant component. Sinks are composed of near-identical images which share all edges amongst themselves. Sources are composed of single vertices that tend to correspond to images with unusual visual characteristics and are therefore not linked to from any other image. Elimination of both sinks and sources ensures that NN^k Networks are strongly connected so that users can enjoy unrestricted navigability. To identify whether or not a vertex belongs to a sink or a source, it is necessary to determine the set of strongly connected components that partition the network. This operation can be achieved in $O(N)$ where N is the order of the network and thus can be done efficiently even for large collections. Sink elimination involves recomputing for each sink vertex the set of its NN^k with the added constraint that for each weight combination, we determine the most similar image from the complement of the set of sink vertices. Source elimination is accomplished by simply reversing the outgoing arcs of the source vertex. Sink elimination is more expensive than source elimination, but with the proportion of sinks being small (0-2.73%) it adds negligibly to the overall cost of NN^k Network construction.

Being able to update NN^k Networks efficiently is a crucial requirement in dynamic environments. We have seen that an image can be removed from the network simply by recomputing the NN^k of all those images that have previously linked to it. The addition of an image is a somewhat greater challenge as any image from the network could potentially link to it. Instead of recomputing the NN^k for every vertex in the network, we suggest to restrict the set of potential focal images to those that are amongst the N most similar images to the new vertex under at least one feature. The method is motivated by the observations that the great majority of an image's NN^k are among the most similar images under at least one feature and that the NN^k relationship is often found to be reciprocated. We show on a collection of 10,000 images that a substantial proportion of focal images can thus be discovered at a greatly reduced computational cost.

Users have to be given some initial access to the networks. This can be provided through the results of a conventional search by keyword or example image. In cases when the user has either no particular information need or no example image at hand, we propose to display the set of highly connected vertices, the hubs of the NN^k Networks, as these will provide a maximum choice in the next step. It turns out that hubs are also the vertices that have the smallest average distance to the other vertices further adding to their value. A potential shortcoming of hubs is that they do not necessarily provide the user with an overview of the collection, a subject which we will address in the next chapter.

We conceive of NN^k Networks as structures that are not only ideal for undirected browsing but also capable of supporting effective search when the information need is fixed and well-defined. To assess the validity of this second claim, we decided to use NN^k Networks in the search task of TREC's video retrieval evaluation contest and defined an evaluation run in which only browsing was permitted. The level of performance attained by most runs involving search by example query and keywords was out of reach but NN^k Network browsing nonetheless fared better than 25% of all other entries. For individual topics with rich visual coherence among relevant images, browsing NN^k Networks proved at least as effective as search by example. Since we excluded the possibility of keyword searches from the browsing run, a functionality included in the great majority of the submitted runs, relative performance of NN^k browsing is expected to increase noticeably when no textual information is available.

With this chapter we complete the exposition of what we consider to be the core of our NN^k framework for searching and browsing image collections. The natural progression from our current position is to widen the scope for user interaction in NN^k Networks by exposing more of their rich structural and semantic information. The next chapter will take first steps in this direction.

Chapter 8

NN^k Network Clustering

8.1 Introduction

We have seen in Chapter 6 that NN^k Networks organise groups of semantically related images in subgraphs that exhibit a notable degree of compactness. We observed in particular that these subgraphs comprise only a small number of strongly connected components with the largest often seizing a large proportion of the constituent images. NN^k Networks thus appear to provide a concise representation of the semantic structure of an image collection. The combination of representational power with small-world properties accounts for the observation that the value of NN^k Networks goes much beyond that of supporting undirected browsing. As evidenced by the performance in the large-scale search task of TRECVID, NN^k Networks can provide an alternative to automated image search in cases where the information need is well defined.

The effectiveness of image search may surprise in view of the minimal interaction on which it relies and the fact that the semantic structure is never fully exposed to the users. At any one time users interact with little more than a focal image and a set of NN^k. Second-degree neighbours and the compact subgraph structure which the focal image may form part of remain invisible to them and thus play no direct role in facilitating image search. The semantic structure is significant only indirectly as it allows users to gather relevant images by traversing the strongly connected components of relevance subgraphs.

After the consolidation accomplished in the previous chapter we now venture a major leap forward. Our principal goal is to automatically extract information about the semantic structure and to purvey it to the browsing users. If relevance classes form compact subgraphs, we may hope that by identifying compact subgraphs we get close to unveiling meaningful structure. The most apt formulation of our objective is that of finding clusters in the graph. Cluster analysis has been the focus of considerable research in machine learning and pattern recognition and we shall begin our analysis in Section 8.2 by providing a brief, and necessarily incomplete survey of existing techniques.

In Section 8.3 we apply one such technique, Markov clustering (MCL), to NN^k Networks. The technique partitions the vertex set of a graph by simulating flow across the network and has the pleasant properties of (i) converging quickly for large sparse graphs and (ii) being able to deal gracefully with directed graphs. NN^k Networks are largely motivated by the observation that images admit to different interpretations and that this multiplicity cannot be captured satisfactorily by imposing fixed feature weightings. The ability to represent different semantics in one structure is a strength of NN^k Networks which we would like to capitalise on when extracting image clusters. Instead of clustering the vertex set, we propose to cluster the arc set instead. This amounts to applying the algorithm to the dual of the original graph and results in a soft clustering in which images may be assigned to more than one cluster.

One reason for the proliferation of clustering techniques is the difficulty of assessing the quality of a clustering as this makes it difficult to demonstrate the relative merit of a particular method. When information about the true structure of the data is at hand, it is common to validate a clustering by measuring the discrepancy between the set of clusters and the set of classes. An alternative way which we will motivate in Section 8.3.3 is to use non-parametric statistical analysis to test for each cluster individually the null hypothesis that its members have been randomly chosen from the population given the shared annotation between images within a cluster. We apply this validation method to the clustering obtained for the Corel collection comprising more than 30,000 images.

Section 8.4 describes two ways of utilising the information obtained from the graph clustering. We argued in Chapter 7 that, in the absence of an image query, highly connected images from a collection form suitable entry points to the network since they maximise the subsequent choice for the user. The disadvantage of displaying these hubs is that they may not provide a good overview of a collection. Such an overview may better be achieved by exposing the cluster structure of the collection. This could be achieved by displaying centroids of the various clusters extracted from the graph. The second, and arguably more important role of the clustering is as an integral component of an augmented interaction scheme in which the static nature of NN^k Networks is complemented by a cluster selection and expansion method. Because clusters are precomputed, this addition does not compromise but only enhances the degree of interactivity of NN^k Networks.

8.2 Cluster analysis

Several decades of research across numerous disciplines have produced a plethora of techniques for data clustering. A thorough exposition must therefore remain outside the purview of this work and readers are referred to a number of good textbooks that attempt to provide a unified treatment of this disparate subject (Anderberg, 1973; Späth, 1985; Jain and Dubes, 1988; Kaufman and Rousseeuw, 1990).

Cluster analysis seeks to discover natural groups of objects by virtue of the clusters they form. Clusters are typically identified as groups of objects that are more similar to each other with respect to a set of attributes than to other objects. Cluster analysis therefore rests on the general premise that similarity in appearance can be informative about deeper, in our case semantic relationships. As in most other applications of cluster analysis we are only interested in clusters in so far as they tell us something about the classes. Clustering hinges on the notion of the dissimilarity between objects. With data often being represented as points in $\mathbb{R}^n$, it is common to characterise dissimilarities by metric distances. This need not be the case, however. When objects are represented as vertices of a graph, alternative distance measures may be better suited or the notion of a metric distance may be meaningless altogether. We distinguish in the following between vectorial and graph clustering but should hasten to say that this distinction is more out of convenience for the differences are often slight.

8.2.1 Vectorial clustering

The conventional data model underlying most clustering methods is that of a vectorial representation of objects in $\mathbb{R}^n$. Given a metric, the set of data points can be mapped to a pair-wise distance matrix which provides the common starting point of virtually all cluster methods. It is commonly distinguished between three broad groups of clustering methods according to how they utilise the pair-wise distance information: (i) Partitional methods such as k-means divide the data into a fixed number of non-overlapping sets so that some score function is minimised. Often the methods involve iterative optimisation to find local optima. (ii) Hierarchical methods obviate the need to specify the number of clusters. Different partitions can be obtained from the cluster hierarchy simply by taking cross sections at different levels. Because of their nested nature, the resulting clusters can cover the whole gamut of granularity from the more general to the more specific and have been the structure of choice for a number of browsing models (for examples in image retrieval see Ma and Manjunath, 1996; Krishnamachari and Abdel-Mottaleb, 1999; Swets and Weng, 1999; Chen et al., 2000; Pečenović et al., 2000; Barnard and Forsyth, 2001). (iii) Probabilistic methods, finally, seek to find an optimal description of the data by fitting generative models. The result is not a partitioning of the data but for each data point a vector of probabilities of belonging to each of the different clusters. The information it yields is richer but it comes at the price of much tighter, and potentially incorrect distributional assumptions.

8.2.2 Graph clustering

When data are represented in the form of a graph, cluster analysis can be formalised as a graph partitioning problem. The quest is for a partitioning of the vertex set such that the subgraphs induced by each of these sets are compact (in the same, similarly loose sense as that which motivated our analysis of Chapter 6). As has been noted by van Dongen (2000), a graph may be regarded as being closer to the optimal partitioning in the sense that a disjoint union of complete subgraphs allows only one sensible partitioning and no other graph would justify such a partition better. For vectorial data this correspondence could only be achieved in the limit of all points of a cluster being infinitesimally close to each other and any two clusters being an infinite distance apart. In practice, however, we do not expect graphs to be so clear-cut and for most graphs different optimisation criteria will produce different partitionings.

Graphs are fully described by their adjacency matrix which we may conveniently construe as a matrix of affinities to highlight the close connection to the distance matrix employed in vectorial clustering. It is then easy to see that given data in $\mathbb{R}^n$, we can readily cast it into a weighted graph representation by turning the pair-wise distance matrix into an adjacency matrix with entries being some monotonically decreasing function of the original distances. Thus, partitioning methods that are most naturally formulated in a graph context can readily be applied to vectorial data by turning the latter into a graph. In fact, in the great majority of applications of graph clustering, the graphs are ultimately derived from vectorial data. NN^k Networks, clearly, are no exception as their structure is entirely determined by the distance matrices of individual features.

Spectral clustering

One class of graph clustering methods that are most commonly employed in conjunction with vectorial data but take their origin in a graph context are spectral methods. These have emerged in a number of fields including computer vision and VLSI (Very Large Scale Integration) design. Despite the substantial variation between particular spectral methods (Chung, 1997) their common objective is to find the top eigenvectors of the Laplacian L of the graph's adjacency matrix A. The second eigenvector, for example, induces a partitioning of the graph into two parts which can be shown to approximate the optimal partitioning for a particular optimality criterion. By employing several eigenvectors simultaneously, one can achieve a k-way partitioning (Alpert et al., 1999; Malik et al., 2001; Meila and Shi, 2001; Ng et al., 2005). The shortcoming of spectral clustering lies in its computational complexity making it prohibitively inefficient for large matrices comprising several thousands of rows.

Graph contraction

Graph contraction methods are similar in rationale to that of agglomerative hierarchical clustering of vectorial data points. In Song et al. (2005) a group of vertices is contracted to one vertex if their induced subgraph contains only edges with weights above some threshold. By decreasing the threshold over a number of steps, successive graph contraction leads to a nested hierarchy of clusters. In Wu et al. (2004) the contraction step is preceded by a computation of so-called median vertices. These are vertices located in the graph such that the average distance of a vertex to its nearest median vertex is minimised. The problem of finding these median vertices is similar in nature to that known as central clustering or vector quantisation of points in a vector space (Gersho and Gray, 1992). The partitioning of the graph can loosely be thought of as the graph equivalent of a Voronoi tessellation. In the ensuing contraction step all vertices of a partition are mapped to its median vertex. By repeating the two steps the entire graph can be collapsed onto one root vertex.

We deem graph contraction methods unsuitable for our purpose as they place a premium on local proximity favouring subgraphs which have a small average distance between constituent vertices. As such, these methods will succeed in finding subgraphs with sphere-like topology but will fail to discover those that are elongated, a property that many of the compact relevance subgraphs of NN^k Networks seem to exhibit.

Clusters as connected components of neighbourhood graphs

A common way to convert data in $\mathbb{R}^n$ into a graph is to represent each data point as a vertex and to connect two vertices if the distance between their vectorial representations lies below some threshold ϵ. The resulting neighbourhood graphs often form the basis for spectral clustering described above but clustering can be obtained more directly by identifying clusters with connected components of the graph. Clusterings of various granularity can then be obtained by varying the threshold (Duda et al., 2001; Cheung and Zakhor, 2005). For the purpose of non-hierarchical partitional clustering, the problem of how to determine ϵ in a principled manner is non-trivial and it is often found that the resulting graphs consist of many isolated connected components (Belkin and Niyogi, 2002; Ng et al., 2005).

A somewhat different approach to graph clustering has been taken by van Dongen (2000) in his doctoral thesis. Like many of the above-mentioned methods, the Markov cluster (MCL) algorithm is motivated by the observation that groups of vertices should be considered forming a cluster if they share many of their edges and are connected to comparatively few vertices of other such clusters. Such groups are found not by minimising some objective function, but by defining a Markov chain on the vertex set and by simulating a random walk across the graph. Each vertex corresponds to a distinct state of the chain and the transition probabilities are given by the normalised adjacency matrix of the graph, the transition matrix. MCL exploits the fact that a random walk will spend most of its time *within* highly connected regions and will rarely cross boundaries between such regions.

The standard procedure of simulating a random walk of length n is by raising the transition matrix P to the nth power. $(P^n)_{ij}$ then specifies the probability that a random walk reaches vertex j in n steps when starting at vertex i. For strongly connected graphs, the corresponding Markov chain will have a unique stationary distribution so that as $n \to \infty$ the probability of being in a particular state becomes independent of the initial state. The crux of the MCL method is that instead of letting the Markov chain reach its stationary distribution, it amplifies large transition probabilities by a process called inflation. Inflation involves element-wise potentiation by some power I and is followed by a normalisation step to render the matrix stochastic again. By alternating matrix multiplication with inflation, groups of strongly connected vertices eventually become segregated as most transition probabilities vanish. The final matrix is exceedingly sparse and when interpreted in terms of an adjacency matrix corresponds to a disjoint set of subgraphs each of which being a tree.

The MCL algorithm has been applied extensively in a number of areas most notably in molecular biology for detecting protein families (Enright et al., 2002, 2003; Li et al., 2003) and in computational linguistics for word sense disambiguation (Dorow and Widdows, 2003; Dorow et al., 2004). We chose it over other clustering methods because it seems a robust and fairly efficient method that can be applied easily to directed graphs. Although the computational complexity of the algorithm is quadratic in the number of vertices and linear in the number of edges, we note that graphs with 100,000 vertices and 2,500,000 edges can be clustered in only one hour on a 3 GHz processor.

8.3 MCL on NN^k Networks

8.3.1 Clustering of the original network

The MCL algorithm achieves a strict partitioning of the vertex set so that a vertex is assigned to exactly one cluster. When applying MCL to the original network, therefore, it is done on the premise that there exists a privileged class to which an image belongs. This assumption seems at first to be at odds with our claim that images are polysemous. The two views are not irreconcilable, however, as it could be argued that there usually exists a semantic facet of an image that stands out and which is agreed upon by most users. Clustering the original network can hence be justified if we thereby hope to discover the most likely class of an image.

We find experimentally that the results obtained from clustering the original network are very sensitive to changes in the inflation parameter I. Recall that I denotes the power to which each matrix element is raised after matrix multiplication. It is the inflation and subsequent normalisation step that have the effect of strengthening flow between strongly connected vertices. High values for I let the matrix converge quickly before flow from one area can spread into other areas of the network. The result is a clustering that captures local structure and in the extreme case groups an image with its strongest NN^k. For low values of I, flow becomes global before the matrix converges and the final result reflects structure at a larger scale. The relationships between the number of clusters, the size of the largest cluster and the value of the inflation parameter for the Corel 10,000 collection are depicted in Figure 8.1.

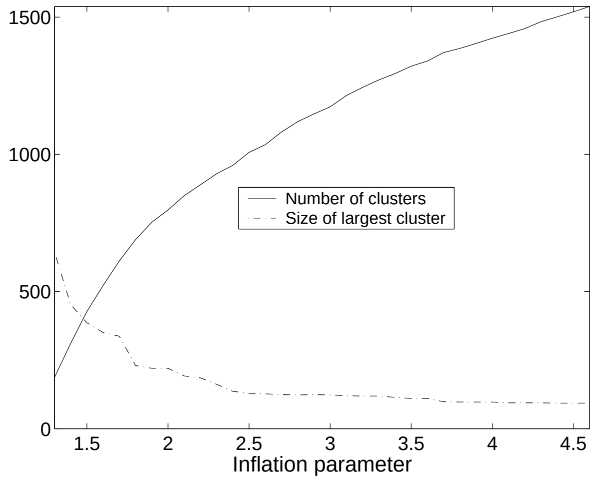


Figure 8.1: Relationship between number of clusters, maximum cluster size and I.

8.3.2 Modelling semantic ambiguity - MCL on the dual network

MCL clustering breaks the graph into equivalence classes with every vertex being assigned to exactly one cluster. The shortcoming of the hard clustering thus obtained is its failure to model the semantic ambiguity of images that to capture was the original motivation behind NN^k Networks. By treating each vertex as an indivisible semantic unit, MCL ignores the semantic multiplicity of images. As we argued in Chapter 4, it is not individual vertices but vertex pairs with which we identify the semantic atoms in NN^k Networks. We should therefore hope to achieve a more faithful representation of the semantic structure of a collection by clustering vertex pairs. Because a vertex participates in a number of such pairs (as many as it has neighbours), clustering vertex pairs allows vertices to belong to different clusters. Clustering vertex pairs of a graph $G = (V, E)$ can be understood as clustering the vertices of the dual of G. The dual of a graph, $G' = (V', E')$, is obtained by converting each edge in G into a vertex of G' and connecting any two vertices in G' if their corresponding edges in G share a vertex. Because the set of edges incident with a vertex are mutually adjacent, a vertex v in G of degree k_v expands into a complete graph in G' of order k_v (and thus with $k_v(k_v-1)/2$ edges). Hence, the number of edges and vertices of G', denoted by n' and e', are

$$\begin{aligned} n' &= e \\ e' &= \frac{1}{2}\sum_{v\in V} k_v(k_v-1) = \frac{n}{2}\left(\mathrm{Var}[k] + \mathrm{E}[k]^2 - \mathrm{E}[k]\right). \end{aligned}$$

Note that the number of edges in G' cannot simply be inferred from the average vertex degree of G as it also depends on its variance. The dual of a graph contains more vertices than G if the average vertex degree is greater than two. From the above expression of e', it follows that it also contains more edges than G whenever $2\mathrm{E}[k] < \mathrm{E}[k^2]$. Thus, it is generally more costly to cluster the dual than the original graph.

For directed graphs like NN^k Networks we can find a dual by a slight modification of the above definition. Instead of establishing an edge in G' if the adjacent vertices correspond to adjacent edges in G, we establish an arc in G' from v to w if the corresponding arcs of v and w in G share the same vertex x and are such that v is directed towards x and w is directed away from x. If we do not want to retain the directionality of the original graph, we may instead convert it first into an undirected graph by establishing an edge between two vertices if there is at least one arc involving both and subsequently construct its undirected dual.

We adopt the former technique and so keep the dual directed. The conversion involves connecting each of the outgoing arcs of a vertex v with the outgoing arcs of w where $v \to w$. Hence, the total number of arcs in the dual graph is given by

$$e' = \sum_{v \in V} \sum_{w \in V : v \to w} k_w \tag{8.1}$$

We note as an aside that this is equal to the sum of the elements of A^2, where A is the adjacency matrix of the original network in which all non-zero weights are replaced by unity. The weight of an edge in the dual graph is taken to be a function of the weights associated with the two edges it connects in the original graph. There are several ways how these weights may be combined. For our experiments we chose $w_{ij} = \sqrt{w_i^2 + w_j^2}$, which has the effect of the larger weight dominating the average. Clearly, the weights thus assigned do not generally add to unity even if they do in the original graph, so they need to be normalised prior to applying the MCL algorithm.

The arc set of G can now be partitioned by applying the MCL algorithm to the vertex set of G'. As before we obtain a strict partitioning of the vertex set but since each vertex now corresponds to an arc in the original graph, the result amounts to a soft clustering of the original vertex set. A vertex can now associate with as many clusters as the product of its in- and outdegrees, although in practice we may expect many of a vertex's incident edges to fall in the same cluster.

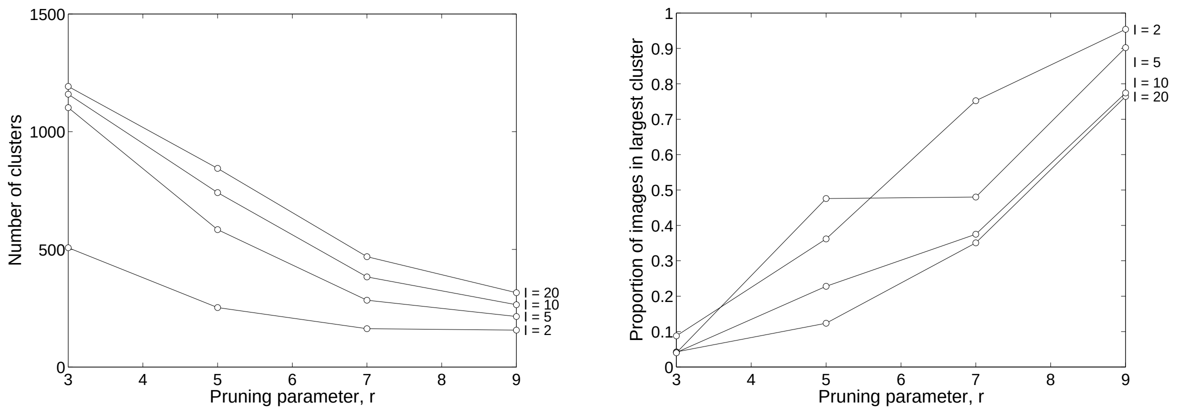
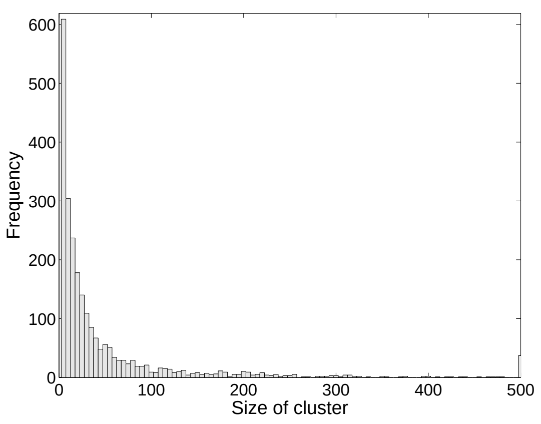


Figure 8.2: Effect of r and I on the number of clusters.

As we noted above the dual graph is typically much larger than the original graph. Not only do we have more vertices but also more edges (even though the graph may well be sparser). One way to reduce the number of both edges and vertices in G' is by not making every edge in G a vertex of

G'. In particular, we may restrict our attention to the r strongest outgoing arcs for each vertex. The total number of edges in G that become vertices in G' is then reduced to at most rN where N is the number of images in the collection (rN is an upper bound since the degree of some vertices may be smaller than r). It is easily seen that the number of edges is then at most r^2N and so decreases quadratically with r.

By thus pruning the network we not only achieve a faster clustering but also appear to improve the quality of the resulting clusters. We find experimentally that both the number of clusters and the distribution of cluster sizes change considerably as we vary r as indicated in Figure 8.2 for the Corel 10,000 collection. For large r an increasing proportion of the images are concentrated in the largest cluster. Allowing all edges to become vertices results in a number of clusters covering virtually the entire image collection. Although large clusters may not in themselves be undesirable (there may after all exist broad categories such as indoors and outdoors), we observe that their preponderance is concomitant with a rapid decrease in the frequencies of clusters at other semantically meaningful scales. Weak arcs seem to have a confounding effect on the clustering process and prevent the MCL algorithm from finding tight groups.



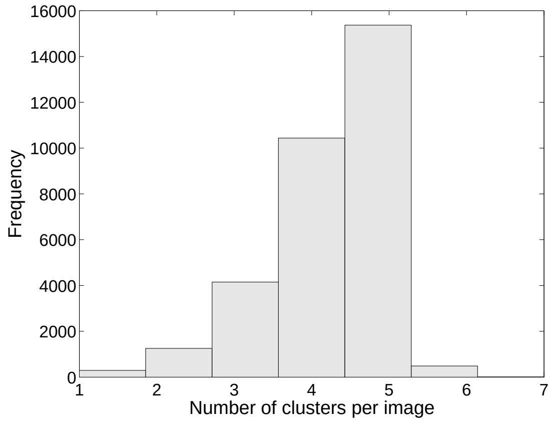


Figure 8.3: Cluster sizes (left) and number of clusters per image (right) for $I = 20$, $r = 5$ and 32,000 images.

The recommended range of values for the inflation parameter I is between two and five and indeed we find that $I = 2$ proves a good value for clustering the original network. For the dual graph, however, we observe that the number of clusters remains at a prohibitively low level even for $I = 5$ and that results do improve by setting I as high as 20. The distribution of cluster sizes for $I = 20$ and $r = 5$ is shown on the left of Figure 8.3. Most clusters are in a range that can be displayed comfortably on a computer screen. The plot on the right shows the frequency distribution of the number of clusters to which images have been assigned to. With $r = 5$, the great majority of images are shared by five clusters with very few belonging to six clusters. Note that r is the maximum number of outgoing arcs of a vertex that we allow to take part in the clustering. Since each vertex also has incoming arcs, however, the total number of clusters a vertex can be assigned to can exceed r.

8.3.3 Cluster validation

Visual inspection of the clusters suggest that the constituent images tend to be closely related. A select few of the 2,399 clusters are shown in Figure 8.4. We shall now consider more objective ways of assessing cluster quality. When the clustering is a strict partitioning and we know the classes which we hope to reveal through the clustering, its quality can be measured in terms of the discrepancy between the class and the cluster distribution. A number of simple metrics have been proposed for this purpose (Rand, 1971; Fowlkes and Mallows, 1983; Milligan et al., 1983) which are based on a 2×2 contingency table that records the number of object pairs that agree or disagree with the class and cluster to which they belong. Given N objects, we have a total of $N(N-1)/2$ object pairs to examine. The larger the number of object pairs that either disagree or agree about both their class and cluster membership, the better the clustering.

A related method counts for each cluster/class pair the number of objects that associate with it and thus involves an $m \times n$ contingency table where n and m denote the number of clusters and classes, respectively. For a perfect match the matrix would consist of exactly one entry per row and one per column. A metric can then be defined in various ways using, for example, empirical conditional entropy (Dom, 2001) or the normalised Hamming distance (Kanungo et al., 1994).

Both approaches measure the quality of the clustering in terms of the discrepancy between the actual and the ideal clustering. Their success relies on the convenient properties of the clustering being a strict partitioning and of objects being uniquely assigned to classes, neither of which is true in our context. We therefore suggest an alternative statistical method that generalises to multiple class and cluster memberships. Instead of measuring how far the actual clustering deviates from the ideal one, we measure its compatibility with the null hypothesis that images within a cluster are assembled randomly. We choose as the test statistic the size of the joint vocabulary associated with a cluster. For convenience we shall call this the description size of a cluster. The choice is based on the not unreasonable premise that the description size is related to the semantic coherence of a cluster: the more terms we need to describe a cluster, the greater the diversity among its members and the lower the quality that we may wish to ascribe to the cluster. Unlike the first two methods, our approach measures the quality of individual clusters rather than of the clustering as a whole but the two are clearly related and the latter can be inferred from the former.

The distribution of the test statistic under the null hypothesis may seem difficult to gauge for not only does the number of terms vary between images, also the overall frequency distribution of terms is highly non-uniform with many terms occurring only once and a few terms like “background” and “outdoor” accompanying more than 60% of the images of the Corel collection. For a small random set of images, however, we may consider the overlap between annotations within a cluster negligible (the overlap will largely be with respect to the few terms occurring in a large proportion of images) so that the description size can be modelled as the sum of identically and independently distributed random variables. By the central limit theorem, therefore, we expect the distribution of the description size to be well approximated by a normal distribution for large clusters. Figure 8.5 shows the description sizes of random sets of order 60. It provide visual confirmation of this conjecture.

Given the distribution of the test statistic under the null hypothesis, we can measure the significance of actually observed description sizes in terms of the probability mass to the left of that value under the null distribution, i.e. the P-value. Instead of fitting a normal distribution and looking up the corresponding P-values in statistical tables, we take a more direct, and arguably more accurate Monte Carlo approach by determining the proportion of random clusters with a description size smaller than the observed value. We approximate the distribution by generating for each observed cluster size 10,000 random image sets of that size for each of which we determine the value of the test statistic (this was also done to generate Figure 8.5 but with a sample size of 1,000).

For each cluster we obtain a P-value which gives the probability that the description size for a cluster

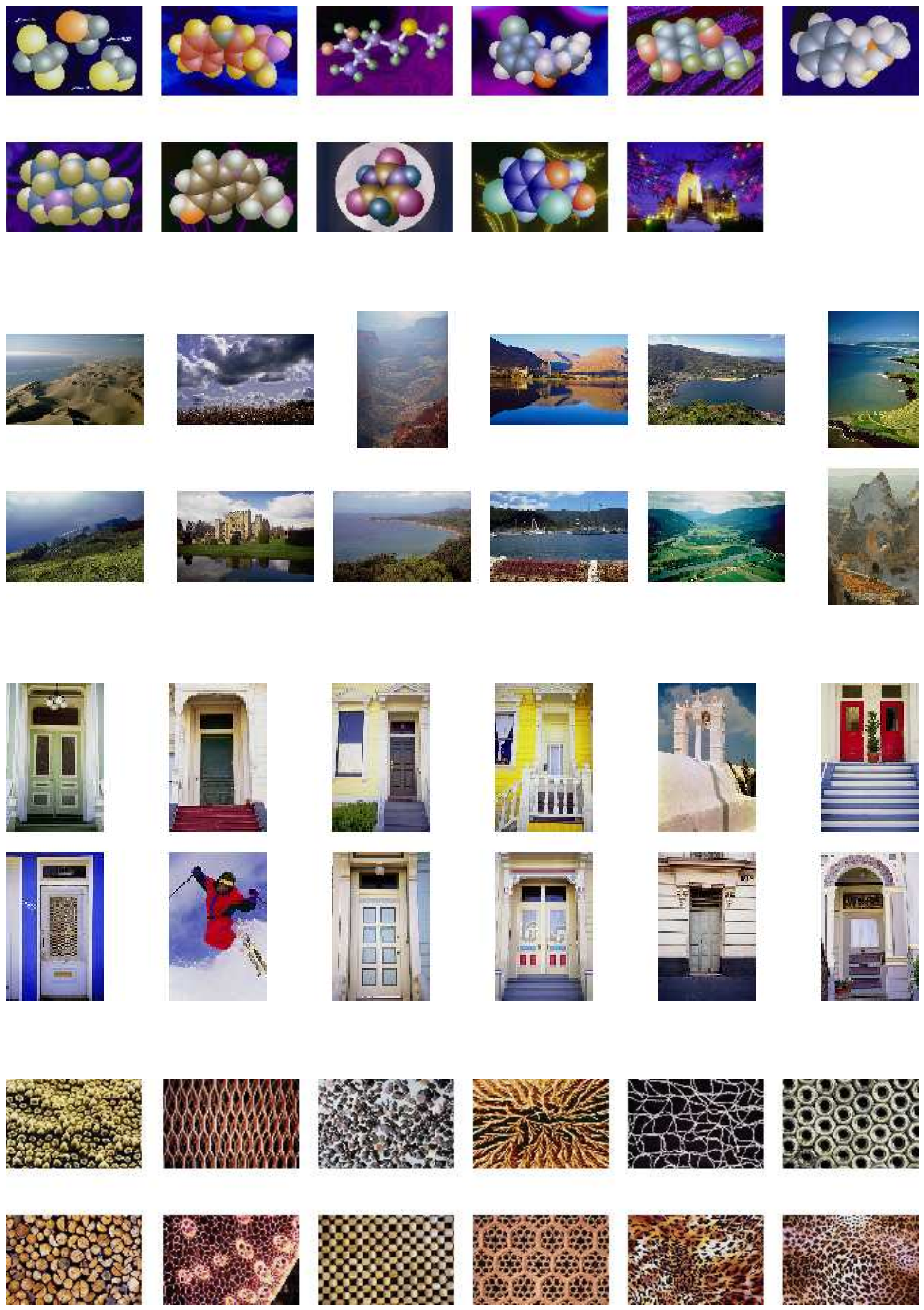


Figure 8.4: Clusters obtained by running MCL on the dual network of 32,000 images.

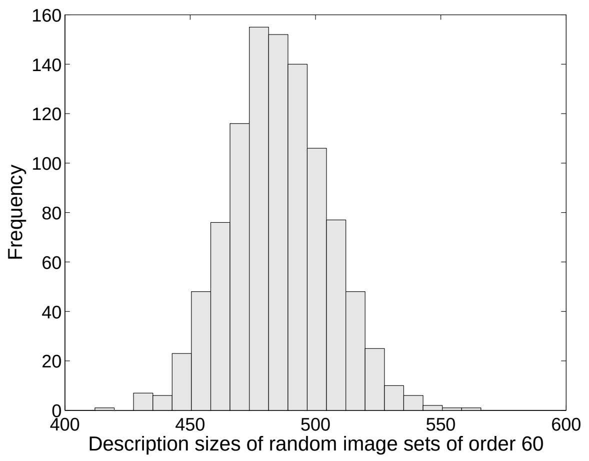


Figure 8.5: Frequency distribution of description size.

of this size would have been as low as or lower than the one observed had the cluster been generated at random. We consider a cluster significantly deviant if the P-value falls below some significance level α.

P-values have been a notorious source of contention and confusion among statisticians and their frequent abuse has brought statistical hypothesis testing somewhat into disrepute. A common misapprehension is the belief commonly found in textbooks that the P-value measures the probability of the null hypothesis being true. If x is the value of our observed test statistic, then by Bayes' rule, $P(H_0|x) \propto P(H_0)P(x|H_0)$, so the probability of H_0 being true in the light of x clearly depends also on our prior belief in H_0. Another problem, which is of particular relevance in our context, arises whenever multiple tests are carried out and are followed by an *ad hoc* selection of those that are significant. Under these conditions the chance of erroneously rejecting a true null hypothesis increases. Imagine, for example, that we test two true null hypotheses with $\alpha = 0.05$. The chance of rejecting at least one of the hypotheses is $1-(1-0.05)^2 = 0.0975$, not 0.05. The problem is appropriately called alpha inflation and much effort has been expended to control the risk of type-I error thus introduced (Hochberg and Tamhane, 1987; Tukey, 1991). With the growing need for automated evaluation of large-scale experiments, in particular microarray experiments in functional genomics involving thousands of pairwise comparisons, the problem of multiple comparisons is currently attracting considerable attention (Simonsen and McIntyre, 2004). A simple method that guards against false positives is known as the Bonferroni correction which reduces the significance level for each test to α/m where m is the number of tests.

Such corrections are important if we are primarily concerned with validating individual, independent hypotheses. In our context, the cluster-specific hypotheses all provide partial evidence in favour of or against a global hypothesis that the clustering as a whole achieves a non-random segregation and the individual tests may be highly non-independent (in the case that the clustering achieves its purpose). What matters for us, therefore, is not the P-value of individual clusters but the distribution of P-values for the ensemble of clusters. For a similar argument in a somewhat different context see the paper by Böhm and Mayhew (2005). If segregation were random, the distribution of P-values would be expected to be symmetrically centred at 0.5. Given some significance level α and m clusters, significant P-values would be expected in $\alpha \times m$ clusters. The actual distribution of P-values for the clusters found in the Corel collection is shown in Figure 8.6.

From the 2,399 clusters, 657 or 27% score a P-value of less than 0.0001, 1446 or 60.2% of less than 0.05. The preponderance of low P-values is not what we expect under the null hypothesis and it strongly suggests that the clustering is of much better quality than what would be expected by chance. This analysis confirms the impression one gains by visual inspection of individual clusters.

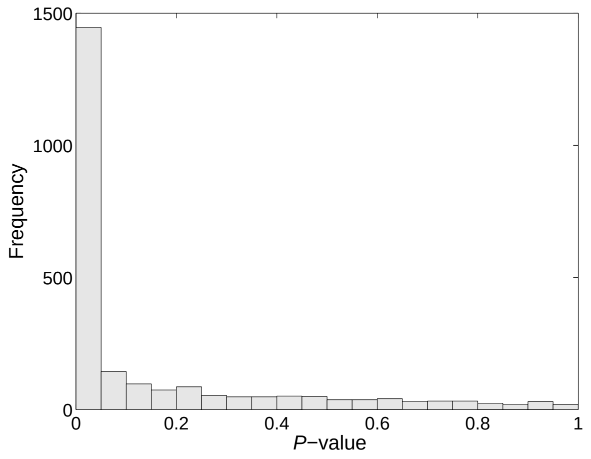


Figure 8.6: Distribution of P-values for 2,399 clusters.

Judging by the P-value, we observe from Figure 8.6 that although the clustering on the whole appears to capture the semantic structure of the collection well, not all clusters are assigned high P-values. If annotations are available, we can readily prune the ensemble of clusters based on some threshold value for P. If they are not, it might be of interest to conceive of alternative ways to measure cluster quality. The problem amounts to finding some property other than small description size that is shared among clusters with low P-value and only these. It is now an opportune moment to turn to an important hypothesis that the NN^k framework almost naturally gives rise to. If the hypothesis were true, it would provide us with such a property. Unfortunately, we shall find that the hypothesis is most likely to be false but we will argue that this result further strengthens the very premises on which the NN^k framework rests. We have repeatedly emphasised that the semantic unit in NN^k Networks are pairs of images that correspond to adjacent vertices, not individual images. This is an expression of the general view that the meaning of an image reveals itself only by placing it in relation with others. These pairs consist of a focal image and one of its NN^k and are partly characterised by the average of all the weight vectors for which that NN^k is more similar to the focal image than is any other image (see Section 3.2.6 in Chapter 3). It is tempting to construe weight centroids as carrying a certain semantics and to conjecture that similar weight centroids carry a similar semantics. If this were so, we expect the set of weight centroids corresponding to semantically coherent clusters to form a comparatively homogenous set with low variance, and thus we should be able to infer cluster quality from the homogeneity of weight centroids. We have put this hypothesis to a test using the clustering obtained from the Corel collection. For each of the 2,399 clusters, we note the weight centroid associated with each of its edges. We then determine the variance with respect to each of the components of the k-dimensional weight vector (where k is the number of features used) across all edges from a cluster and then average over all k components. This forms our aggregate measure of within-cluster variance. For each cluster, we may now plot its variance against the previously computed P-value. In order to make the plot more easy to interpret, we do not plot each of the $2,399$ points but compute the variance at 10 equally spaced P-values p_i averaged over a neighbourhood $p_i \pm 0.05$ covering the interval $[0, 1]$. The results are depicted in Figure 8.7. The dotted lines mark the standard deviation and we linearly interpolate between points. Clearly, if we wished to discern a trend, then it would be one in a direction opposite to what we expect were our hypothesis correct. On the basis of these data we have no reason to assume that the variance is smaller for high-quality, semantically coherent clusters.

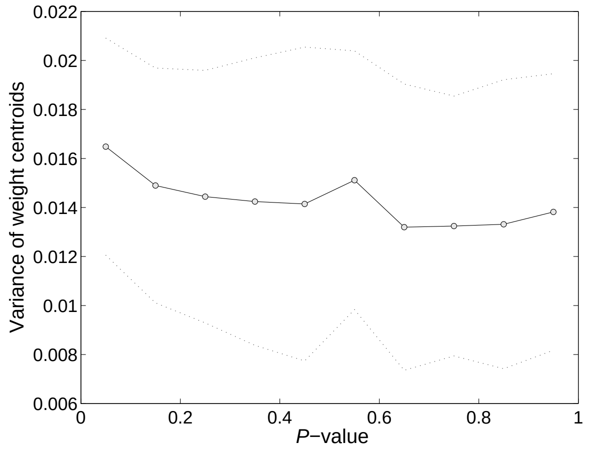


Figure 8.7: Variance within clusters with respect to weight centroids for different P-values.

Although this is undoubtedly a negative result in the narrow context that motivated the analysis in the first place, it supports a conclusion of wider significance: The NN^k idea is predicated on the observation that the multiplicity of semantic facets of an image prohibits any fixation of feature weights prior to user interaction. We resolve the dilemma of feature weighting by computing the nearest neighbours under all possible weighted combinations of feature distances. This forms the basis of NN^k search in Chapter 4 and of NN^k Networks as developed since. The discovery that the feature weights which give rise to semantically coherent clusters are not necessarily homogenous suggests that even to capture individual semantic facets it may not be sufficient to guess a single global metric. If image classes describe complex distributions in feature space, any global distance metric induced by a fixed set of feature weights should often prove locally suboptimal. Relevance feedback techniques that aim to optimise a single metric from positive examples are therefore severely limited as the learned parameters will provide an inadequate model for the complex metric structure of the class as a whole. This echoes the observations of Li et al. (e.g. 2002); Qamra et al. (e.g. 2005) which motivated their introducing dynamic distance functions. We could of course estimate the 'correct' local metric if the class distribution were known. Otherwise, however, we are well advised to try different metrics and let the user decide which one is locally optimal. This is precisely what NN^k Networks seek to achieve by representing the neighbourhood of an image in different feature directions. When the distribution of a particular class falls on a low-dimensional manifold, most metrics will cause non-relevant images to be woven into the network, but by also including the locally optimal metric, we have a high chance of connecting the current focal image to a member of the same relevance class.

It is common practice to learn manifolds by approximating the topology of the data by means of neighbourhood graphs (Tenenbaum, 1997; Tenenbaum et al., 2000; Belkin and Niyogi, 2002, 2003; He et al., 2004). The geodesic distance between data points can then be approximated as the topological distance across the graph and dimensionality reduction techniques may then be applied to the geodesic distance matrix. Because NN^k Networks employ a set of different metrics to connect each data point, we may fruitfully construe them as approximations of a large number of possible manifolds. The approximations are embedded in the network structure in the form of the relevance subgraphs encountered in Chapter 6 with paths along the subgraphs corresponding to trajectories along manifolds.

8.4 Applications of clustering

8.4.1 Collection overview

In a query-free environment alternative ways of accessing the network need to be found. We argued in Chapter 7 that highly connected vertices or hubs are suitable candidates for initial points of access as they lead users quickly into different areas of the collection. For users unfamiliar with the content of a collection, it might be a more pressing objective to gain an overview of the collection. The clusters obtained from the MCL algorithm can achieve this purpose by representing the semantic structure of the collection. If annotations are available to check cluster quality directly, we can choose a subset of high-quality clusters obtained from the dual which cover the collection well. Even a subset should provide good coverage as the dual contains substantial overlap. For display purposes we need to determine cluster representatives. For vectorial data the standard approach is to compute cluster centroids as those data points that minimise the average distance to all other points of the cluster. For graph data we may define centroids in a similar way by measuring distance in terms of the length of the shortest path between two vertices. We find in practice that the centroids thus found tend to be veritable outliers. This is a result of the fact that the subgraphs often have a tree-like topology with arcs directed towards the centre. A second possibility is to examine the behaviour of the Markov chain defined on the subgraph corresponding to the cluster. If P is the transition matrix derived from the adjacency matrix through row-wise normalisation and $p_{ij}^n = (P^n)_{ij}$ then the probability of finding the random walk at vertex j can be obtained by finding the limits

$$\lim_{n\to\infty} p_{ij}^n.$$

If the Markov chain is irreducible (i.e. the subgraph is strongly connected) and aperiodic, then the limit is the same for all i and denoted by π_j. To ensure strong connectedness we reverse each arc of the subgraph. The maximum period of the resulting graphs is then two. If the period is two then we can obtain a stationary distribution by averaging entries over two successive matrices.

$$\pi_j = \frac{1}{2}\lim_{n\to\infty}\left(p_{ij}^n + p_{ij}^{n+1}\right).$$

Vertices can now be ranked according to their π_j value and the most frequently visited vertex can be chosen as the cluster representative. We find that the resulting centroids represent the clusters much better than the first, distance-based method.

As an aside it might be interesting to note that by displaying not the hubs but cluster representatives we may expect to increase the chances that the initial display contains exactly one relevant image because we are less likely to find two or more if the clusters really capture different relevance classes. The reasoning is similar to that in Chapter 4 where we argued that the chance of finding exactly one relevant image among the NN^k of an image should be higher than were we to retrieve the same number of images with some fixed metric.

8.4.2 NN^k Clusters

We can take further advantage of the clustering by selectively informing users about any potentially relevant structure associated with their current position in the network. When choosing a vertex as the new focal image, we display not only the set of NN^k but also the representatives of all the clusters in which the image appears. We may call these clusters NN^k clusters as they contain the current focal image and at least one of the focal image's NN^k. The number of possible clusters for each focal image is given by the product of their in- and outdegrees if all arcs participate in the construction of the dual graph. If we only consider the five strongest arcs of each vertex for dual computation, the actual distribution for 32,000 images is given by the right histogram of Figure 8.3.

Again we may think of different clusters as highlighting different semantic facets of the current focal image. Unlike the display of NN^k the clusters are meant to be semantically uniform and as such they provide users with images that would otherwise need to be collected by following relevance paths through the network. By choosing a cluster, users choose a particular semantics of the current focal image. The process of cluster selection and expansion is therefore akin in rationale to NN^k search of Chapter 4. While the latter involves ranking the entire collection by applying a fixed weight set, the former matches different semantics with precomputed clusters.

Figure 8.9 is a screenshot of a prototype system that provides a physical instantiation of the NN^k framework by integrating NN^k browsing, NN^k search with NN^k cluster selection and expansion. The partitioning of the interface is shown diagrammatically in Figure 8.8. The bottom panel relies on a linear organisation of images according to some metadata (time a photograph has been taken, order in a directory etc.) and shows the focal image in the centre. The left panel displays the set of NN^k that are associated with the current focal image. The size of each image reflects its support, i.e. the number of weight sets for which it is closest to the focal image. The top panel on the right contains a list of five cluster centroids. Each of these clusters contains the focal image and some of the NN^k found on the left panel. By selecting a cluster centroid, the content of the cluster is expanded in the panel underneath. Different centroids expand into different NN^k clusters. Here the rightmost centroid represents a cluster with landscape images predominantly with meadows or grassland in the foreground.

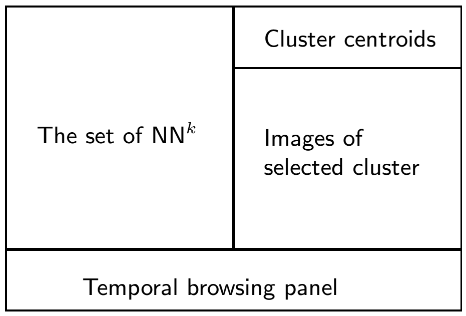


Figure 8.8: Schematic interface.

It is interesting to note that there is common ground between this idea of cluster selection and expansion and some relatively early work on image retrieval. Minka and Picard (1996) suggest to precompute a large number of image groupings by applying a variety of different similarity criteria, or 'society of models'. The method seeks to capture as many of the potentially relevant groupings beforehand and is motivated by the observation that for very large collections, it becomes impractical

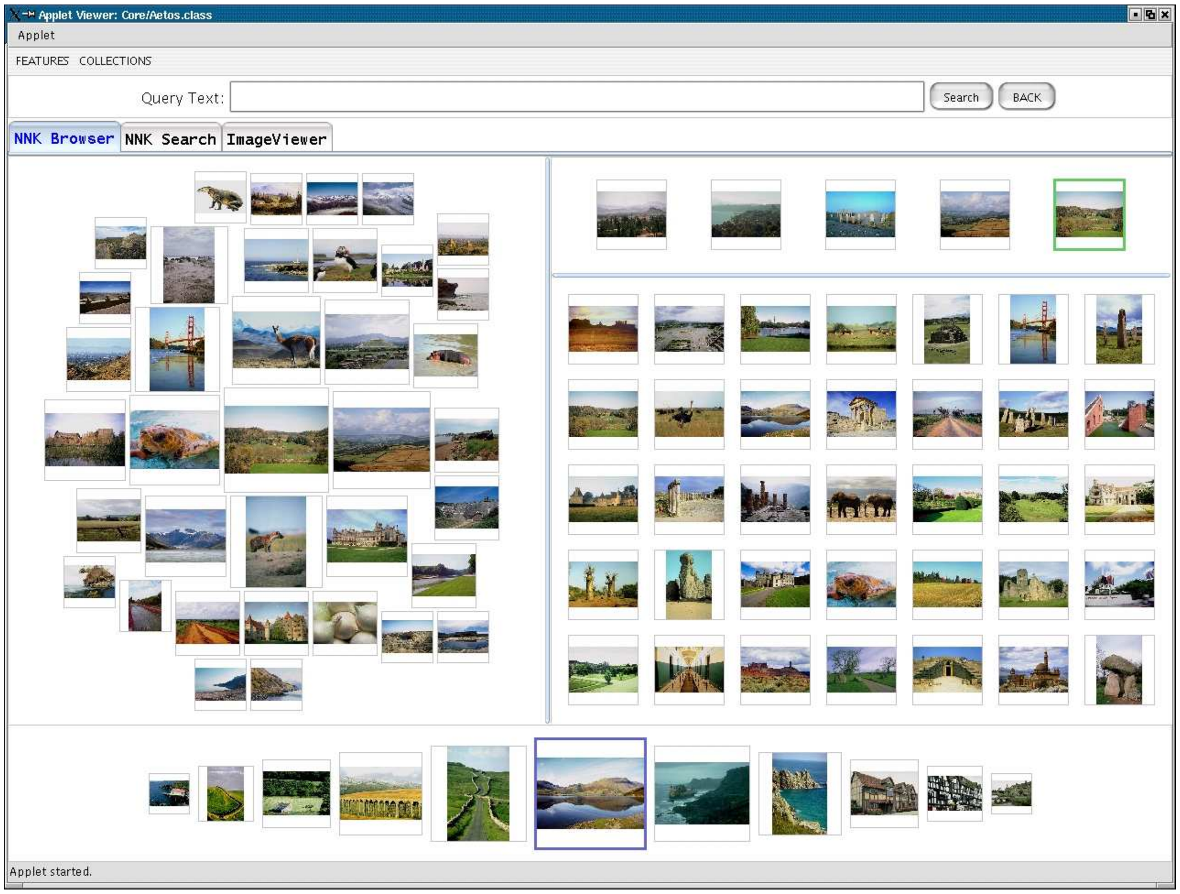


Figure 8.9: Screenshot of integrated system.

to perform online comparison of a query image with every image from the collection. Given a set of positive images, groupings are retrieved that exhibit the greatest overlap with the selected images.

8.5 Conclusions

Since the NN^k of an image can form a rather diverse set, images from a relevance class may form distributed subgraphs spread out across the network. These subgraphs are never presented to users in their entirety. Users only ever interact with a focal image and the set of its first-degree neighbours so the semantic structure can only be exploited indirectly by following relevance paths through the network. The goal of this chapter was to formulate ways to automatically extract this semantic structure and to make it available as another source of information guiding users during their interaction with NN^k Networks.

We identified a recently developed graph clustering algorithm as well-suited to our problem and applied it to the dual of one particular NN^k Network. By clustering the dual of a graph we allow any vertex to be part of more than one cluster. Our cluster validation method is based on the assumption that semantically coherent clusters should have a comparatively small joint vocabulary (the set of unique words present in the annotations that accompany the images of a cluster). We call this the description size of a cluster and make it our test statistic. For every cluster we compare its description size with the distribution of description sizes expected under the null hypothesis that the cluster has been assembled randomly. With over 60% of the clusters scoring significant P-values our formal validation confirms the positive impression we gain from manually inspecting individual clusters.

In cases where no annotations are available to check cluster quality, it might be desirable to have alternative methods to weed out the few clusters of low quality. This motivated our conjecture that semantically coherent clusters may be composed of edges associated with similar weight centroids. If borne out by the data, weight centroid homogeneity could be taken as a surrogate measure of cluster quality. Our analysis of the 2,399 clusters from the Corel collection rules out such a conclusion and suggests instead that the distributions in feature space of images belonging to the same relevance class are generally too complex to be captured by single feature combinations. This observation highlights the importance of employing a multitude of different metrics for image search and also brings out clearly a major limitation of any relevance feedback method that seeks to establish an optimal global metric for a particular relevance class. By computing the neighbourhood of an image in feature space according to a number of metrics, NN^k Networks are shown to be more flexible and can represent as part of their structure even the most contorted distribution.

The image clusters can serve a number of functions, two of which are outlined in this chapter. Both require clusters to be represented concisely in the form of cluster centroids. We motivate two ways how such a centroid may be defined. One is based on the notion of the topological distance between vertices of subgraphs, the other involves finding the stationary distribution of a Markov chain defined on the vertex set. The first application of the clustering is as an alternative form of accessing an image collection. Cluster centroids can replace image hubs as initial points of access when no query is formulated thus affording users with a potentially much better overview of the content of the collection than can be achieved through hubs. The second function is that of providing users with a summary view of the different classes the current focal image may belong to. When users select a new focal image we display not only its NN^k but also the centroids of all those clusters in which the focal image appears. Instead of gathering relevant images by following relevance paths through the network, users may frequently find it more profitable to select and expand appropriate cluster centroids. By revealing otherwise hidden semantic structure to the browsing user, NN^k clusters constitute a valuable addition to the existing NN^k framework.

Chapter 9

Conclusions and Future Work

9.1 Summary of results

The broad objective of this thesis was to formulate new ways of searching and browsing large image collections. The proposed framework is based on a simple but powerful idea which we have called the NN^k idea, where NN stands for nearest neighbour and k denotes the number of visual features. The conventional way of ranking images based on similarity involves computing a set of near neighbours under a fixed metric. We suggest instead to parametrise the metric and determine the nearest neighbour under all different parameter settings. This provides us with an eclectic set of images which we call the lateral neighbours or NN^k. The set of NN^k of an image are a distinguished set of neighbours in a k-dimensional vector space. They may be viewed as an exemplification of the various semantic facets of the image.

We have seen two major applications of the NN^k idea. The first application is to relevance feedback. The conventional approach to relevance feedback is to refine feature weights over a number of iterations based on positive and negative examples. NN^k search exploits the fact that each NN^k of a query image corresponds to a particular parameter setting, namely the feature weights for which that NN^k is more similar to the query than is any other image. The technique involves two steps: by first displaying the NN^k of the query image and letting the user select one or more relevant NN^k, we can in a second step retrieve using the weight sets associated with the selected images. The method turns out to be highly effective even for large collections of more than 30,000 images and is shown to outperform alternative relevance feedback methods.

The second application, which occupies the greater part of the thesis, can be viewed as a natural extension of NN^k search. It provides an effective solution in situations when the first retrieval result is unsatisfactory or when users have no clearly defined information need. Rather than retrieving with the weight set of a chosen image, we instead enable users to view the NN^k of that image. Because we can precompute the NN^k for every image in a collection, the interaction can then take the form of browsing a static, hyperlinked network in which every image in connected to all its NN^k.

We have investigated the suitability of the resulting NN^k Networks for content-based search from a number of different viewpoints. Investigation of their topological properties revealed the presence of small-world properties: even for large collections of 100,000 images vertices are on average not more than four links apart (like in random graphs) but the networks also exhibit a substantial degree of local structure (unlike random graphs). These findings suggest that users should be able to navigate quickly across the network whilst always having a sense of locality and neighbourhood. An analysis of the distribution of relevant images across the network showed that these tend to cluster in ways that should support directed search. Relevant images form compact subgraphs that allow users to satisfy their information needs by following relevance paths. These analyses were complemented by a large-scale quantitative evaluation on 32,000 images and a set of realistic search tasks. The study provides strong evidence that even for difficult collections and tough queries, interactive search across NN^k Networks provides a very fast and effective alternative to automated content-based search.

By applying the MCL algorithm for graph clustering to the dual of NN^k Networks, we demonstrate how some of the semantic structure can be captured automatically and made available to users during the interaction.

In addition to providing an evaluation of the new interaction scheme, we highlight and solve a number of questions of practical relevance: how can we update the network efficiently in dynamic environments, how can we ensure that the networks are strongly connected and how do we access the network in query-free environments?

The thesis has been concerned with the development and the evaluation of a new framework for image search and browsing and much what has been covered is, by necessity, groundwork. There are two strands along which the work can be taken further: it can be applied to other domains and it can be extended and refined in numerous ways. Its application to new domains should be an exciting way to characterise better the strengths and weaknesses of the proposed framework. We expect there to be a fertile interplay between the two strands as new applications will come with their own requirements and any refinements to the core framework may open up application areas not previously thought of. We will in the following sketch a few suggestions.

9.2 Application to other domains

NN^k Networks have been developed primarily for interactive image retrieval but the same framework can be applied to other types of media as well as multimedia. The networks provide a well-motivated data structure for any objects that can be given a vectorial representation and for which it is not always obvious how individual features are to be combined. We shall here consider its applicability to music and document retrieval.

9.2.1 Music retrieval

With the dramatic increase in the amount of digitised music available on the Web both in the public domain and on commercial sites, content-based music retrieval is gradually gaining greater prominence. Text search requires music pieces to have been annotated but even if they are, annotation is typically restricted to composer and genre and may thus prove insufficient for finding pieces that are musically similar. Content-based retrieval can work from the raw audio (Cano et al., 2005) but is most frequently applied to more structured formats such as MIDI (e.g. Doraisamy and Rüger, 2004). For monophonic music it is straightforward to find a robust representation in terms of low-level features related to pitch, harmony and rhythm. For polyphonic music this is somewhat more difficult (Pickens, 2001), and certainly much harder than for retrieval of text documents where words tend not to overlap.

Automated search for music pieces based on their content is therefore still fraught with difficulties for general collections. The greater fault-tolerance of browsing over automated search should render NN^k Networks a valuable complement if not alternative.

The transfer of NN^k Networks from images to music is not, however, without difficulties. Although Cunningham (2005) concludes from an extensive study of user interaction with music archives that "the most significant lesson learned from this exercise is that people do a lot of browsing with music." and observes that "browsing has not been well supported in MIR systems, relative to searching", browsing typically takes place within classification hierarchies that are established manually and populated on the basis of metadata. Browsing takes place by following textual cues. Purely auditory browsing is more problematic as has been noted elsewhere (Blandford and Stelmaszewska, 2002). Unlike images, music pieces have a temporal extension and need to be listened to in real-time before users can judge their relevance. Besides, downloading music files can be comparatively time-consuming. In many cases, however, some metadata is available and can be used to label NN^k so as to facilitate pre-selection. As mentioned above, the presence of metadata does not at all diminish the value of content-based organisation of music. The two are rather complementary: On the one hand, metadata is generally poor and can capture music content to an even lesser degree than image captions. Composer and genre information on their own provide groupings that are not always desirable or sufficient from a musicological perspective. Content-based organisation helps to unveil relationships on the content level between pieces that outwardly have little in common. On the other hand, content-based browsing benefits from visual cues which can be provided through metadata by labelling pieces accordingly. One should also note that metadata can be incorporated not only as a means to label NN^k but also as another feature during network construction.

Semi-structured music files are much more similar to text documents than images: they are essentially linear and allow easier access to semantic primitives such as harmonies, pitch and rhythm. This provides an opportunity to enrich NN^k Networks by labelling NN^k not simply by metadata, if present, but also with the musical feature that is assigned greatest weight for that particular NN^k. This makes less sense in content-based image browsing not only because image features tend to be more obscure but also, of course, because the content of an image is best obtained by visual inspection.

Although attempts have been made to cluster music files without feature extraction (Cilibrasi and Vitányi, 2005), it has been observed that the clustering is incapable of modelling the fact that music pieces, like images, may be categorised in different ways. Pieces by J. S. Bach may rightfully be grouped together with those by Gershwin for reasons of harmonies and with Chopin on the basis of their melodic structure. Dual clustering of NN^k Networks provides an interesting solution to recover these multiple relationships. NN^k Networks should not only appeal to the ordinary consumer but also open up exciting opportunities for scholarly research.

9.2.2 Document retrieval

Information retrieval research was originally concerned with text documents to the extent that text retrieval was, and still is, synonymous with information retrieval. The features traditionally used for text retrieval are individual terms: documents are represented as term vectors in which each component represents some information about the occurrence of the respective term. The most common term statistics is the TF-IDF weight (Salton and McGill, 1982). TF is simply the frequency of a term in the document under consideration. IDF is the inverse of the number of documents in which the term occurs and is typically taken to the logarithm to render it less sensitive to the collection size. The number of features, therefore, is larger by many magnitudes than in typical image and music retrieval applications. Another difference of some significance relates to the fact that queries take the form of a set of terms. Unlike in image and music retrieval, users thus specify explicitly which features they deem important for similarity computation. Determining feature relevance is therefore less prominent a problem in automated text retrieval but it becomes an issue when users do not specify terms, as would be the case in browsing applications. Documents also display a degree of polysemy not only because they may treat a number of different topics but also because each topic can be related to many others. An article on the Italian renaissance relates to articles covering the spread of renaissance ideas throughout the rest of Europe but also to articles that portray historical figures of the time. Clearly, terms will need to be assigned different degrees of relevance in order to distil these different facets. With a few million terms for moderately sized corpora, the k in NN^k is too large to make network construction even remotely practical.

An exciting solution is latent semantic indexing, or LSI (Deerwester et al., 1990; Landauer and Dumais, 1997). LSI seeks to extract from text not just term occurrences but semantic structure thereby dramatically reducing the number of dimensions. LSI approximates the original T-dimensional term space by the first k principal component directions in this space using the $N \times T$ document-term matrix to estimate the directions. This principal components approach exploits redundancy in the terms: terms that regularly co-occur in documents may be merged and replaced by a new surrogate feature. The new k-dimensional representation of the documents is much more amenable to NN^k Network computation.

Documents share with music pieces that it takes time to judge their relevance but documents have the advantage that they may be summarised more meaningfully. The possibility of concise representations makes browsing a valuable complement to automated search and it is already used as such. Citeseer, for example, provides links to related documents based on sentence similarity and co-citations thus allowing researchers to navigate a complex network structure with different types of links. Google provides for each retrieved document a link to related documents. Unlike NN^k Networks, however, these navigation structures are not versatile enough to extract different semantic facets of the documents because the similarity metric is fixed.

Another application in which it is not always clear how to assign weights to different features is to structured documents. The Extensible Markup Language (XML) has by now assumed such a ubiquity on the Web and in many other computing applications that XML retrieval has become a topic of great interest. An important issue pertaining to XML retrieval is the presence of two distinct dimensions along which relevance may be judged, namely content and structure. Content can be captured using traditional techniques from flat document information retrieval while structure fits well into a database framework. This poses not only the problem of how to build efficient index structures that combine structure and content information (Weigel et al., 2004, e.g.) but also how to allow for inexact matches (e.g. Lalmas and Rölleke, 2004) of content-and-structure queries (CAS queries). If both content and structure can be assigned continuous similarity values, then NN^k Networks could provide a way to build navigable structures that are unbiased with respect to either feature. This should be particularly valuable as users may often only have a vague idea how to weigh the relative importances of structure and content.

9.3 Extensions and ramifications

9.3.1 Synergy effects with text search

We have envisaged NN^k Networks as static structure that encompass entire collections and whose principal application is to query-free image search. We mentioned in Chapter 7 that they can easily be integrated into conventional image search systems by using the results of a visual search as entry points into the network. We observed from our user experiments in 7 that for difficult queries it may sometimes take some imagination to find the first relevant image but that more relevant images are then found quite readily. Often, the converse holds for text-based image as supported, for example, by Google. The inconsistency of the annotation results in comparatively low precision but there are often at least some images that are relevant. It is inordinately difficult, however, to then accumulate more of the same kind. These observations suggest that a powerful synergy between text-based search and NN^k Networks should be possible. The former provides excellent starting points which may be difficult to obtain through automated content-based search. The latter allows effective exploration of the neighbourhood around the set of relevant images. For this to work, the image collection would not even need to be fully annotated.

The conventional presentation of search results is that of a ranked list distributed across a number of hyperlinked sites. Alternatives include clustering of the search results (Hearst and Pedersen, 1996; Zamir and Etzioni, 1998; Leuski and Allan, 2000; Carey et al., 2003). If a collection has been annotated exhaustively and reliably so that text search becomes an option, NN^k Networks can be employed to organise search results. Rather than browsing the NN^k Network that corresponds to the entire collection, users may instead navigate across a much smaller network that contains only the top n images from the automated search. This could be based, for example, on the topological distance between retrieved images in the original network.

9.3.2 Image annotation by label propagation

Automated image annotation provides a different route towards solving the problem of content-based image retrieval. By learning associations between visual features and labels on a training set, labels may be predicted on unseen images, see for example the work by Yavlinsky et al. (2005) and references therein. Once a collection has thus been annotated, conventional keyword search can be carried out. The approach may appear circuitous but has the advantage of users no longer having to specify a query pictorially. Moreover, once labels are available, images can be organised according to known semantic relationships between terms. If the distribution of terms and visual features is flexible enough, we can discover semantic relationships between visually dissimilar images.

NN^k Networks provide a concise representation of the visual similarities and semantic relationships between images. If a collection is partially annotated, the graphs provide natural structures to propagate annotation terms to unlabelled images. Such an approach would be model-free in the sense that it does not assume any particular dependency between features and terms and it thus circumvents the need to learn a model which might not always be possible in the absence of a good training set. As such it is similar to early work on automated annotation where labels were propagated based on textural similarity (Picard and Minka, 1995) but also to recent work in functional genomics where graph structures are increasingly used to represent the functional relationships between genes. In a typical functional-linkage graph, for example, each vertex corresponds to a protein, and an edge connects two proteins if some experimental or computational procedure suggests that these proteins might share the same function (Yanai et al., 2002; Marcotte et al., 1999). Zhou et al. (2002) and Karaoz et al. (2004) suggest ways to propagate evidence systematically across the entire graph to provide functional annotations for new proteins.

9.3.3 Variations on the NN^k theme

Beyond linear convex combinations

We defined an image to be an NN^k if it is closest to another under some convex combination of k feature-specific distances. If d denotes the k-dimensional distance vector of an image and D be the set of all distance vectors for a collection, we found in Chapter 3 that the linear combination of distances in the form of a dot product $w \cdot d$ has the consequence that the NN^k are images whose d are among the extreme points of the convex hull of D. If we relax the linearity constraint and allow aggregation functions which are non-linear in the distances, i.e. $f(d) = \sum_{i=1}^{k} w_i(d_i)^\alpha$, $\alpha > 0$, we obtain a larger set of neighbours. In fact, the solution for general α is the solution of the maximum vector problem (Preparata and Shamos, 1985) which contains all those d for which there does not exist another point that is closer along every dimension. The solution set contain the NN^k as a subset and can be quite large in particular if features are not positively correlated (Börzsöny et al., 2000).

In all our experiments we noticed that the number of NN^k is in pleasant and almost striking agreement with the number of images which users may be able to interact with at any one time. They also seem to exhibit a degree of variability that proves adequate for both directed search and undirected exploration. For applications with a small number of different features (such as content and structure in XML retrieval) the more comprehensive non-linear aggregation method might be of interest. An appealing feature of this approach is that the NN^k can be found explicitly by solving the maximum vector problem rather than by scanning of the parameter space. The most efficient algorithms to achieve this have $O(N \log N)$ for $k = 2$ and $O(N(\log N)^{k-2})$ for $k \geq 3$ (Börzsöny et al., 2000).

Region-based NN^k

The visual features that we employ for NN^k Network construction are computed either for the entire image or for each of a set of tiles that partition the image. The spatial information extracted through the tiling renders the image representation non-invariant with respect to rotation and reflection, a feature that may often be desired as spatial relationships between image parts can help distinguish between different scenes. But because the final feature vector for a tiled image is a concatenation of the individual feature vectors computed for each tile, the representation does not lend itself well for detecting similarity between regions. Images that share similar objects in different environments cannot readily be detected even if the objects occupied the same position in the image because any local similarity is diluted by the dissimilarity in the rest of the image.

We could greatly enrich the network structure by also computing features for a set of image regions as in Carson et al. (1999) and Ma and Manjunath (1999). This would allow us to find the set of NN^k for each *region* of an image and offers a potentially more powerful selection strategy (but also additional computational overhead during network construction).

9.3.4 Linearised networks for large collections

NN^k Network construction as proposed in Chapter 5 has a time complexity that is quadratic in the number of images and linear in the number and the average length of the image features. The quadratic dependency results from the need to compute pair-wise distances for every feature. As of September 2005, Google provides access to $2,187,212,422$ images. Turning the image content of the web into a NN^k Network would certainly be a major challenge given our experience on a collection of 100,000 images.

There are a number of remedies that can be combined to alleviate the problem. One solution is to use compact features of small size rather than sparse histogram features. Often visual features consist of several hundreds of bins only a minority of which are non-zero. Another solution, to which the problem lends itself very well, is parallelisation. Both should improve run times substantially without compromising accuracy. For very large collections, approximations seem inevitable. As mentioned in Chapter 5, reducing the complexity of k-nearest neighbour search is one promising approach. Constant time k-nearest neighbour search would reduce the complexity of network construction from $O(N^2)$ to $O(N)$. In addition, we propose to linearise NN^k Networks as follows: instead of populating the entire $N \times N$ distance matrix for every feature, we can tile the matrix along its diagonal into overlapping, quadratic submatrices, subsequently compute the pair-wise distances for the submatrices only and finally construct the network on the basis of the computed distances. This corresponds to dividing the collection into overlapping image sets. Because of the partial overlap images from the intersection may connect to other images from adjacent sets thus helping to keep the resulting network connected. The final network can be viewed as a interlinked ensemble of individual networks. To increase the degree of interconnectedness between adjacent image sets, we would want to keep each image set rather diverse. An example of a distance matrix divided into overlapping submatrices is shown in Figure 9.1.

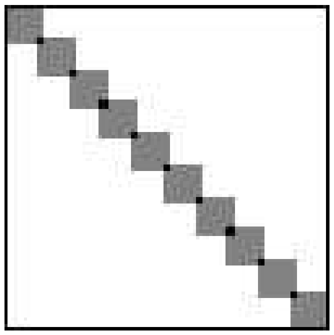


Figure 9.1: Linearisation of distance matrix.

Let us consider the computational savings. Consider a collection of N images divided into overlapping subsets of M images with an overlap between adjacent image sets of $0 < P < M/2$ (so that an image occurs in at most two sets). Assuming for simplicity that the first and the last set in the collection do not overlap with each other, the number of pairwise computations is given by

$$T(N, M, P) = (N - P)(M + P) + P^2,$$

which has time complexity $O(NM)$ rather than $O(N^2)$. It needs to be investigated to what extent this linearisation affects network topology and navigability but if users are not in pursuit of that one image but a class of images, the restriction may be quite acceptable.

Glossary

Affine combination: A vector y is an affine combination of vectors $x^1, x^2, ..., x^m$ if $y = \sum_{i=1}^{m} a_i x^i$ and $\sum_{i=1}^{m} a_i = 1$

Affinely independent: A set S is called affinely independent if no point in S is an affine combination of other points from S.

Convex combination: An affine combination of vectors with the additional constraint that all coefficients are non-negative.

Convex hull: The convex hull of a set S is the set of all convex combinations of the points in S. For planar objects the convex hull has the shape an elastic rubber band would assume when stretched around the object.

Dynamic partial distance functions: A non-metric distance function that compares two k-dimensional vectors x and y by considering only the m components $(0 < m \leq k)$ with smallest Manhattan distance, i.e. $|x_i - y_i|$. The m component-wise distances are aggregated in a general Minkowski metric.

Focal image: In the context of this thesis, we refer to a focal image as the image whose NN^k we compute.

Geodesic: A geodesic between two points P and Q is the curve between P and Q that has minimum length. The length depends on the metric of the space. In the plane, the geodesics are straight lines. On the sphere, the geodesics are great circles.

Hyperplane: A hyperplane is a higher-dimensional analogue of a two-dimensional plane in three-dimensional space. In an n-dimensional vector space, a hyperplane is a linear subspace of $(n-1)$ dimensions.

Iso-NN^k region: The set of all weight sets for which a particular image is the NN^k of some focal image is referred to as the iso-NN^k region of that NN^k. It is a convex region within the k- dimensional weight space.

Linear combination: A vector y is a linear combination of vectors $x^1, x^2, ..., x^m$ if $y = \sum_{i=1}^{m} a_i x^i$, $a_i \in \mathbb{R}$.

Mahalanobis distance: The Mahalanobis distance is a distance measure introduced by P Mahalanobis in 1936 to determine the degree of similarity between multi-variate vectors. It differs from

Euclidean distance by taking into account the correlations between vectors components. Given two n-dimensional vectors x and y, the Mahalanobis distance is

$$d(x, y) = \sqrt{(x - y)^T \Sigma^{-1} (x - y)}.$$

where Σ is the covariance matrix for the distribution from which x and y are assumed to come from.

Manifold: A manifold is a topological space that is locally Euclidean, i.e. around every point, there is a neighborhood that is topologically the same as the open unit ball in $\mathbb{R}^n$. Intuitively, any object that is nearly "flat" on small scales is a manifold.

Metric: A non-negative function $g(x, y)$ describing the distance between neighbouring points for a given set. A metric needs to satisfy:

1. $g(x, y) = g(y, x)$
2. $g(x, y) = 0$ iff $x = y$
3. $g(x, y) + g(y, z) \geq g(x, z)$ (Triangle inequality)

Minkowski metric: The Minkowski or L_α metric is given by

$$D(x, y) = \left[\sum_i |x_i - y_i|^\alpha \right]^{\frac{1}{\alpha}}, \quad \alpha \geq 1.$$

This reduces to the Euclidean metric for $\alpha = 2$ and to the L_1 metric for $\alpha = 1$. The case $\alpha \to \infty$ yields the L_∞ metric, $\max_i |x_i - y_i|$.

Multi-modality: This term is used in two different senses. Firstly, it denotes the presence of multiple *modes* in a distribution, such as the distribution of image features in feature space. Secondly, it is used when referring to the integration of different data types or types of interaction (generally *modalities*). For example, multi-modal image search commonly implies the joint utilisation of visual features and textual annotation for retrieval. Multi-modal interfaces allow for different input modalities, e.g. speech and text.

NN^k: An image that is closest to some other image (the focal image) under some instantiation of a parametrised metric. NN stands for nearest neighbour and k denotes the number of parameters, or weights. This thesis considers the NN^k when the distance function is a convex combination of feature-specific distances,

$$D = \sum_{i=1}^{k} w_i d_i, \quad \sum_{i=1}^{k} w_i = 1, \quad w_i \geq 0$$

where d_i is an arbitrary distance metric defined for the ith feature. Because all weights are non-negative, this combination is convex.

Polytope: The word polytope is used to mean a number of related, but slightly different mathematical objects. We use it to describe a finite region of n-dimensional space enclosed by a finite number of hyperplanes and thus as the general term of the sequence "point, line segment, polygon, polyhedron...".

Simplex: A k-simplex is the convex hull of a set of $k + 1$ affinely independent points in $\mathbb{R}^n$. We say k is the dimension of the simplex. In one dimension, the simplex is the line segment $[-1, 1]$. In two dimensions, the simplex is the convex hull of the equilateral triangle. In three dimensions, the simplex is the convex hull of the tetrahedron. The simplex is so-named because it represents the simplest possible polytope in any given space. It turns out that the weight space considered in this thesis forms a simplex and we refer to it as the weight simplex. The weight simplex for k features is $(k-1)$-dimensional.

Skyline operator: The Skyline, or Pareto operator selects those tuples in relational databases that are not dominated by any others, that is, no other tuple is better over each of the designated attributes.

Support: The support of an NN^k with respect to some focal image is the relative size of its iso-NN^k region. We approximate the support as the proportion of weight sets for which that image is the NN^k.

***t*-test:** A t-test is any statistical hypothesis test in which the test statistic has a Student's t distribution if the null hypothesis is true. It can be used to test the null hypothesis that the means of two normally distributed random variables are equal based on two finite samples.

χ^2 **distribution:** The χ^2 distribution is a probability distribution frequently used in statistical significance tests. It is useful because, under reasonable assumptions, easily calculated quantities can be proved to follow distributions that are well approximated by the χ^2 distribution if the null hypothesis is true.

It is commonly used to test the fit between a theoretical frequency distribution and a frequency distribution of observed data for which each observation may fall into one of n classes. In particular, the test statistic is

$$\chi^2 = \sum_{i=1}^{n} \frac{(O_i - E_i)^2}{E_i},$$

where the sum is over all n classes. The deviation is considered significant if the χ^2 statistic is greater than some critical value (which depends on n and the level of confidence at which we would like to be able to reject the null hypothesis).

Bibliography

Aggarwal, C. and P. Yu (2000). The IGrid index: reversing the dimensionality curse for similarity indexing in high dimensional space. In *Proc Knowledge Discovery and Data Mining*, pp. 119–129.

Aggarwal, G., T. Ashwin, and S. Ghosal (2002). An image retrieval system with automatic query modification. *IEEE Trans Multimedia 4*(2), 201–213.

Aho, A., J. Hopcroft, and J. Ullman (1983). *Data Structures and Algorithms.* Addison-Wesley.

Albert, R. and A.-L. Barabási (2002). Statistical mechanics of complex networks. *Rev Mod Phys 74*, 47–97.

Albert, R., H. Jeong, and A. Barabási (1999). Diameter of the world wide web. *Nature 401*, 130–131.

Alpert, C., A. Kahng, and S. Yao (1999). Spectral partitioning: The more eigenvectors, the better. *Discrete Applied Mathematics 90*, 3–26.

Anderberg, M. (1973). *Cluster Analysis for Applications.* New York: Academic Press.

Andrews, G. (2004). *The Theory of Partitions.* Cambridge University Press.

Aslam, J. and M. Montague (2001). Models for metasearch. In *Proc Int'l ACM SIGIR Conf on Research and Development in Information Retrieval*, pp. 276–284.

Bang, H. and T. Chen (2002). Feature space warping: An approach to relevance feedback. In *Proc IEEE Int'l Conf Image Processing.*

Barabási, A. (2004). *Linked - The New Science of Networks.* Perseus Publishing.

Barabási, A., R. Albert, and H. Jeong (2003). Scale-free characteristics of random networks: the topology of the world-wide web. *Physica A 281*, 69–77.

Barnard, K. and D. Forsyth (2001). Learning the semantics of words and pictures. In *Proc IEEE Int'l Conf Computer Vision*, Volume 2, pp. 408–415.

Belkin, M. and P. Niyogi (2002). Laplacian eigenmaps and spectral techniques for embedding and clustering. In *Proc Conf Neural Information Processing Systems.*

Belkin, M. and P. Niyogi (2003). Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Computation 15*, 1373–1396.

Benitez, A. and S.-F. Chang (2003). Multiresolution organization and browsing of images using multimedia knowledge networks. Technical report, Columbia University, US.

Benson, R. (1966). *Euclidean Geometry and Convexity.* McGraw-Hill.

Blandford, A. and H. Stelmaszewska (2002). Usability of musical digital libraries: a multimodal analysis. In *Proc Int'l Symp Musical Digital Libraries*, pp. 231–237.

Böhm, M. and P. Mayhew (2005). Historical biogeography and the evolution of the latitudinal gradient of species richness in the Papionini. *Biological Journal of the Linnean Society 85*, 235–246.

Bollobás, B. (1985). *Random Graphs*. Springer.

Bornholdt, S. and H. G. Schuster (2003). *Handbook of Graphs and Networks - From the Genome to the Internet*. Wiley-VCH.

Börzsöny, S., D. Kossmann, and K. Stocker (2000). The skyline operator. In *Proc Int'l Conf Data Engineering*, pp. 235–255.

Broder, A., R. Kumar, F. Maghoul, P. Raghavan, S. Rajagopalan, R. Stata, A. Tomkins, and J. Wiener (2000). Graph structure in the web. *Computer Networks 33*(1-6), 309–320.

Campbell, I. (2000). *The ostensive model of developing information-needs*. Ph. D. thesis, University of Glasgow.

Cano, P., M. Koppenberger, and N. Wack (2005). An industrial-strength content-based music recommendation system. In *Proc Special Interest Group Information Retrieval*, pp. 673.

Carey, M., D. Heesch, and S. Rüger (2003). Info Navigator: A visualization tool for document searching and browsing. In *Proc Int'l Conf Distributed Multimedia Systems*, pp. 23–28.

Carson, C., M. Thomas, S. Belongie, J. Hellerstein, and J. Malik (1999). Blobworld: a system for region-based image indexing and retrieval. In *Proc Int'l Conf Visual Information Systems*, pp. 509–516.

Cha, G.-H. (2003). Bitmap indexing method for complex similarity queries with relevance feedback. In *Proc ACM Int'l Workshop Multimedia Databases*, pp. 55–62.

Chang, Y., C. Cirillo, and J. Razon (1971). Evaluation of feedback retrieval using modified freezing, residual collection and test and control groups. In *The SMART Retrieval System – Experiments in Automatic Document Processing (ed. G Salton)*, pp. 355–370.

Chen, C., G. Gagaudakis, and P. Rosin (2000). Similarity-based image browsing. In *Proc Int'l Conf Intelligent Information Processing*, pp. 206–213.

Chen, J., C. Bouman, and J. Dalton (1998). Similarity pyramids for browsing and organization of large image databases. In *Proc SPIE Conf Human Vision and Electronic Imaging III*, Volume 3299, pp. 563–575.

Chen, J., C. Bouman, and J. Dalton (2000). Hierarchical browsing and search of large image databases. *IEEE Trans Image Processing 9*(3), 442–455.

Chen, Y., X. Zhou, and T. Huang (2001). One-class SVM for learning in image retrieval. In *Proc IEEE Int'l Conf Image Processing*, pp. 34–37.

Cheung, S. S. and A. Zakhor (2005). Fast similarity search and clustering of video sequences on the world-wide-web. *IEEE Trans Multimedia 7*(3), 524–537.

Chung, F. (1997). *Spectral Graph Theory.* CBMS Regional Conference Series in Mathematics. American Mathematical Society.

Cilibrasi, R. and P. Vitányi (2005). Clustering by compression. *IEEE Trans Information Theory 51*(4), 1523–1545.

Clough, P., H. Müller, and M. Sanderson (2005). The CLEF cross language image retrieval track 2004. In *Proc Workshop Cross-Language Evaluation Forum*, pp. 459–474. LNCS 3491, Springer.

Cohn, D., L. Atlas, and R. Ladner (1994). Improving generalization with active learning. *Machine Learning 15*(2), 201–221.

Cox, I., M. Miller, T. Minka, T. Papathomas, and P. Yianilos (2000). The Bayesian image retrieval system PicHunter: theory, implementation, and psycho-physical experiments. *IEEE Trans Image Processing 9*(1), 20–38.

Cox, I., M. Miller, S. Omohundro, and P. Yianilos (1998). An optimized interaction strategy for Bayesian relevance feedback. In *Proc IEEE Conf Computer Vision and Pattern Recognition*, pp. 553–558.

Cox, K. (1992). Information retrieval by browsing. In *Proc Int'l Conf New Information Technology.*

Cox, K. (1995). *Searching through browsing.* Ph. D. thesis, University of Canberra.

Croft, B. and T. Parenty (1985). Comparison of a network structure and a database system used for document retrieval. *Information Systems 10*, 377–390.

Crucianu, M., M. Ferecatu, and N. Boujemaa (2004). Relevance feedback for image retrieval: a short review. In *State of the Art in Audiovisual Content-Based Retrieval, Information Universal Access and Interaction including Datamodels and Languages (DELOS2 Report).*

Cunningham, S. (2005). People and their music. In *Workshop Joint Conference Digital Libraries.*

Cuthill, E. and J. McKee (1969). Reducing the bandwidth of sparse symmetric matrices. In *Proc ACM Conf Assoc Computing Machinery*, pp. 157–172.

Darwin, C. (1859). *The Origin of Species by Means of Natural Selection.* John Murray, London.

de Vries, A., N. Mamoulis, N. Nes, and M. Kersten (2002). Efficient k-nn search on vertically decomposed data. In *Proc ACM SIGMOD Int'l Conf on Management of Data*, pp. 322–333. ACM Press.

Deerwester, S., S. Dumais, G. Furnas, T. Landauer, and R. Harshman (1990). Indexing by latent semantic analysis. *Journal of the American Society for Information Science 41*, 391–407.

Dom, B. (2001). An information theoretic external cluster-validity measure. In *Research Report, IBM T. J. Watson Research Center RJ 10219.*

Doraisamy, S. and S. Rüger (2004). A polyphonic retrieval system using n-grams. In *Proc Int'l Conf Music Information Retrieval*, pp. 204–209.

Dorogovtsev, S. and J. Mendes (2003). *Evolution of Networks: From Biological Nets to the Internet and the WWW.* Oxford University Press.

Dorow, B. and D. Widdows (2003). Discovering corpus-specific word-senses. In *Proc Conf European Chapter Association for Computational Linguistics*, pp. 79–82.

Dorow, B., D. Widdows, K. Ling, J.-P. Eckmann, D. Sergi, and E. Moses (2004). Using curvature and markov clustering in graphs for lexical acquisition and word sense discrimination. In *Workshop MEANING-2005*.

Duda, R., P. Hart, and D. Stork (2001). *Pattern Recognition.* Wiley.

Enright, A., V. Kunin, and C. Ouzounis (2003). Protein families and tribes in genome sequence space. *Nucleic Acids Research 31*(15), 4632–4638.

Enright, A., S. van Dongen, and C. Ouzounis (2002). An efficient algorithm for large-scale detection of protein families. *Nucleic Acids Research 30*(7), 1575–1584.

Erdös, P. and A. Rényi (1960). On the evolution of random graphs. *A Magyar Tudományos Akadémia Matematikai Kutató Intézetének Közleményei 5*, 17–61.

Fagin, R., R. Kumar, and D. Sivakumar (2003). Efficient similarity search and classification via rank aggregation. In *Proc ACM Int'l Conf Management of Data*, pp. 301–312.

Feng, S., R. Manmatha, and V. Lavrenko (2004). Multiple Bernoulli relevance models for image and video annotation. In *Proc Int'l Conf Computer Vision and Pattern Recognition*, pp. 1002–1009.

Fowler, R., B. Wilson, and W. Fowler (1992). Information Navigator: An information system using associative networks for display and retrieval. Technical report, Department of Computer Science, University of Texas, No. 92-1.

Fowlkes, E. and C. Mallows (1983). A method for comparing two hierarchical clusterings. *Journal of American Statistical Association 78*, 553–569.

Fox, E. and J. Shaw (1994). Combination of multiple searches. In *Text Retrieval Conference, Gaithersburg, MD*, pp. 243–249.

George, A. and J. Liu (1981). *Computer Solution of Large Sparse Positive Definite Systems.* Prentice-Hall.

Gersho, A. and R. Gray (1992). *Vector Quantisation and Signal Processing.* Boston: Kluwer Academic Publisher.

Goldberger, J., S. Roweis, G. Hinton, and R. Salakhutdinov (2005). Neighbourhood component analysis. In *Proc Conf Neural Information Processing Systems.*

Guttman, A. (1984). R-trees: a dynamical index structure for spatial searching. In *Proc ACM SIGMOD Intl Conf Management of Data*, pp. 47–54.

Hawking, D., E. Voorhees, N. Craswell, and P. Bailey (1999). Overview of the TREC-8 web track. In E. Voorhees and D. Harman (Eds.), *Proc Text Retrieval Conference, NIST Special Publication 500-246*, pp. 131–150.

He, J., M. Li, H.-J. Zhang, H. Tong, and C. Zhang (2004). Mean version space: a new active learning method for content-based image retrieval. In *Proc Multimedia Information Retrieval*, pp. 15–22.

He, X., W.-Y. Ma, and H.-J. Zhang (2004). Learning an image manifold for retrieval. In *Proc Multimedia Information Retrieval*, pp. 17–23.

Hearst, M. and J. Pedersen (1996). Reexamining the cluster hypothesis: Scatter/gather on retrieval results. In *Proc Special Interest Group Information Retrieval*, pp. 76–84.

Heesch, D., M. Pickering, A. Yavlinsky, and S. Rüger (2004). Video retrieval within a browsing framework using keyframes. In *Proc TREC Video 2003*.

Heesch, D. and S. Rüger (2002). Combining features for content-based sketch retrieval — a comparative evaluation of retrieval performance. In *Proc Europ Colloquium Information Retrieval Research*, pp. 41–52. LNCS 2291, Springer.

Heesch, D. and S. Rüger (2003). Performance boosting with three mouse clicks – relevance feedback for CBIR. In *Proc European Conf Information Retrieval*, pp. 363–376. LNCS 2633, Springer.

Heesch, D. and S. Rüger (2004). Three interfaces for content-based access to image collections. In *Proc Int'l Conf Image and Video Retrieval*, pp. 491–499. LNCS 3115, Springer.

Heesch, D. and S. Rüger (2005). Image browsing: A semantic analysis of NN^k networks. In *Proc Int'l Conf Image and Video Retrieval*, pp. 609–618. LNCS 3568, Springer.

Heesch, D., A. Yavlinsky, and S. Rüger (2003). Performance comparison between different similarity models for CBIR with relevance feedback. In *Proc Int'l Conf Image and Video Retrieval*, pp. 456–466. LNCS 2728, Springer.

Helman, D. (1988). *Analogical Reasoning*. Kluwer.

Hochberg, Y. and A. Tamhane (1987). *Multiple Comparison Procedures*. Wiley.

Hong, P., Q. Tian, and T. Huang (2000). Incorporate support vector machines to content-based image retrieval with relevant feedback. In *Proc IEEE Int'l Conf Image Processing*, pp. 750–753.

Howarth, P. and S. Rüger (2004). Evaluation of texture features for content-based image retrieval. In *Proc Int'l Conf Image and Video Retrieval*, pp. 326–334. LNCS 3115, Springer.

Howarth, P. and S. Rüger (2005). Trading precision for speed: localised similarity functions. In *Proc Int'l Conf Image and Video Retrieval*, pp. 415–424. LNCS 3568, Springer.

Ishikawa, Y., R. Subramanya, and C. Faloutsos (1998). Mindreader: querying databases through multiple examples. In *Proc Conf Very Large Databases*, pp. 433–438.

Jain, A. and R. Dubes (1988). *Algorithms for Clustering Data*. Englewood Cliffs, Prentice-Hall.

Jing, F., M. Li, L. Zhang, H. Zhang, and B. Zhang (2003). Learning in region-based image retrieval. In *Proc Int'l Conf Image and Video Retrieval*, pp. 198–207. LNCS 2728, Springer.

Kanungo, T., B. Dom, W. Niblack, and D. Steele (1994). A fast algorithm for MDL-based multi-band image segmentation. In *Proc IEEE Computer Vision and Pattern Recognition*, pp. 9 – 16.

Karaoz, U., T. Murali, S. Letovsky, Y. Zheng, C. Ding, C. Cantor, and S. Kasif (2004). Whole-genome annotation by using evidence integration in functional-linkage networks. *Proceedings of the National Academy of Sciences 101*(9), 2888–2893.

Kaufman, L. and P. Rousseeuw (1990). *Finding Groups in Data: An Introduction to Cluster Analysis.* New York: Wiley.

Kim, D.-H. and C.-W. Chung (2003). Qcluster: relevance feedback using adaptive clustering for content-based image retrieval. In *Proc ACM SIGMOD Int'l Conf Management of Data*, pp. 599–610.

Knuth, D. (1973). *The Art of Computer Programming - Sorting and Searching*, Volume 3. Addison-Wesley.

Krishnamachari, S. and M. Abdel-Mottaleb (1999). Image browsing using hierarchical clustering. In *IEEE Symp Computers and Communications*, pp. 301–307.

Kruskal, J. (1964). Multi-dimensional scaling by optimizing goodness-of-fit to a nonmetric hypothesis. *Psychometrika 29*, 1–27.

Lalmas, M. and T. Rölleke (2004). Modelling vague content and structure querying in XML retrieval with a probabilistic object-relational framework. In *Proc Int'l Conf Flexible Query Answering Systems.*

Landauer, T. and S. Dumais (1997). A solution to Plato's problem: the latent semantic analysis theory of acquisition, induction, and representation. *Psychological Review 104*(2), 211–240.

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

Lee, J. (1997). Analyses of multiple evidence combination. In *Proc Int'l ACM SIGIR Conf on Research and Development in Information Retrieval*, pp. 267–276.

Leuski, A. and J. Allan (2000). Improving interactive retrieval by combining ranked list and clustering. In *Proc Int'l Conf Recherche dinformation assistée par ordinateur*, pp. 665–681.

Li, B., E. Chang, and C. Wu (2002). Dpf — a perceptual distance function for image retrieval. In *Proc IEEE Int'l Conf Image Processing*, pp. 597–600.

Li, L., C. Stoeckert, and D. Roos (2003). OrthoMCL: Identification of ortholog groups for eukaryotic genomes. *Genome Research 13*, 2178–2189.

Lim, J., J. Wu, S. Singh, and D. Narasimhalu (2001). Learning similarity matching in multimedia content-based retrieval. *IEEE Trans Knowledge and Data Engineering 13*(5), 846–850.

Lin, W.-H. and A. Hauptmann (2004). Merging rank lists from multiple sources in video classification. In *Proc IEEE Int'l Conf on Multimedia and Expo*, pp. 1535–1538.

Ma, W. and B. Manjunath (1996). Texture features and learning similarity. In *Proc IEEE Conf Computer Vision and Pattern Recognition*, pp. 425–430.

Ma, W. and B. Manjunath (1999). Netra: a toolbox for navigating large image databases. *Multimedia Systems 7*(3), 184–198.

Mainzer, K. (2004). *Einführung in die Computerphilosophie.* Junius Verlag.

Malik, J., S. Belongie, T. Leung, and J. Shi (2001). Contour and texture analysis for image segmentation. *Int'l Journal Computer Vision 43*(1), 7–27.

Manjunath, B. and W. Ma (1996). Texture features for browsing and retrieval of image data. *IEEE Trans Pattern Analysis and Machine Intelligence 18*(8), 837–842.

Manjunath, B., J.-R. Ohm, V. Vasudevan, and A. Yamada (2001). Color and texture descriptors. *IEEE Trans Circuits and Systems for Video Technology 11*(6), 703–715.

Marcotte, E., M. Pellegrini, M. Thompson, T. Yeates, and D. Eisenberg (1999). A combined algorithm for genome-wide prediction of protein function. *Nature 402*, 83–86.

Meila, M. and J. Shi (2001). Learning segmentation by random walks. In *Proc Conf Neural Information Processing Systems*, Volume 13.

Meilhac, C. and C. Nastar (1999). Relevance feedback and category search in image databases. In *Proc Int'l Conf Multimedia Communications Systems*, pp. 512–517.

Milligan, G., S. Soon, and L. Sokol (1983). The effect of cluster size, dimensionality, and the number of clusters on recovery of true cluster structure. *IEEE Trans Patterns Analysis and Machine Intelligence 5*(1), 40–47.

Minka, T. and R. Picard (1996). Interactive learning using a society of models. In *Proc IEEE Conf Computer Vision and Pattern Recognition*, pp. 447–452.

Müller, H., S. Marchand-Maillet, and T. Pun (2002). The truth about Corel - evaluation in image retrieval. In *Proc Int'l Conf Image and Video Retrieval*, pp. 38–49. LNCS 2383, Springer.

Müller, H., W. Müller, D. M. Squire, M. Marchand-Maillet, and T. Pun (2000). Strategies for positive and negative relevance feedback in image retrieval. In *Proc Int'l Conf Pattern Recognition*, pp. 5043.

Müller, W. and A. Henrich (2004). Faster exact histogram intersection on large data collections using inverted VA-files. In *Proc Int'l Conf Image and Video Retrieval*, pp. 455–463. LNCS 3115, Springer.

Nastar, C., M. Mischke, and C. Meilhac (1998). Efficient query refinement for image retrieval. In *Proc IEEE Conf Computer Vision and Pattern Recognition*, pp. 547.

Newman, M. (2003). The structure and function of complex networks. *SIAM Rev 45*, 167–256.

Ng, A., M. Jordan, and Y. Weiss (2005). On spectral clustering: analysis and an algorithm. In *Proc Conf Neural Information Processing Systems*.

Peng, J., B. Bhanu, and S. Qing (1999). Probabilistic feature relevance learning for content-based image retrieval. *Computer Vision and Image Understanding 75*(12), 150–164.

Pečenović, Z., M. Do, M. Vetterli, and P. Pu (2000). Integrated browsing and searching of large image collections. In *Proc Int'l Conf Advances in Visual Information Systems*, pp. 279–289. LNCS 1929, Springer.

Picard, R. and T. Minka (1995). Vision texture for annotation. Technical report, MIT Media Laboratory.

Pickens, J. (2001). A survey of feature selection techniques for music information retrieval. Technical report, Center for intelligent information retrieval, University of Massachusetts.

Porkaew, K., K. Chakrabarti, and S. Mehrotra (1999). Query refinement for multimedia similarity retrieval in MARS. In *Proc ACM Int'l Conf Multimedia*, pp. 235–238.

Preparata, F. and M. Shamos (1985). *Computational Geometry*. Springer Verlag.

Qamra, A., Y. Meng, and E. Chang (2005). Enhanced perceptual distance functions and indexing for near-replica image recognition. *Trans Pattern Analysis and Machine Intelligence 27*(3), 379–391.

Rand, W. (1971). Objective criterion for evaluation of clustering methods. *Journal Of American Statistical Association 66*, 846–851.

Rocchio, J. J. (1971). *The SMART Retrieval System. Experiments in Automatic Document Processing*. Prentice Hall.

Rubner, Y., L. Guibas, and C. Tomasi (1997). The earth mover's distance, multi-dimensional scaling, and color-based image retrieval. In *Proc ARPA Image Understanding Workshop*, pp. 661–668.

Rubner, Y., C. Tomasi, and L. Guibas (1998). A metric for distributions with applications to image databases. In *Proc IEEE Int'l Conf Computer Vision*, pp. 59–66.

Rui, Y. and T. Huang (2000). Optimizing learning in image retrieval. In *Proc IEEE Conf Computer Vision and Pattern Recognition*, pp. 236–243.

Rui, Y., T. Huang, and S. Mehrotra (1997). Content-based image retrieval with relevance feedback in MARS. In *Proc IEEE Int'l Conf on Image Processing*, pp. 815–818.

Rui, Y., T. Huang, M. Ortega, and S. Mehrotra (1998). Relevance feedback: A power tool for interactive content-based image retrieval. *IEEE Trans Circuits, Systems and Video Technology 8*(5), 644–655.

Russell, S. (1988). Analogy and similarity. In *Analogical Reasoning (D. Helman, ed.)*. Kluwer.

Ruthven, I. and M. Lalmas (2003). A survey on the use of relevance feedback for information access systems. *Knowledge Engineering Review 18*(1), 95–145.

Salton, G. and M. McGill (1982). *Introduction to Modern Information Retrieval*. McGraw-Hill Book Company.

Santini, S., A. Gupta, and R. Jain (2001). Emergent semantics through interaction in image databases. *IEEE Trans Knowledge and Data Engineering 13*(3), 337–351.

Santini, S. and R. Jain (2000). Integrated browsing and querying for image databases. *IEEE Multi-Media 7*(3), 26–39.

Schettini, R., G. Ciocca, and I. Gagliardi (1999). Content-based color image retrieval with relevance feedback. In *Proc IEEE Int'l Conf Image Processing*, pp. 75–79.

Schvaneveldt, R. (1990). *Pathfinder Associative Networks: Studies in Knowledge Organization*. Norwood, NJ: Ablex.

Sclaroff, S., L. Taycher, and M. L. Cascia (1997). ImageRover: A content-based image browser for the world wide web. Technical report, Boston University.

Shepard, R. (1962a). Multidimensional scaling with an unknown distance function. Part I. *Psychometrika 27*, 125–140.

Shepard, R. (1962b). Multidimensional scaling with an unknown distance function. Part II. *Psychometrika 27*, 219–246.

Si, L. and J. Callan (2002). Using sampled data and regression to merge search engine results. In *Proc Int'l ACM SIGIR Conf on Research and Development in Information Retrieval*, pp. 19–26.

Si, L. and J. Callan (2003). A semisupervised learning method to merge search engine results. *ACM Trans Information Systems 21:4*, 457–491.

Simonsen, K. and L. McIntyre (2004). Using alpha wisely: Improving power to detect multiple QTL. *Statistical Applications in Genetics and Molecular Biology 3*, 1–24.

Song, C., S. Havlin, and H. Makse (2005). Self-similarity of complex networks. *Nature 433*, 392–395.

Späth, H. (1985). *Cluster Analysis and Dissection.* Chichester, U.K.: Ellis Horwood.

Stanyon, C., G. Liu, B. Mangiola, N. Patel, L. Giot, B. Kuang, H. Zhang, J. Zhong, and R. Finley (2004). A Drosophila protein-interaction map centered on cell-cycle regulators. *Genome Biology 5*(12), R96.

Swets, D. and J. Weng (1999). Hierarchical discriminant analysis for image retrieval. *IEEE Trans Pattern Analysis and Machine Intelligence 21*(5), 386–401.

Tamura, H., S. Mori, and T. Yamawaki (1978). Textural features corresponding to visual perception. *IEEE Trans Systems, Man and Cybernetics 8*(6), 460–472.

Tenenbaum, J. (1997). Mapping a manifold of perceptual observations. In *Proc Conf Neural Information Processing Systems.*

Tenenbaum, J., V. de Silva, and J. Langford (2000). A global geometric framework for nonlinear dimensionality reduction. *Science 290*(5500), 2319–2323.

Tieu, K. and P. Viola (2000). Boosting image retrieval. In *Proc IEEE Conf Computer Vision and Pattern Recognition*, pp. 228–235.

Tong, S. and E. Chang (2001). Support vector machine active learning for image retrieval. In *Proc ACM Int'l Conf Multimedia*, pp. 107–118. ACM Press.

Torgerson, W. (1958). *Theory and methods of scaling.* Wiley.

Torgerson, W. (1965). Multidimensional scaling of similarity. *Psychometrika 30*, 379–393.

Tukey, J. (1991). The philosophy of multiple comparisons. *Statistical Science 6*, 100–116.

Tversky, A. (1977). Features of similarity. *Psychological Review 84*, 327–352.

Urban, J. and J. Jose (2004a). EGO: A personalised multimedia management tool. In *Proc Int'l Workshop Adaptive Multimedia Retrieval*, pp. 3–17.

Urban, J. and J. Jose (2004b). Evidence combination for multi-point query learning in content-based image retrieval. In *Proc IEEE Int'l Symposium Multimedia Software Engineering*, pp. 583–586.

Urban, J., J. Jose, and C. van Rijsbergen (2003). An adaptive approach towards content-based image retrieval. In *Proc Int'l Workshop Content-based Multimedia Indexing*, pp. 119–126.

van Dongen, S. (2000). *Graph Clustering by Flow Simulation.* Ph. D. thesis, University of Utrecht.

Vapnik, V. N. (1995). *The Nature of Statistical Learning Theory.* Springer.

Vasconcelos, N. and A. Lippman (2000). Bayesian relevance feedback for content-based image retrieval. In *Proc IEEE Workshop Content-based Access of Image and Video Libraries*, pp. 63.

Vorhees, E. and L. Buckland (2005). *The Thirteenth Text REtrieval Conference.* NIST Special Publication: 500-261 Text Retrieval Conference 2004.

Watts, D. and S. Strogatz (1998). Collective dynamics of 'small-world' networks. *Nature 393*, 440–442.

Weber, R., J.-J. Schek, and S. Blott (1998). A quantitative analysis and performance study for similarity-search methods in high-dimensional space. In *Proc Int'l Conf Very Large Databases*, pp. 194–205.

Weigel, F., H. Meuss, K. Schulz, and F. Bry (2004). Content and structure in indexing and ranking XML. In *Int'l Workshop Web and Databases*, pp. 67–72.

Wolfram, S. (2004). *A New Kind of Science.* Wolfram Ltd.

Wu, A., M. Garland, and J. Han (2004). Mining scale-free networks using geodesic clustering. In *Proc Int'l Conf Knowledge Discovery and Data Mining*, pp. 719–724.

Wu, L., C. Faloutsos, K. Sycara, and T. Payne (2000). Falcon: Feedback adaptive loop for content-based retrieval. In *Proc Int'l Conf Very Large Databases*, pp. 297–306.

Wu, P. and B. Manjunath (2001). Adaptive nearest neighbour search for relevance feedback in large image databases. In *Proc ACM Multimedia*, pp. 89–97.

Yanai, I., J. Mellor, and C. DeLisi (2002). Identifying functional associations by linking genes through conserved chromosomal proximity. *Trends Genetics 18*, 176–179.

Yavlinsky, A., M. Pickering, D. Heesch, and S. Rüger (2004). A comparative study of evidence combination strategies. In *Proc IEEE Int'l Conf Acoustics, Speech, and Signal Processing*, Volume 3, pp. 1040–1043.

Yavlinsky, A., E. Schofield, and S. Rüger (2005). Automated image annotation using global features and robust nonparametric density estimation. In *Proc Int'l Conf Video and Image Retrieval*, pp. 507–517. LNCS 3568, Springer.

Yeung, M. and B. Liu (1995). Efficient matching and clustering of video shots. In *Proc IEEE Int'l Conf Image Processing*, pp. 338–341.

Yeung, M. and B. Yeo (1997). Video visualization for compact presentation and fast browsing of pictorial content. *IEEE Trans Circuits and Systems for Video Technology 7*, 771–785.

Yoon, J. and M. Jayant (2001). Relevance feedback for semantics based image retrieval. In *Proc IEEE Int'l Conf Image Processing*, pp. 42–45.

Zamir, O. and O. Etzioni (1998). Web document clustering: A feasibility demonstration. In *Proc Special Interest Group Information Retrieval*, pp. 46–54.

Zhang, H. and D. Zhong (1995). A scheme for visual feature based image indexing. In *Proc SPIE/IS&T Conf Storage and Retrieval for Image and Video Databases III, Vol 2420*, pp. 36–46.

Zhang, H.-J. and Z. Su (2001). Improving CBIR by semantic propagation and cross modality query expansion. In *Proc Multimedia Content-based Indexing and Retrieval*, pp. 79–82.

Zhang, R., Z. Zhang, M. Li, W.-Y. Ma, and H.-J. Zhang (2005). A probabilistic semantic model for image annotation and multi-modal image retrieval. In *Proc Int'l Conf Computer Vision (to appear)*.

Zhou, X. and T. Huang (2003). Relevance feedback in image retrieval: a comprehensive review. *ACM Multimedia Systems 8*(6), 536–544.

Zhou, X., M. Kao, and W. Wong (2002). Transitive functional annotation by shortest path analysis of gene expression data. *Proceedings of the National Academy of Sciences 99*, 12783–12788.