NOTE TO USERS

This reproduction is the best copy available.

UMI®

uOttawa

L'Université canadienne
Canada's university

FACULTÉ DES ÉTUDES SUPÉRIEURES ET POSTDOCTORALES

uOttawa

L'Université canadienne
Canada's university

FACULTY OF GRADUATE AND POSTDOCTORAL STUDIES

Ilya Volnyansky
AUTEUR DE LA THÈSE / AUTHOR OF THESIS

M.Sc. (Mathematics)
GRADE / DEGREE

Department of Mathematics and Statistics
FACULTÉ, ÉCOLE, DÉPARTEMENT / FACULTY, SCHOOL, DEPARTMENT

Curse of Dimensionality in the Application of Pivot-based Indexes to the Similarity Search Problem
TITRE DE LA THÈSE / TITLE OF THESIS

V. Pestov
DIRECTEUR (DIRECTRICE) DE LA THÈSE / THESIS SUPERVISOR

CO-DIRECTEUR (CO-DIRECTRICE) DE LA THÈSE / THESIS CO-SUPERVISOR

EXAMINATEURS (EXAMINATRICES) DE LA THÈSE / THESIS EXAMINERS

P. Bose

R. Kulic

D. McDonald

Gary W. Slater
Le Doyen de la Faculté des études supérieures et postdoctorales / Dean of the Faculty of Graduate and Postdoctoral Studies

Curse of Dimensionality in the Application of Pivot-based Indexes to the Similarity Search Problem

Ilya Volnyansky

Thesis Supervisor: Vladimir Pestov

Thesis Submitted to the Faculty of Graduate and Postdoctoral Studies in Partial Fulfilment of the Requirements for the Degree of Master of Science in Mathematics[1]

Department of Mathematics and Statistics
Faculty of Science
University of Ottawa

©Ilya Volnyansky, Ottawa, Canada, 2009

[1]The M.Sc. program is a joint program with Carleton University, administered by the Ottawa-Carleton Institute of Mathematics and Statistics

Library and Archives Canada

Published Heritage Branch

395 Wellington Street
Ottawa ON K1A 0N4
Canada

Bibliothèque et Archives Canada

Direction du Patrimoine de l'édition

395, rue Wellington
Ottawa ON K1A 0N4
Canada

Your file Votre référence
ISBN: 978-0-494-58175-9
Our file Notre référence
ISBN: 978-0-494-58175-9

NOTICE:

The author has granted a non-exclusive license allowing Library and Archives Canada to reproduce, publish, archive, preserve, conserve, communicate to the public by telecommunication or on the Internet, loan, distribute and sell theses worldwide, for commercial or non-commercial purposes, in microform, paper, electronic and/or any other formats.

The author retains copyright ownership and moral rights in this thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without the author's permission.

AVIS:

L'auteur a accordé une licence non exclusive permettant à la Bibliothèque et Archives Canada de reproduire, publier, archiver, sauvegarder, conserver, transmettre au public par télécommunication ou par l'Internet, prêter, distribuer et vendre des thèses partout dans le monde, à des fins commerciales ou autres, sur support microforme, papier, électronique et/ou autres formats.

L'auteur conserve la propriété du droit d'auteur et des droits moraux qui protège cette thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation.

In compliance with the Canadian Privacy Act some supporting forms may have been removed from this thesis.

While these forms may be included in the document page count, their removal does not represent any loss of content from the thesis.

Conformément à la loi canadienne sur la protection de la vie privée, quelques formulaires secondaires ont été enlevés de cette thèse.

Bien que ces formulaires aient inclus dans la pagination, il n'y aura aucun contenu manquant.

Canada

Abstract

In this work we study the validity of the so-called curse of dimensionality for indexing of databases for similarity search. We perform an asymptotic analysis, with a test model based on a sequence of metric spaces (Ω_d) from which we pick datasets X_d in an i.i.d. fashion. We call the subscript d the dimension of the space Ω_d (e.g. for $\mathbb{R}^d$ the dimension is just the usual one) and we allow the size of the dataset $n = n_d$ to be such that d is superlogarithmic but subpolynomial in n.

We study the asymptotic performance of pivot-based indexing schemes where the number of pivots is $o(n/d)$. We pick the relatively simple cost model of similarity search where we count each distance calculation as a single computation and disregard the rest.

We demonstrate that if the spaces Ω_d exhibit the (fairly common) concentration of measure phenomenon the performance of similarity search using such indexes is asymptotically linear in n. That is for large enough d the difference between using such an index and performing a search without an index at all is negligeable. Thus we confirm the curse of dimensionality in this setting.

Introduction

One often hears the complaint that we live in a data rich but information poor society. Indeed, enabled by advances in IT hardware all sorts of data are being gathered at astonishing rates. The force behind this is the assumption that useful information is to be found in the heaps of data. However slim the ratio of useful to useless data may be, the exercise is considered worthwhile.

As this is almost inevitably complicated, a new area of active research and a business sector were created – those of "data mining". We will take a look at what is perhaps the most fundamental data mining problem of all: given a new piece of data, finding similar pieces of data in the pile you have accumulated already. This is the problem of similarity search. It should not be confused with exact search, the topic usually discussed in introductory computer science textbooks.

The difference is that of finding a book in a library 20 years ago and today. In the past, knowing that you wanted to find a book having "Pantagruel" in its title meant somehow finding out, by perhaps talking to librarians (who one would assume read many books) that the author is one F. Rabelais. Then, you would walk down the "fiction" section of authors starting with R, and hopefully within a few minutes locate the title. Today, sitting at home and without relying on knowledgeable librarians you would start reading the first page of the book within seconds of searching for "Pantagruel" in a Web browser. This miracle is due in no small part to clever solutions to the similarity search problem called indexes. Indexes are structures that organize a database in such a way that fast similarity search is possible.

Is the similarity search problem solved then? Let us consider a slightly different problem: we are given a photograph of a mountain landscape with no clear giveaways as to the location and yet we would like to know where it was taken. A person who knows those mountains will immediately tell you the approximate location, as she is able to identify the particular vegetation, rock formations and glaciers. But finding such a person is considerably tougher than going to your local library. It would be most helpful if one could submit this photograph to a search engine containing millions of tagged pictures (the Web?) to find the most similar ones. This way our untagged picture will obtain a tag: namely geographic information. That no such solution exists (yet) is a testament to the difficulty of search in high dimensions. Untagged pictures are composed of thousands if not millions of coloured pixels and it is not immediately obvious

how to teach a computer to quickly find similar ones.

This has come to be known as the "curse of dimensionality" and is the primary topic of this work. We aim to provide an asymptotic analysis of a class of indexes applied to high dimensional datasets of "typical" behaviour. The broad conclusion is that high dimension leads almost inevitably to unacceptably slow performance of search, akin to waiting 3 days for the web browser to tell you about *Gargantua and Pantagruel* (enough to make the local library suddenly competitive).

In Chapter 1, we introduce the formal mathematical setting of this search problem by defining a space of all queries and datapoints. A distinction is made between *the database, a finite collection of objects* and *the potentially infinite* larger space that acts as the source of "new" objects – the centres of queries. Similarity queries are based on objects from the larger space but all that is known a priori is the database which must serve to infer patterns in the larger space. We provide a cost model for pivot-based indexes and setup the relationships between all the relevant variables for asymptotic analysis: primarily the database size, index size and dimension.

In Chapter 2, a related phenomenon from asymptotic geometric analysis is presented: the concentration of measure. It is the somewhat counterintuitive observation that high-dimensional objects appear to be small when we base our measurements on samples. For example, when a sphere is sampled the so-called observable diameter as a function of dimension tends to 0 very quickly.

We present various families of spaces, which we call Lévy families, that exhibit the more particular normal concentration of measure. These families are used as examples of spaces that are interesting to index yet exhibit geometric properties that make indexing hard. We show how these geometric properties of the larger spaces imply something reminiscent of the curse of dimensionality for pivot-based indexes.

In Chapter 3, we introduce Statistical Learning theory as a tool to connect concentration of measure on spaces to finite datasets, which are treated as random samples from these spaces. Our interest is in a generalization of the theorem of Glivenko-Cantelli, due to Vapnik and Chernovenkis, about uniform convergence of sampled quantities to their true values. The crucial condition for the applicability of this theorem is that the class of balls in a space have a low "capacity": among the different capacity measures that can be used here is the VC dimension. If this condition is met we can make conclusions about the behaviour of any random dataset.

The main theorem is presented in Chapter 4: that concentration of measure leads, under certain very natural conditions, to the curse of dimensionality for pivot indexes. That is, within our model we give a mathematically rigorous proof that asymptotically in dimension, all pivot-based indexes are not significantly better than a simple sequential scan of the database. We derive certain properties of the speed with which this degradation in performance takes place as well. The asymptotic bound is strong: the performance degrades quickly in dimension to the point that at least half of all possible queries will almost surely require a full scan of the database no matter which pivot-based is used

and with which parameters (under some reasonable limits on space). Care is taken to present the full set of assumptions of this asymptotic analysis, as its conclusion may not hold in other cases, for example when the index is allowed a large amount of storage for pre-computation.

Although the very results we prove demonstrate that indexing in high dimensions is often hard, in Chapter 5 we perform several experiments with pivot-based indexes on two different kinds of datasets. The broad conclusion is that no particular flavour of index does much better than the rest: performance quickly diminishes, which would leave some tough choices for database designers who deal with high-dimensional data. Perhaps a pertinent question is whether real-life datasets are ever truly high dimensional: it seems to be the opinion of some researchers that they almost never are. In this case the doom and gloom we prognosticate is mainly theoretical.

Contents

1 Indexing of metric space databases for similarity search **6**
- 1.1 The search problem . . . 8
- 1.2 Indexing for search . . . 10
- 1.3 The curse of dimensionality . . . 13
- 1.4 Pivot-based indexing . . . 17
- 1.5 Approximate Search . . . 19

2 The concentration of measure phenomenon **21**
- 2.1 Examples . . . 24
- 2.2 Link to concentration of measure . . . 28
- 2.3 Radius of queries in Lévy families . . . 30

3 Statistical learning theory **32**
- 3.1 Calculating a VC dimension . . . 35
- 3.2 Rates of convergence . . . 38

4 Linking theoretical and empirical observations **41**
- 4.1 Main result . . . 43

5 Indexing experiments **46**

Chapter 1

Indexing of metric space databases for similarity search

A database is a collection of records with an added structure that allows the user to query, update, and delete records in a variety of ways. Databases are extremely pervasive in modern life: the contacts list on a phone, a bank's client records, the whole Internet, a grocery store's transactions... In fact this general definition of a database is not restricted to computerized systems: the ancient library of Alexandria was a database as well.

What computerized systems have allowed a veritable explosion in the size and number of databases. A parallel for processors is the famous Moore's law that hypothesized an exponential growth in the number of components on chips. The original prediction shown in Figure 1.1 has been valid, up to a constant, for over 40 years. Although less well documented, it has been an accepted truth in the business community that the size of databases has been expanding exponentially as well: e.g. [Kim] p.295 and [Heg01].

What seems to be happening is a sort of Moore's Law for the size of databases as they keep pace with the rise in processing power.

At stake is the scalability of database systems, as superlinear algorithms for querying, updating and deleting are necessarily experiencing a continuous degradation of their performance. Thus finding more efficient algorithms is not becoming less important with rising computing speed as is sometimes suggested by non-specialists. Put another way, our expectations of increase in performance exceed even the astonishing pace of Moore's Law and so more clever algorithms need to be devised.

To illustrate the main topic of this work, we will briefly summarize what searching a database typically means in real life. The most common databases today consist of several dozen (or more) big *tables*. An example of a 6-column database table is provided in (pardon the overloading of meaning) Table 1. The

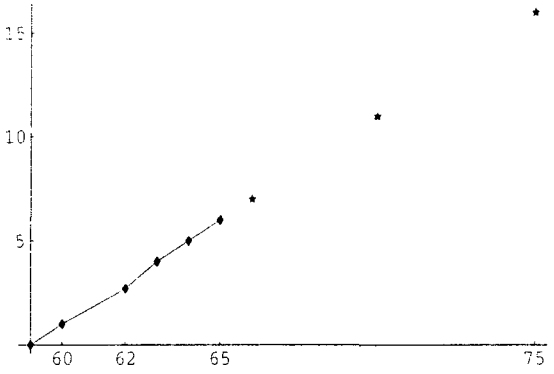

Figure 1.1: The original Moore's Law. $\log_2$ of Components per chip vs Year

two example rows are *records*, and typically millions of them would exist in a large company or government department. Customer ID is in database parlance a *primary key*: it is used by the system to uniquely identify the record.

Customer ID	last name	first name	address	postal code	add date
12783	Black	Conrad	50 Beechwood Dr.	M1F6H0	2008-05-01
25456	Minkowski	Alice	204-190 5th Avenue	H8F1K4	2006-09-25

Table 1.1: a typical database table

Perhaps the most well known function of a database is a search of its records. An example could be the retrieval of all customers who were entered into the database in May 2007. While it is possible to execute this request by looking sequentially at each record and seeing if the "add date" attribute matches, a more efficient solution is possible.

Roughly speaking, in a database with n records the sequential scan method proposes to look at n records while the more efficient one will only do about $\log(n)$ lookups. One such method is the B-tree ([Sed98]), a structure that puts the values of the attribute to be searched in a tree of depth $\log(n)$ and hence exact search takes only about that many operations. In fact the time to perform search can be thought of as constant: even for $n = 10^9$ a search query requires only 2 comparisons. [Sed98]

This "classical" search problem is actually a very particular case of *similarity search*. To understand why other kinds of search are interesting, it is helpful to realize that a B-tree or similar structures only deal with one numeric attribute at a time. A structure is built on that attribute alone, which makes for efficient searches of a very particular kind: those based on that single attribute. Multiple B-trees would have to be built if search on other attributes is to be expected, and things get more complicated still with complex queries involving multiple attributes at the same time.

When more dimensions are added to the mix, the problem not only gets more difficult, but no efficient solutions seem to exist. As multiple attributes at once are to be taken into account it is no longer possible to order all the records in a line. We can only say, given two records, how close they are to each other.

A database of fingerprints, stored as an array of black and white pixels, can be used to illustrate similarity search. The typical query is to find all the similar fingerprints to a new sample from the street. This similarity is computed using some function of the pixels forming the image of the query and any candidate from the database. This is an expensive operation that we would like to minimize. In classical search we would have liked to assign a key to each of the set of fingerprints in the world. This way we could ask for "fingerprint set 13747"

The major problem however is that the point precisely is that no two sets are exactly the same, even if taken from the same person. The first set will be 13747 while the second could be 78965, thus making this numbering scheme of little use – unless we have some way of ensuring that the second set of fingerprints will be always assigned a number "close" to 13747. It is realistic to assume that police investigators are interested in the "closest 50 matches" which they then would like to review by hand. The database designer then has to provide this capability with just one piece of structural information: the function that measures the similarity between two sets of fingerprints.

This problem therefore lends itself well to be characterized in terms of metric spaces, (alternative characterizations, without metric spaces, exist [Goy08]).

1.1 The search problem

Formally, the problem is framed in terms of a metric space:

Definition 1.1. A *metric space* is a set Ω equipped with function $\rho : \Omega \times \Omega \to \mathbb{R}$ s.t.

- $\rho(\omega_1, \omega_2) \geqslant 0$, $= 0$ if and only if $\omega_1 = \omega_2$
- $\rho(\omega_1, \omega_2) = \rho(\omega_2, \omega_1)$ and
- $\rho(\omega_1, \omega_2) \leqslant \rho(\omega_1, \omega_3) + \rho(\omega_3, \omega_2)$

The function ρ is a *distance function* (also: *metric*) and it is the main mathematical structure of Ω. Its most important feature perhaps is the last property, known as the *triangle inequality*. Essentially it is the only tool available for inferring distances in a metric space.

In addition the metric space is equipped with a probability measure, the definition of which we assume is known to the reader or can be looked up in a text like [Tay06]. It is a way of assigning a weight to "measurable" subsets in the space Ω to account for different likelihoods of a random point falling into them. At least in the case of a finite set it is simple enough:

Definition 1.2. For a finite set X a *probability measure* on the set is a function $\mu : 2^X \to [0,1]$ s.t. $\sum_{x \in X} \mu(\{x\}) = 1$, so that $\forall A \subset X$, $\mu(A) = \sum_{x \in A} \mu(\{x\})$.

What we call a *dataset* is a finite subset $X \subset \Omega$ equipped with the inherited metric $\rho|_X$ and normalized counting measure

$$\mu_\#(A) = \frac{|A \cap X|}{|X|}$$

(which is indeed a probability measure).

There is good reason to distinguish between the space Ω and the dataset X we have on hand. In the fingerprints database example, we cannot claim to have all known and future fingerprints already in the database. More likely is the opposite: every time a set of fingerprints come in for inspection, no exact matching set exists on file. So the new set must have come from outside X, yet X is the space we are trying to search.

What can Ω be in this case? All the possible fingerprint impressions, taken under all possible various conditions, of the entire human population? Past, present, future? The exact specification of Ω may be hard to formalize and is ultimately unimportant. Perhaps a parallel can be drawn with probability theory where Ω stands for sample space and X for a random variable.

Furthermore, the measure μ is as a consequence also hard to specify, but it seems reasonable to assume that in any case, some fingerprints are more likely than others to show up as queries to our database. Then all we can safely assume about μ is that it is some non-uniform measure, which is unknown to us.

X on the other hand is quite concrete – we literally have a list of all the elements at our disposal. In addition to being a subset of Ω it is a metric subspace in the sense that the same metric ρ is used to calculate distances between points of X. To simply inherit μ is problematic as we don't actually know what it is. Nevertheless we would like to do computations in X taking probabilities into account. The solution is to use the counting measure, also known as the empirical measure. This may seem crude but it is actually a pretty good approximation of the "real" measure as a consequence of a well-known theorem in statistics (more on this later).

Given this structure we can perform the following similarity queries:
Given the key $q \in \Omega$,

- Nearest neighbour: find the k closest elements in X to q
- Range: find all the elements in X within distance r from q
- Proportion (variation on the first): return k closest fraction of X to q

An example above was of a 50 nearest neighbour query. Supposing that such a similarity query on a fingerprints databases would rely on counting common "features", a range query can be "find all records that have at least 10 features identical to this sample q". What these two queries have in common is the

underlying idea that they will return a small set of results that a human can then manually examine. Effectively the different kinds of similarity queries are closely related, and studying one sort is not a serious limitation (see [Cha01]).

More specialized queries, e.g. finding *all* pairs of nearest neighbours, are approached differently both in the building and analysis of algorithms and so will not be covered.

1.2 Indexing for search

To answer a similarity query we can revert to the strategy of looking up every element in $x \in X$ and calculating $\rho(q, x)$. We will call this a linear scan, as it is clearly linear in the size of the dataset. However we will suppose (what is often the case) that the calculation of distances $\rho(q, x)$ is the most expensive operation [Cha01].

In such a setting the linear scan approach is very slow: if a single distance computation takes 1/100th of a second, several weeks would be needed to traverse a database with 100 million records. Distance functions that take milliseconds and more to execute are mentioned for examples in [Cia98].

An *index* is an added structure to the database that facilitates operations like searching. Here we will restrict the use of the word to the scope of this work:

Definition 1.3. An index is a data structure whose aim is to speed up the execution of similarity queries on a particular dataset, typically consisting of some pre-calculated values and an algorithm.

Let's quickly introduce a simple index, Orchard's algorithm [Cla05], as an example:

Example 1.1. For the n points in X, we create an $n \times n$ matrix of distances $\rho(x, y)$: each row corresponding to all the distances to x, sorted in increasing order. To perform 1-nearest neighbour search query with centre q, we pick a random element x and go throgh the row of distances corresponding to x. If we find y s.t. $\rho(y, q) < \rho(x, q)$ we switch to the row of y.

We avoid going through all the elements of X by applying the following criterion: we stop going through the list of y if the last seen element z satisfies $\rho(z, y) > 2\rho(y, q)$. This follows from

$$\begin{aligned} & \rho(z, y) \leqslant \rho(y, q) + \rho(q, z) \\ \Rightarrow \quad & \rho(z, q) \geqslant \rho(z, y) - \rho(y, q) \\ \Rightarrow \quad & \rho(z, q) > 1/2\rho(z, y) > \rho(y, q) \end{aligned}$$

The assumption having been applied twice in the last part. If we can't find z such that $\rho(z, y) > 2\rho(y, q)$, y is returned as the answer. We can further improve the search by applying various strategies to avoid unnecessary lookups.

Practically speaking the reason to have an index is that the several weeks taken by a naïve approach can be reduced to a few hours, maybe even only a

few minutes. The consequences cannot be underestimated, for, as amazing is our ability to amass data the "killer app" is search: a collection of 100 million sets of fingerprints is interesting, but a system capable of handling thousands of queries per day is downright useful.

Dozens of ideas of how to build such an index were presented and new ones, having various advantages and analyzed in different ways, are invented continuously. Attempts at categorizing and providing a unified framework are recent: a book on the subject appeared in 2005 [Zez05] and the first international conference on similarity search [SISAP] was held in 2008.

Several high-level views exists as to how mathematically describe the function of an index in performing similarity search.

One is outlined in [Cha01]: an indexing scheme works by partitioning the space Ω into regions. In other words the space is decomposed as $\Omega = \cup_i R_i$ where the R_i are finitely many, pairwise disjoint subsets of the space. A query algorithm takes advantage of this partitioning as follows. Through the use of some inequalities – the triangle inequality in some form – we *discard* some of the regions as it can be proved that no elements of X lying in those regions can possibly be in the query results. The elements of X in the undiscarded regions have their distances $\rho(q, x)$ computed through a linear scan to determine if they should be returned. As mentioned above a linear scan is nothing but a sequential lookup of each of the region's elements, so the key is to eliminate as many regions as possible. A high level pseudocode description is given in Algorithm 1.

```
Data: query, index
Result: queryResults
for each region in index.Regions do
    if (NOT index.IsExcluded(region , query)) then
        append LinearSearch(query, region) to queryResults;
    else
    end
end
return queryResults;
```

Algorithm 1: Use of an index for query execution: regions

The triangle inequality and some variations thereof, as used in indexing to avoid sequential scan are listed in [Zez05]:

- Double sided triangle inequality :

$$|\rho(\omega_1, \omega_3) - \rho(\omega_2, \omega_3)| \leqslant \rho(\omega_1, \omega_2) \leqslant \rho(\omega_1, \omega_3) + \rho(\omega_3, \omega_2)$$

  in other words, knowing distances to a third point ω_3 gives us constraints on $\rho(\omega_1, \omega_2)$

- We only know that $r_l \leqslant \rho(\omega_1, \omega_3) \leqslant r_h$ while $\rho(\omega_2, \omega_3)$ is still known exactly. Then:

$$\max\{\rho(\omega_2, \omega_3) - r_h, r_l - \rho(\omega_2, \omega_3), 0\} \leqslant \rho(\omega_1, \omega_2) \leqslant \rho(\omega_2, \omega_3) + r_h$$

- Only a range is known: both $r_l \leqslant \rho(\omega_1, \omega_3) \leqslant r_h$ and $r_l^{'} \leqslant \rho(\omega_2, \omega_3) \leqslant r_h^{'}$ Then

$$\max\{r_l^{'} - r_h, r_l - r_h^{'}, 0\} \leqslant \rho(\omega_1, \omega_2) \leqslant r_h + r_h^{'}$$

Given these simple tools the diversity of the various indexing schemes is astonishing. There are tree and flat structures, structures that try to optimize inserts and deletes, trees that are deep or shallow, balanced or not, with claimed computational complexities from constant to exponential in n = size of X. Furthermore because search is a topic of research in multiple disciplines, including for example pattern recognition [Dev97] where a "nearest neighbour" algorithm is almost equivalent to nearest neighbour search, the same solutions have been reinvented multiple times and called different names [Cha01], [Cla05].

Another reason for this multiplicity of algorithms has to do with the fact that no "best" algorithm has been found yet and so a number of solutions offering specific space-time tradeoffs or other features have been developed.

It is important to note that in some situations indexing is not possible. Some metric spaces are so general that the distance function does not provide any usable information. We will first consider a trivial example, that of the 0-1 distance [Cla05].

Example 1.2. Suppose we have (Ω, ρ, μ) such that

$$\rho(\omega_1, \omega_2) = \begin{cases} 0 & \text{if } \omega_1 = \omega_2 \\ 1 & \text{if } \omega_1 \neq \omega_2 \end{cases}$$

Then, all query types reduce to finding an exact match. Furthermore no amount of storage of the distance functions between various elements of X can facilitate a search query as no information can be added. So no *metric based* index works.

Of course this particular example may not seem very convincing as we already mentioned the existence of efficient solutions for exact match. Even in the absence of keys, we could use a hash function [Sed98] to reduce the problem to a quick binary search.

Another, more substantive example is given by [Lif07].

Example 1.3. In this space, based on a graph generated from web mining [Blo00], the distance function is such that for any two distinct elements ω_1, ω_2:

$$1/2 \leqslant \rho(\omega_1, \omega_2) \leqslant 1$$

Then in this case inferring $\rho(\omega_1, \omega_2)$ based on knowledge of distances to a third point ω_3 is not informative. We can verify that all the commonly used variations of the triangle inequality listed above reduce to something we already know, namely that distances lie between 1/2 and 1.

Spaces that are impossible or at least very hard to index are by no means rare – their high incidence is a whole subject of study.

1.3 The curse of dimensionality

An often repeated observation is the inability of algorithms to deal with high dimensional datasets (e.g. [Bey99]) – a phenomenon described as the *curse of dimensionality*. Simply put, when algorithms are run on Euclidean datasets of increasing dimension, performance drops exponentially as a function of dimension.

The concept of dimension in a general metric space is less precise. Clearly it has to obey our intuition in Euclidean space so for example a plane in the 10-dimensional space $\mathbb{R}^d$ is still 2-dimensional, and it would be desirable for a uniformly distributed ball in $\mathbb{R}^d$ to be d-dimensional. But what to do about datasets like sets of fingerprints, or movies, where an intuitive notion of dimension is harder to develop?

One approach is to focus on the metric space properties of Ω, and take advantage of already known concepts for metric spaces, such as packing numbers and ϵ-nets.

Definition 1.4. An ϵ-net C of (Ω, ρ) is a set of points from Ω satisfying

$$\cup_{c \in C} B(c, \epsilon) = \Omega$$

where $B(c, \epsilon) = \{\omega \in \Omega | \rho(\omega, c) \leqslant \epsilon\}$

The size of the *minimal* ϵ-net is called the covering number of (Ω, ρ) and denoted $\mathcal{C}(\Omega, \epsilon)$ as it is a function of ϵ.

It is not hard to convince oneself that at least in Euclidean spaces higher dimension leads to higher covering numbers. For the unit cube in $\mathbb{R}^d$ the covering number is on the order of $1/\epsilon^d$ [Cla05]. This concept is extended [Cla05], to define the *Assouad dimension* (see algorithmic complexity notation later in Table 1.3):

Definition 1.5. The Assouad dimension of (Ω, ρ) is the number d satisfying:

$$\sup_{\omega \in \Omega, r > 0} \mathcal{C}(B(\omega, r), \epsilon r) = 1/\epsilon^{d + o(1)}$$

A small Assouad dimension is a very strong requirement: all the balls in the space have to be well behaved in the sense that they admit small covers. Perhaps unsurprisingly results exist that show feasibility of indexing for similarity search in case of small Assouad dimension [Cla05]. However the curse of dimensionality stands: these algorithms have an exponential dependence on the Assouad dimension. This concept is also too complicated to compute for real datasets where Ω is unknown: it is not clear how to estimate it from X alone.

As the number of proposed concepts of "intrinsic dimension" for the purposes of similarity search is growing, [Pes07] outlines desirable properties we should look for. Included are ease of computation and definition for discrete spaces in such a way that the intrinsic dimension of X is closely related to the intrinsic dimension of Ω.

An easy to compute and reasonable proposal for a dimension is mentioned in [Cha01b]:

$$\tilde{d} = \frac{\mathrm{E}(\rho(X,Y))^2}{2\mathrm{Var}(\rho(X,Y))}$$

where $x, y \sim \mu$, the distribution of points in Ω. Using relatively small samples, the empirical intrinsic dimension as computed for various objects in table 1.2 seems to give reasonable answers.

Space	calculated dimension
20-dimensional uniform unit cube	27.6
the NASA dataset [DIMACS]	3.9
20-dimensional uniform sphere	20.8
100-dimensional uniform sphere	139.5

Table 1.2: Some empirical approximations of Chávez intrinsic dimension

This concept is based on looking at the histogram of distances

$$\{d(q,x)|x \in X\}$$

Given a query centre q, if the histogram of distances from q to points in X shows a lot of "concentration", this will be a hard query to process as most points will need to be checked. By concentrated we mean of low variance, while the mean of the distances is in the numerator to account for different scales. Explained another way, we would like a uniformly distributed unit cube in $\mathbb{R}^d$ have the same dimension as a uniformly distributed cube twice the size in the same space, so there is a need to normalize. This will often be a non-issue as we would normalize the distance function so that $\mathrm{E}(\rho(x,y)) \sim 1$.

The intrinsic dimension of [Cha01b] then is an average measure of all the possible histograms taken from "viewpoints" q. In reality both μ and Ω are either unknown or unworkable so this measure is to be estimated from X. Underlying is the assumption that datasets exhibit a certain amount of homogeneity of viewpoints, that is histograms taken from different q will still look similar – a hypothesis with some experimental validation [Cia98].

For "truly" d-dimensional structures in Euclidean space, e.g. uniformly distributed unit cube, this $\tilde{d}$ corresponds to d (asymptotically).

Using this intrinsic dimension it is possible to derive a lower bound on the number of distance computations required as a function of $\tilde{d}$ [Cha01b]. This bound however is not very strong – on the order of $\tilde{d}\ \ln(n)$. Thus only for large $\tilde{d}$ relative to n is this lower bound significant. This leads to our next topic: how big is dimension allowed to get, in relation to n?

Henceforth we will use d somewhat ambiguously, referring to either the usual notion from vector space theory or one of the intrinsic dimension concepts with the understanding that they are all coincide, at least asymptotically.

As mentioned in [Cha01], for a fixed dimension d and fixed n, we can find indexing schemes that are fast. The problem that we would like to analyze has

to do with scaling these algorithms as *both* d and n grow. This requires us to make precise how fast we let these two quantities grow in relation to each other. This invariably leads to the use of algorithmic complexity notation, a summary of which appears in table 1.3.

notation	definition
$f(t) = O(g(t))$	for some $C > 0$, eventually $f(t) \leq Cg(t)$
$f(t) = o(g(t))$	for *any* $C > 0$, eventually $f(t) \leq Cg(t)$
$f(t) = \Theta(g(t))$	for some $C_1, C_2 > 0$, eventually $C_1 g(t) \leq f(t) \leq C_2 g(t)$
$f(t) = \omega(g(t))$	for *any* $C > 0$, eventually $f(t) \geq Cg(t)$

Table 1.3: Algorithmic complexity notation

An asymptotic analysis will therefore involve both:

$$d \to \infty$$

and

$$n \to \infty$$

We will try to argue for what the relation between d and n should be by going back to the more fundamental question of what is an efficient index. It is clear that we should be able to perform similarity queries with less time than that taken by a linear scan. In the language of algorithmic complexity, we require a sublinear complexity in n, that is

$$\text{querytime} = o(n)$$

where by querytime we mean the average time it takes for a similarity query to execute, time measured in distance computations. The average here is computed over a reasonable space of possible queries, on which we will touch later.

Storage is also important, with at most polynomial storage allowed (but in practice even n^2 may be too much):

$$\text{storage} = n^{O(1)}$$

For our particular indexing scheme the storage is measured by the number of distances *stored*. We do not make a distinction among the different ways to store a real number, in all cases it is considered as 1 unit (of cost). This covers a large number of indexing schemes that are essentially arrays of pre-computed distances.

As our main concern is with asymptotic analysis it is also to specify bounds on d. We will follow an approach in the authoritative survey by [Ind04] and focus on a particular range for d: superlogarithmic but subpolynomial in n. Expressed using the notation,

$$d = \omega(\log n) \tag{1.1}$$

$$d = n^{o(1)} \tag{1.2}$$

The reason for the lower bound (1.1) is due to a case study which requires the definition of Hamming cubes.

Definition 1.6 (The Hamming Cubes Σ^d). The Hamming cube of dimension d is defined as the set of all binary sequences of length d, that is its elements are of the form

$$\boldsymbol{x} = (0, 1, 1, 0, 1, \ldots, 1)$$

and the distance between two strings is just the number of elements they don't have in common divided by d:

$$\rho(\boldsymbol{x}, \boldsymbol{y}) = \frac{\sum_{i=1}^{d} |x_i - y_i|}{d}$$

This metric is known as the normalized Hamming distance. We will give the cube a uniform measure for this discussion.

It turns out that for at least this case, when d grows slowly, say $d = O(\log n)$ the entire space Ω is so small relative to the size of the dataset that all possible queries can be pre-computed and stored without breaking the polynomial storage requirement. The size of $\Omega = \Sigma^d$ is just 2^d which becomes on the order of n for sublogarithmic d. As there are on the order of n possible radii, there are only n^2 possible queries which can be all precomputed. So to build a general framework for asymptotic bounds it seems necessary that d grow strictly faster than $\log n$

As we consider algorithms that are exponential in d to suffer from the curse of dimensionality, we will require querytime polynomial in d ([Ind04]):

$$\text{querytime } = d^{O(1)}$$

This upper bound on d results from the observation that if d grew so fast that $n = d^{O(1)}$ a sequential scan would be polynomial in d. As nothing needs to be proven in that case, we focus on when d is subpolynomial in n and require an algorithm polynomial in d and hence subpolynomial in n.

We will adopt the view that these bounds on d are a reasonable setting for the investigation of performance of various index based query algorithms. While d grows fast enough to not render the problem trivial, we disregard high rates of growth for which proven examples of the "curse" already exist.

Summarizing:

> The goal of finding a scalable index is to find polynomial (preferably degree less than 2) n storage algorithm that allows search in polynomial d time.

This stands in contrast to the **curse of dimensionality conjecture**, whose form we borrow from [Ind04]:

> If $d = \omega(\log n)$ and $d = n^{o(1)}$, any sequence of indexes built on a sequence of datasets $X_d \subset \Sigma_d$ allowing exact nearest neighbour search in time polynomial in d must use $n^{\omega(1)}$ space.

At the moment of writing a proof of above has not been found. We provide it here for pivot-based indexes.

1.4 Pivot-based indexing

Pivot-based indexing algorithms (for example AESA, MVPT, BKT....see [Cha01] and [Zez05]) rely on a selection of elements from X that are used as proxies for the rest of the dataset. That is, distances from all elements of X to the pivot elements are computed and then used to cut down computations through the triangle inequality:

Given pivot set $\{p_1 \dots p_k\}$, we compute the $n \times k$ array of distances

$$\rho(x, p_i),\ 1 \leqslant i \leqslant k,\ x \in X$$

This array serves as the index.

Given a range query with radius r and centre q, the k distances $\rho(q, p_1) \dots \rho(q, p_k)$ are computed so that $\rho(q, x)$ can be lower-bounded as follows:

$$\rho(q, x) \geqslant |\rho(q, p_i) - \rho(x, p_i)|$$

since this happens for any i, we can establish:

$$\rho(q, x) \geqslant \sup_{1 \leqslant i \leqslant k} |\rho(q, p_i) - \rho(x, p_i)|$$

It is useful to think of a new distance function, based on the k pivots:

$$\rho_k(q, x) := \sup_{1 \leqslant i \leqslant k} |\rho(q, p_i) - \rho(x, p_i)|$$

The fact $\rho(q, x) \geqslant \rho_k(q, x)$ can be used as a condition to discard all x satisfying:

$$\rho_k(q, x) > r$$

Therefore the algorithm consists of checking this condition, and if it is not satisfied, performing (the expensive) distance calculation to verify if

$$\rho(q, x) > r$$

Only if it is again not true do we know that the point should be returned in the query. This process is described in Algorithm 2: we call the new distance function ρ_k as *index.distanceK* to emphasize that is a function belonging to the index.

We will focus on range queries with pivot-based algorithms chiefly because they are easier to execute. At least in theory k-nearest neighbour queries can always be simulated by a range query with the radius set to the distance to the kth neighbour [Zez05].

As the iteration Algorithm 2 is happening on all the points of the dataset it may appear as this algorithm does not fit the framework of "regions". But it can always be considerate a degenerate case where the regions consist of singletons of points in the dataset plus the rest:

$$\Omega = (\cup_{x \in X} \{x\}) \cup (\Omega \backslash X)$$

```
Data: query, index
Result: queryResults
for each point in dataSet do
    if index.distanceK(point , query.center) < query.radius then
        if distance(point , query.center) < query.radius then
            append point to queryResults
        else
        end
    else
    end
end
return queryResults;
```

Algorithm 2: Querying a pivot-based index

This differs at least on a theoretical level from the decomposition presented in [Cha01] where an equivalence relation on the points of Ω is proposed:

$$\omega_1 \sim \omega_2 \iff \forall 1 \leqslant i \leqslant k, \rho(\omega_1, p_i) = \rho(\omega_2, p_i)$$

This equivalence relation is then made to induce a partition of the space. In Euclidian space these partitions are intersections of spheres, which for even small k will be single points.

Perhaps a more useful characterization also presented in [Cha01] is to think of the pivot based indexing as sending Ω to a different space and then performing a range similarity search in the new space:

$$(\Omega, \rho) \longrightarrow (l^\infty(k), l^\infty\text{-norm}) : \omega \longmapsto (\rho(\omega, p_i))_{i=1}^k)$$

The new space consists of sequences of reals of length k, with the max-distance also known as the l^∞-norm. This is akin to our musing at the beginning of the chapter where we admitted that having a function that sends every set of fingerprints to a number could be useful if the function had properties that allowed us to avoid a sequential scan of the *original space*.

As our cost model only counts distance computations in the original space, a range search in l^∞ is considered free. That is our results stand even under the generous assumption that it takes 0 time to perform a search in l^∞.

We will denote by C all the points of X satisfying

$$\rho_k(q, x) > r$$

that is all the discarded elements. Making C large is the primary way of cutting the cost of search in the setting of distance computations as dominating cost. Of course we can achieve this trivially with a very large number of pivots. This will defeat the purpose however as

$$\text{Cost of range search } = k + |X \backslash C|$$

The most often used solution is to keep adding pivots as long as it is found experimentally to decrease the cost of search. If k is small, on the order of $\log n$ (as often space limitations require), the most important component of cost becomes the size of C and this is where the choice of pivots would seem to matter. Various approaches to pivot selection have been investigated in [Bus03]. The empirical results seem to suggest that a moderate reduction in the number of distance computations can be achieved, although the relative improvement drops with increasing dimension.

There are numerous refinements on this basic approach to pivot-based indexes, but the underlying idea of using the triangle inequality together with the pre-computed distances is the same. Moreover [Cha01] argues that pivot-based indexes are one of only two types of metric space indexing algorithms, the other type being also closely related. Therefore investigating this barebones pivot index can be thought of as representative of a large number of actual implementations with the unecessary complications removed.

To recap, we are hoping that a judicious choice of the pivots (in particular their number k) will result in an *average* C that is big, preferably on the order of 99% of n (the size of X). Better yet would be to guarantee that $X \backslash C$ is no more than some fixed number, say 1000, irrespective of size of n. This way only the remaining elements will have to be totally searched - which will produce an efficient algorithm as long as we keep k reasonably small.

In situations involving the concentration of measure phenomenon this scenario cannot happen. In fact we will show that the exact opposite takes place. The set C will almost certainly be small, and most of the dataset will have to be exhaustively searched.

1.5 Approximate Search

A related problem to similarity search is approximate similarity search [Zez05]. The approximate version of say nearest neighbour search only requires that the element returned be within $1 + \epsilon$ distance of the "true" result:

Definition 1.7. Approximate Nearest Neighbour Search. Fix ϵ, $\eta > 0$ Let

$$\rho_{\mathrm{NN}}(q) = \inf \{\rho(q, x) | x \in X, x \neq q\}$$

represent the distance from q to its nearest neighbour in X. Then an approximate nearest neighbour of q in X is any element $\tilde{x}$ satisfying

$$\rho(q, \tilde{x}) \leqslant (1 + \epsilon)\rho_{\mathrm{NN}}(q)$$

with probability at least $1 - \eta$. An aproximate nearest neighbour search query asks for any such $\tilde{x}$, with apriori set confidence factor η.

There are some indications that approximate search is more efficient [Zez05]. However as pointed out in (e.g. [Lif07], [Bey99], [Pes00]), due to the concentration that many spaces exhibit, almost all points in a typical high dimensional

dataset lie at about the same distance to q so approximate search is of limited usefulness. We will look close at concentration in the next chapter.

Chapter 2

The concentration of measure phenomenon

High dimensional spaces pose a problem for algorithms: dimensionality appears to affect [Don00] whole classes of algorithms in optimization, numerical integration and database search. Cost estimates (running time) of solutions depend on dimension exponentially – something that has come to be known as the "curse of dimensionality". This term, although now liberally applied to any problem that seems to depend exponentially on dimension, was first used in [Bel61] to describe the unit hypercube $\mathbb{I}^d$ in d dimensions.

This space exhibits in a certain sense a growing sparseness. That is if we were to take a "small hypercube" neighborhood around a point expecting to capture a proportion r of the space $\mathbb{I}^d$, the side-lengths of the neighborhood will have to be $r^{1/d}$ ([Fri01]). For a given proportion, say 1%, this means $r = 0.1$ when $d = 2$ yet it grows to $r = 0.79$ when $d = 20$. Meanwhile the side lengths of the whole space remain one.

Another way of looking at this effect is to compare the volume of the unit ball and the unit hypercube. The volume of the unit hypercube is clearly 1, while the ball's is

$$\sim \frac{(2\pi)^{d/2}}{d!}$$

a value that goes to 0 with increasing dimension. This observation is used to argue that most of the points in the hypercube lie "near the edges".

Yet another approach is to spread points uniformly and ask what is the average distance between any given two. In the case of the hypercube it seems that no closed form expression for this number exists, but an approximate expression is $\sqrt{d/3}$ ([And76]). As there is nothing special about the centre except that it is somewhat closer to the "average" point, this shows that the mean distance between any two points grows at a rate of about $\sqrt{d}$ as a function of dimension. This is a heuristic argument often quoted for the hardness of function approximation in high dimensions ([Fri01]).

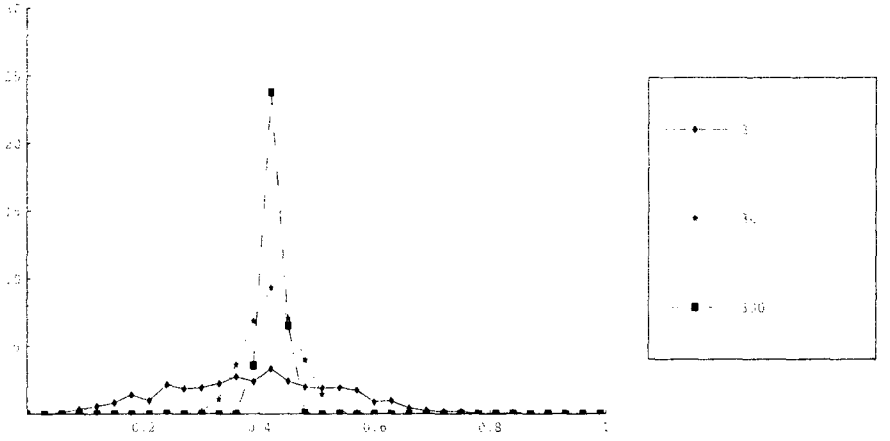

Figure 2.1: Simulation of Normalized Distance between two points in the hypercube for $d = 3, 30, 300$

It is legitimate to observe that the diameter of the hypercube is exactly $\sqrt{d}$ and thus a question that comes naturally to mind is why should this be called a curse of *dimensionality* when perhaps a more appropriate description is "curse of *hugeness*". After all a rectangle with side lengths k and $1/k$ exhibits a similar behaviour: the average distance between two points is about $k/3$ [Dun97] which will go to infinity with k. Yet it remains a low dimensional object that does not pose a problem for the aforementioned algorithms.

So is there some effect specific to high dimension? A simulation approach is to generate (pseudo) random points on cubes of various dimensions and take the resulting histograms as approximate probability densities. The result presented in Figure 2.1 is a more nuanced view of the distribution of distances in high dimensions: the histograms plotted show the distribution of distances for various dimensions d, normalized by $\sqrt{d}$. The average normalized distance tends to a constant, but it appears that the distribution is more *concentrated* with increasing dimension. A plot of the empirical standard deviation [She07] shows a decrease with dimension.

d	standard deviation
3	0.14
30	0.04
300	0.014
3000	0.004

This illustrates what is sometimes called a "benefit of dimensionality" [Don00], namely the concentration of measure phenomenon.

It is a well-studied topic of geometric analysis and is a much more powerful

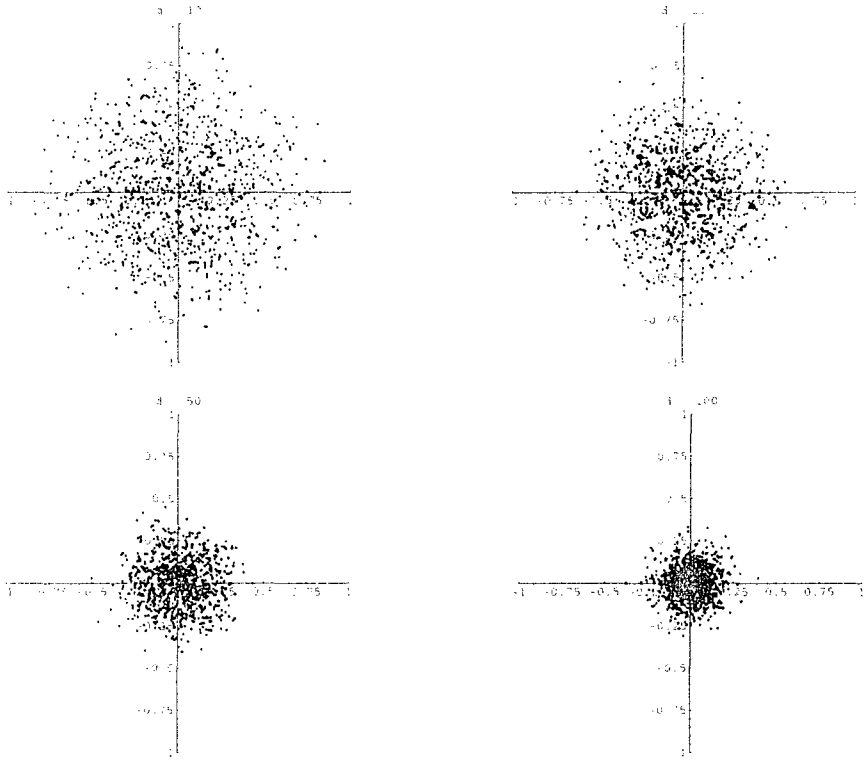

Figure 2.2: Projection of uniform samples on spheres of various dimensions d

statement than the one we have made about variances. As a preview we shall take a look at one more set of pictures – of the high dimensional unit sphere. In addition to being an appealing object it is naturally normalized with respect to distance: the maximal distance between two points remains 2 no matter the dimension.

In order to draw a (2-D) picture of an object we have to find a way of projecting it onto flat space. An orthodox choice would be to take any two coordinates, say the first couple, and plot the resulting figure. Doing it for several different pairs will give us different views, and thus maybe the whole object will be known. When the d-dimensional sphere is sampled according to the uniform measure, and projected onto the plane by say taking the first two coordinates the result is rather peculiar: most points concentrate near the centre of the image. A simulation for various values of d is provided in Figure 2.2.

This happens no matter which coordinates are chosen for the projection. The 2-D picture is the same: a small core in the centre, with nearly nothing around. If we were to attempt to calculate the diameter based on this picture

it will seem that the sphere, actually of constant diameter, is shrinking (with sample size kept constant).

Although a bit more difficult to imagine, when taking a look at the equator of the sphere, most points will lie a short distance from it. Again, it doesn't matter if the equator is the standard one – *any* equator will have this concentration around it.

The most convenient setting for finding "high-dimensional" objects is $\mathbb{R}^d$ but the phenomenon of concentration is phrased in terms of measure and distances, so it can be defined on metric spaces equipped with a probability measure. As usual, whenever we take the measure of a subset of the space we will restrict the discussion to measurable sets.

Definition 2.1. Given a metric space (Ω, ρ) equipped with (probability) measure μ, A_ϵ is the ϵ-neighborhood of $A \subset \Omega$, that is

$$A_\epsilon = \{\omega \in \Omega | \rho(\omega, a) < \epsilon \text{ for some } a \in A\}$$

We want to define a function α s.t. if $\mu(A) \geqslant 1/2$ then

$$\mu(A_\epsilon) \geqslant 1 - \alpha(\epsilon)$$

In a sense we will pick the best such α and call it the *concentration function*:

Definition 2.2. Given a metric space equipped with (probability) measure (Ω, ρ, μ) its concentration function $\alpha = \alpha_{(\Omega,\rho,\mu)}$ is defined as

$$\begin{aligned} &\alpha(0) = 1/2 \\ &\alpha(\epsilon) = \sup\{1 - \mu(A_\epsilon) | A \subset \Omega, \mu(A) \geqslant \tfrac{1}{2}\} \quad , \epsilon > 0 \end{aligned}$$

To put it less formally, we are trying to measure how much of the space remains after "fat" is added to a somewhat large set in the form of an ϵ neighborhood. When very little remains, we say that the concentration of measure takes place. Making the concept of "little" more precise, *normal* concentration of measure is considered to be taking place when $C, c > 0$ exist such that

$$\alpha(\epsilon) < Ce^{-cd\epsilon^2}$$

Where d is the (intrinsic) dimension.

2.1 Examples

Example 2.1. $\mathbb{R}^d$ with the Gaussian measure γ. The Gaussian measure is defined on the completion of the Borel σ-algebra. It is the generalization of the normal probability measure on $\mathbb{R}$. For any A in the above-defined σ-algebra of measurable sets,

$$\gamma(A) = \frac{1}{(2\pi)^{d/2}} \int_A e^{-\frac{\|x\|^2}{2}} d\lambda^d(x)$$

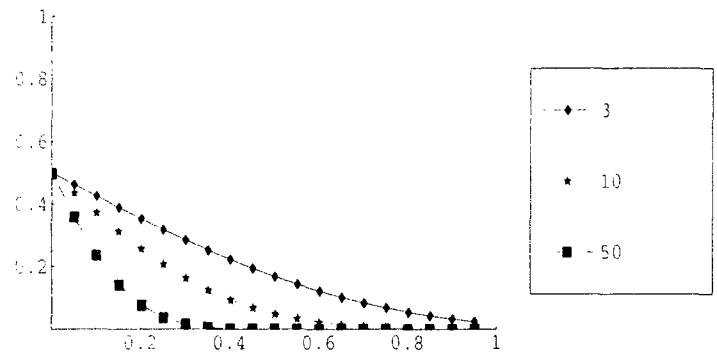

Figure 2.3: The *concentration functions* of various spheres

where λ^d is the d-dimensional Lebesgue measure.

The space $\mathbb{R}^d$with the measure γ and the standard Euclidean metric coming from the L_2 norm produces a concentration function bounded as follows:

$$\alpha(\epsilon) \leqslant \mathrm{e}^{-\epsilon^2/2}$$

This does not produce a normal concentration function. This is due to a certain stretching of the space that occurs as d grows, something that is not desirable from our perspective. In the upcoming example of Hamming cubes we will show explicitly how a distance measure can be "properly" normalized so as to produce a normal concentration function.

Example 2.2. The spheres $\mathbb{S}^d$ in $\mathbb{R}^{d+1}$.

Taken with the geodesic or Euclidian distance and the normalized invariant measure they produce a family of concentration functions bounded as follows [Led01]:

$$\alpha_d(\epsilon) \leqslant \mathrm{e}^{-(d-1)\epsilon^2/2}$$

In this case an exact expression for the concentration function is known [Ben00] p.282. as the half-sphere, of all subsets of measure at least 1/2 will always produce the smallest ϵ-neighborhood, no matter the ϵ. The measure of this neighborhood is given by

$$\left(\int_{-\pi/2}^{\epsilon} \cos^{d-2} x \mathrm{d}x\right) \Big/ \left(\int_{-\pi/2}^{\pi/2} \cos^{d-2} x \mathrm{d}x\right)$$

An estimation of this value can be arrived at via numeric integration. A plot of the resulting concentration functions, for several values of d, appears in Figure 2.3.

This example is particularly interesting as increasing dimension leads to increased concentration of measure phenomenon.

Definition 2.3. A sequence of spaces $(\Omega_d)_{d=1}^{\infty}$ a *normal Lévy family* [Mil86] if $C, c > 0$ exist such that

$$\alpha_d(\epsilon) < Ce^{-c\epsilon^2 d}$$

Thus it the same notion of what is a tight concentration function as above.

Example 2.3. The Balls $\mathbb{B}^d$. Taken with the Euclidean distance and the uniform probability measure (d-dimensional Lebesgue) form a normal Lévy family.

Example 2.4. The Hamming Cubes Σ^d. The Hamming cubes, as defined in Section 1.3, with the normalized distance and uniform measure form a normal Lévy family.

The concentration of measure can be equivalently described in terms of Lipschitz functions.

Definition 2.4. A function $f : \Omega \to \mathbb{R}$ is *1-Lipschitz* if

$$\forall x, y \in \Omega,\ |f(x) - f(y)| \leqslant \rho(x, y)$$

In general a function f is p-Lipschitz if for all x and y, $|f(x) - f(y)| \leqslant p\rho(x, y)$.

We note that $\rho(q, .)$ the function assigning to ω its distance to q is 1-Lipschitz.

In spaces that have a tight concentration α, Lipschitz functions will be nearly constant, and one candidate for this constant is a median value.

Definition 2.5. A median of function $f : (\Omega, \rho, \mu) \to \mathbb{R}$ is any number M satisfying:

$$\mu\{\omega | f(\omega) \leqslant M\} \geqslant 1/2 \text{ and } \mu\{\omega | f(\omega) \geqslant M\} \geqslant 1/2$$

This is a slight generalization of the usual concept of median of a set of numbers, only no attempt is made to make it unique. Discrete functions may very well have multiple valid values for M.

Theorem 2.6. *For a 1-Lipschitz function f defined on space (Ω, μ, ρ):*

$$\forall \epsilon > 0, \quad \mu\{\omega |\ |f(\omega) - M| > \epsilon\} < 2\alpha(\epsilon)$$

Proof. Fix $\epsilon > 0$. Set

$$A = \{\omega | f(\omega) \leqslant M\} \text{ and } B = \{\omega | f(\omega) \leqslant M + \epsilon\}$$

then

$$\begin{aligned} A_\epsilon &= \{\omega | \rho(\omega, a) < \epsilon \text{ for some } a \in A\} \\ &\subset \{\omega | f(\omega) - f(a) \leqslant \epsilon \text{ for some } a \in A\} \\ &\subset B \end{aligned}$$

Then by definition of α, we have

$$\mu(B) \geqslant 1 - \alpha(\epsilon)$$

The same argument can be applied to

$$\tilde{A} = \{\omega | f(\omega) \geqslant M\} \text{ and } \tilde{B} = \{\omega | f(\omega) \geqslant M - \epsilon\}$$

Since the probability of the entire space is 1, we rework the well-known

$$\mu(B \cup \tilde{B}) = \mu(B) + \mu(\tilde{B}) - \mu(B \cap \tilde{B}) \leqslant 1$$

into

$$\mu(B \cap \tilde{B}) \geqslant \mu(B) + \mu(B) - 1$$

thus

$$\mu(B \cap \tilde{B}) \geqslant 1 - \alpha(\epsilon) + 1 - \alpha(\epsilon) - 1 = 1 - 2\alpha(\epsilon)$$

□

Lipschitz functions allow us to formulate the concept of observable diameter, as illustrated by Figure 2.2, more rigorously.

Definition 2.7. Let $\kappa > 0$ be fixed. The κ-observable diameter of (Ω, ρ, μ), denoted obs − $\text{diam}_\kappa\Omega$ is defined as (M_f is the median of f):

$$\text{obs-diam}_\kappa\Omega = 2\inf\{D > 0 | \forall \text{ 1-Lipschitz } f, \mu\{\omega | \, |f(\omega) - M_f| > D\} \leqslant \kappa\}$$

We could reformulate the concentration of measure in terms of how fast the observable diameter shrinks to 0.

Actually calculating the exponential expressions for α is much harder: the sometimes complicated derivations can be found in [Led01] and [Mil86]. However certain details of such proofs are important to understand the role normalization of distance plays. In spaces where the diameter is allowed to grow, e.g. cubes of dimension d as at the beginning of the chapter or Hamming cubes with non-normalized measure

$$\rho(\boldsymbol{x}, \boldsymbol{y}) = \sum_{i=1}^{d} |x_i - y_i|$$

the concentration function may end up with a bound of the form:

$$\alpha(\epsilon) \leq Ce^{-c\epsilon^2}$$

that is, without dependence on d.

For purposes of similarity search however what matters is not the absolute value of the distance, but the proportion in relation to the space. That is, the issue is not so much if the range query is of radius 10 or 0.1 but what proportion of the space falls in a ball with this radius. So in an asymptotic analysis the aim is to keep the radius constant as dimension is growing – in "reasonable"

spaces this is accomplished through normalizing by the diameter. In general spaces another normalization, perhaps utilizing the expected distance between two points, is required.

We can illustrate the normalization by looking at two versions of the "Blowing Up Lemma" (e.g. [Pes07b])for the Hamming Cubes:

Theorem 2.8. *For a 1-Lipschitz function $f : \Sigma^d \to \mathbb{R}$ with respect to the non-normalized Hamming distance, we have that*

$$\forall \epsilon > 0. \quad \mu\{\omega | \ |f(\omega) - E(f)| > \epsilon\} < 2e^{\frac{-\epsilon^2}{d}}$$

If however the function is 1-Lipshitz w.r.t. the normalized Hamming distance, we have

$$\forall \epsilon > 0. \quad \mu\{\omega | \ |f(\omega) - E(f)| > \epsilon\} < 2e^{-d\epsilon^2}$$

The second part follows from the first since a 1-Lipshitz function f w.r.t. the normalized Hamming distance is $1/n$-Lipshitz w.r.t the non-normalized one. Therefore nf is 1-Lipshitz w.r.t. the non-normalized distance and so

$$\forall \epsilon > 0, \quad \mu\{\omega | \ |df(\omega) - dE(f)| > d\epsilon\} < 2\mathrm{e}^{\frac{-d^2\epsilon^2}{d}} = 2\mathrm{e}^{-d\epsilon^2}$$

We are making the broad claim that the exponential decrease in d is a broad phenomenon for spaces that are properly normalized for similarity search. So if no such dependence on d is observed as in Example 2.1 it may be just a matter of bad choice of distance.

2.2 Link to concentration of measure

We would like to demonstrate why in so many familiar spaces indexing, and in particular indexing using pivots is impossible. The use of concentration of measure in indexing is noted in [Pes00]. It relies on the observation that

$$\rho(\cdot, p) : \Omega \to \mathbb{R} : \omega \mapsto \rho(\omega, p)$$

is 1-Lipschitz for any p and in particular a pivot. Hence Theorem 2.6 can be applied to obtain a bound on the deviation from the median $M = M_p$ of function $\rho(\cdot, p)$:

$$\forall r > 0. \quad \mu\{\omega | \ |\rho(\omega, p) - M| > r\} < 2\alpha(r)$$

since p is a general element of Ω the statement holds individually for each pivot p_i. We combine these statements:

$$\forall r > 0. \quad \mu\{\omega | \sup_{1 \leqslant i \leqslant k} |\rho(\omega, p_i) - M_i| > r/2\} < 2k\alpha(r/2)$$

as the probability of the union can always be upperbounded, if roughly, by the sum of the probabilities. We note that no assumptions about independence are

used: the sequence (p_i) can be chosen in any way. We used $r/2$ so as to get rid of the M_j:

$$\forall r > 0, \quad \mu\{\omega | \sup_{1 \leqslant i \leqslant k} |\rho(\omega, p_i) - \rho(q, p_i)| > r\} < 2k\alpha(r/2)$$

Comparing this to the definition of C:

$$C = \{x | \rho_k(q, x) > r\}$$

it is apparent that the only difference between the set we are upperbounding and C is that one is defined over all of Ω and the other, just for X. We could introduce a set

$$\mathcal{C} = \{\omega | \rho_k(q, \omega) > r\}$$

and think of C as the observation of $\mathcal{C}$ under $\mu_\#$.

To restate the upperbound in terms of $\mathcal{C}$,

$$\forall r > 0, \quad \mu(\mathcal{C}) < 2k\alpha(r/2)$$

So in effect, assuming that $2k\alpha(r/2)$ is small, we have (roughly):

$$\mu(\mathcal{C}) \approx 0 \tag{2.1}$$

this is along the lines of [Pes00] yet we would like to find out what happens to C. The point here is that if something happens in Ω it will not necessarily hold *mutatis mutandis* in the dataset X.
The statement

$$\mu_\#(C) \approx 0 \tag{2.2}$$

describes a situation in which much of the dataset X needs to be totally searched. The more theoretical statement (2.1) refers to a situation where the query searches over all the elements of Ω in effect forcing $X = \Omega$, something we would like to avoid.

The probability measure $\mu_\#$ is a function of n, the size of the dataset. In fact the dataset is a sample (i.i.d) of size n from the probability metric space (Ω, μ) and hence $\mu_\# = \mu_n$ is also a random variable. Such complications show that it is important to fully describe the variables that underly the apparently simple statements.

If we let q be fixed, equation (2.1) in effect states

$$\mu(\mathcal{C}_{q, p_1, \ldots p_{k(n)}, r(n)}) \longrightarrow 0 \text{ as } n, d \longrightarrow \infty$$

where n, d are related as described in section 1.3 and

$$\mathcal{C}_{q, p_1, \ldots p_{k(n)}, r(n)} = \{x | \rho_k(q, x) > r\}$$

like before, with all variables made explicit. Similarly (2.2) states

$$\mu_n(C_{q, p_1, \ldots p_{k(n)}, r(n)}) \xrightarrow{P} 0 \text{ as } n, d \longrightarrow \infty$$

where convergence in probability is based on the sample, also known as our dataset X of size n.

This situation is further complicated by considering the query center q, the pivots p_i, the radius r and even k to be random variables. Here we will allow q to vary over all of Ω, while r can be thought of as proportional to the nearest neighbour in X to q. In the next section we will see that the median nearest neighbour distance in our properly normalized Lévy families is bounded away from 0 asymptotically.

2.3 Radius of queries in Lévy families

We have described above how we would like to normalize spaces so that the "average" distance between two points stays about the same. Here we will show why this also implies the typical radius of a query – which here we will assume to be the distance to the nearest neighbour of query centre – also behaves nicely.

Lemma 2.9 (M. Gromov, V.D. Milman). *[Gro83] Let (Ω, ρ, μ) denote a metric space with measure and α its concentration function. Then if $A \subset \Omega$ is such that $\mu(A) > \alpha(\gamma)$ for some $\gamma > 0$, it implies that $\mu(A_\gamma) > 1/2$.*

Proof. Assume not and let $B = A_\gamma^c$. Then $\mu(B) \geqslant 0$, which implies $\mu(B_\gamma^c) \leqslant \alpha(\gamma)$. But $\mu(A) \leqslant \mu(B_\gamma^c)$, a contradiction. □

Theorem 2.10. *Let $(\Omega_d, \rho_d, \mu_d, X_d)_{d=1}^\infty$ be a sequence of metric spaces with measure, forming a Lévy family together with i.i.d. samples X_d from Ω_d. Assume that $n = n_d = |X_d| = d^{o(1)}$. Furthermore, if M_d denotes the median value of $\{\rho_d(\omega_1, \omega_2) | \omega_i \in \Omega_d\}$ we assume that $M_d = \Theta(1)$, that is, for some fixed $c_1, c_2 > 0$. $\quad \forall d, c_1 < M_d < c_2$.*

Let $\rho_d^{(NN)}(\omega)$ denote the distance to the nearest neighbour of $\omega \in \Omega_d$ in X_d. Define m_d to be the median of $\rho_d^{(NN)}(\omega)$. Then there exists some $c_3 > 0$ and some D such that $\forall d \geqslant D$, $m_d > c_3$.

Proof. Assume the conclusion fails, then without loss of generality and proceeding to subsequence if necessary, $m_d \to 0$. By definition of m_d, we know that for any d,

$$\mu_d \left(\bigcup_{x \in X} B_{m_d}(x) \right) \geqslant \frac{1}{2}$$

It follows that

$$n_d \sup_{\omega \in \Omega_d} \mu_d \left(B_{m_d}(\omega) \right) \geqslant \frac{1}{2}$$

and so we can find for any d a point $\omega_d \in \Omega_d$ such that

$$\mu_d \left(B_{m_d}(\omega_d) \right) \geqslant \frac{1}{2n}.$$

If we denote by α_d the concentration functions of our spaces Ω_d we know by assumption the existence of $C, c > 0$ s.t.

$$\forall d, \alpha_d(\epsilon) \leqslant Ce^{-c\epsilon^2 d}.$$

Hence we can find d' s.t. $\alpha_{d'}(\gamma) < 1/2n_{d'}$ and $m_{d'} < c_1/8$ where $\gamma = c_1/8$ as well: this is since eventually,

$$Ce^{-c\gamma^2 d} < \frac{1}{2n_d} = \frac{1}{d^{\phi(1)}}$$

Then by lemma 2.9

$$\mu_{d'}\left(B_{m_{d'}}(\omega_{d'})\right)_\gamma \geqslant \frac{1}{2}$$

It then follows that

$$\mu_{d'}\left(\left(B_{m_{d'}}(\omega_{d'})\right)_\gamma\right)_\gamma \geqslant 1 - Ce^{-c\gamma^2 d'}$$

that is, since $m_{d'} + 2\gamma < 3c/8$,

$$\mu_{d'}\left(B_{3c_1/8}(\omega_{d'})\right) \geqslant 1 - Ce^{-c\gamma^2 d'}$$

But

$$\text{diameter}\left(B_{3c_1/8}(\omega_{d'})\right) \leqslant \frac{3c_1}{4}$$

So in $\Omega_{d'} \times \Omega_{d'}$ the measure of the set of points (ω_1, ω_2) for which $\rho_{d'}(\omega_1, \omega_2) < c_1$ is at least

$$\left(1 - \frac{1}{2n_{d'}}\right)^2,$$

obviously contradicting $M_{d'} > c_1$.

□

This result frees us from having to consider a radius that vanishes to 0 as n, d go to infinity. With this achieved, let us recap our goal: to show that most queries are slow, i.e. what we casually referred to as equation 2.2 takes place for most queries. What we in fact want is something along the lines of:

$$\text{median}_{q,p_1,\ldots,r}\left(\mu_n(C_{q,p_1,\ldots,p_{k(n)},r(n)})\right) \xrightarrow{P} 0 \text{ as } n, d \longrightarrow \infty, \tag{2.3}$$

where the median is taken over all the queries under consideration: any $q \in \Omega_d$ and any r at least as large as the distance to the nearest neighbour of q in X. As well, for each d and $n = n_d$ we would like to also consider all possible pivot-based index schemes (as long as k is within certain ranges we will specify later). Why the median? The aim is to show a certain behaviour for *many* queries: at least half is dramatic enough. So far we have shown, although the proof was just sketched (and with the detail about k left out) that

$$\text{median}_{q,p_1,\ldots,r}\left(\mu(C_{q,p_1,\ldots,p_{k(n)},r(n)})\right) \longrightarrow 0 \text{ as } n, d \longrightarrow \infty \tag{2.4}$$

Which is fine as long as $X = \Omega_d$ and hence the selection of a small dataset from an underlying large space is not taken into account. A more likely situation however is of a finite $X = X_d$ and an infinite (or at least much larger) $\Omega = \Omega_d$. What we need is to find out when statement 2.4 implies 2.3. To do so we will summon the powerful machinery of statistical learning theory.

Chapter 3

Statistical learning theory

Statistical learning theory has already been used in the analysis and design of indexing algorithms [Kle97] and is a vast subject. Instead of concerning ourselves with the whole, we will just focus on an important part: the generalization of the Glivenko-Cantelli theorem due to Vapnik and Chervonenkis.

In keeping with previous notation, X or (X) will denote a sample of n points $X_1, X_2 \ldots X_n$, sampled according to some unknown probability measure μ from a space Ω. The sampling is independent, thus in statistical jargon X is an i.i.d sample. Although μ is unknown, we can obtain useful approximations through X. In particular we can create a measure on the space Ω induced by X, the counting measure $\mu_n = \mu_n(X)$:

$$\mu_n(A) = \frac{|A \cap X|}{|X|}$$

In section 1.1 we also referred to it as $\mu_\#$. We will use the subscript n to emphasize the sample size and that in effect we are dealing with a whole class of measures. As they depend on the sample, these measures are also *random* in the sense that they are random variables.

To talk of a random variable approximating another one, it is necessary to describe what convergence of random variables is.

Definition 3.1 (Convergence of random variables)**.** It is said that a sequence of random variables Y_n converges to the random variable Y *in probability* if

$$\forall \epsilon > 0, \quad \lim_{n \to \infty} P\left(|Y_n - Y| > \epsilon\right) = 0$$

We denoted convergence in probability by $Y_n \xrightarrow{P} Y$. *Almost everywhere* convergence takes place if a stricter condition is met:

$$P\left(\lim_{n \to \infty} Y_n = Y\right) = 1$$

It is denoted by $Y_n \xrightarrow{a.e.} Y$

It is not too hard to see that the a.e. convergence implies convergence in probability. In fact the well known Law of Large Numbers has two versions, one for each type of convergence. We present what is in a sense just a variation on this law, the Glivenko-Cantelli theorem.

Theorem 3.2 (Glivenko-Cantelli). *Given sample* $(X) = X_1, X_2 \dots X_n$ *distributed i.i.d. according to* any *measure* μ *on* $\mathbb{R}^d$ *we have:*

$$\sup_{r \in \mathbb{R}} |\mu_n(-\infty, r] - \mu(-\infty, r]| \xrightarrow{P} 0$$

The convergence is taken with respect the product measure induced by the sample. This theorem provides a means of linking the empirical distribution

$$F_n(r) := \mu_n(-\infty, r]$$

and the actual distribution

$$F(r) := \mu(-\infty, r].$$

To paraphrase, it tells us that the empirical distribution converges "uniformly in probability" to the actual one. Incidentally, the almost sure convergence also takes place.

We can also see this statement in terms of the empirical measures of particular subsets converging to their true measure. This is made clear when we restate the theorem as follows:

$$\sup_{A \in \mathcal{A}} | \mu_n(A) - \mu(A) | \xrightarrow{P} 0 \tag{3.1}$$

where

$$\mathcal{A} = \{(-\infty, r] | r \in \mathbb{R}\}$$

which makes more apparent a path for extension: to generalize to other collections $\mathcal{A}$, on spaces (Ω, μ) other than the real line.

The question of when does (3.1) hold is not trivial as for a general μ and $\mathcal{A}$, the answer is not always positive:

Example 3.1. Let

$$\mathcal{A} = \{\text{countable unions of open intervals in } [0, 1]\}$$

that is, the collection of sets which are countable unions of open intervals, equipped with the inherited Lebesgue measure μ. Then for any sample $(X) = X_1, X_2 \dots X_n$ of points in $[0, 1]$, we can find $A \in \mathcal{A}$ consisting of say small disjoint intervals around the X_i so that $\mu(A) < 0.1$ and $\mu_n(A) = 1$ (by virtue of containing all the X_i). This results in $P(\sup_{A \in \mathcal{A}} | \mu_n(A) - \mu(A) | > 0.8) = 1$ for any n and hence no convergence to zero can take place.

The required convergence fails to take place because the class $\mathcal{A}$ is too rich. Of course it is also possible that with a different measure convergence will take place, but in a setting where μ is unknown it is best to make little if any assumptions about it. Extensions of (3.1) with a focus on finding the correct restrictions of $\mathcal{A}$ is a topic explored in [Vap98]. The bulk of the work is in finding appropriate measures of "size" of collections that can determine if (3.1) takes place, and if so at what rate.

To that purpose, we connect the potentially infinite collection $\mathcal{A}$ with the finite sample:

Given $(X) = X_1, X_2 \ldots X_n$ the collection $\mathcal{A}$ "colours" the sample as follows. Each $A \in \mathcal{A}$ will assign 1 to X_i if $X_i \in A$ otherwise it assigns 0. Hence we get a colouring of type

$$0, 1, 0, 0, 1, 0$$

which is an n-length encoding that might as well have been

white-black-white-white-black-white

We denote by $N(X)$ the *number* of such different encodings, when all $A \in \mathcal{A}$ are used to colour X. Clearly $N(X) \leqslant 2^n$. What is surprising is that in many situations despite a seemingly rich $\mathcal{A}$, we have $N(X) \ll 2^n$.

Definition 3.3. The *(random) entropy* of sample $X = X^{(n)}$ is just $\ln(N(X))$, denoted by $H(X)$. It follows that $H(X) \leqslant n \ln 2$. The expected value of $H(X)$, w.r.t. the sample distribution (in effect a product of μ's) is called the *entropy* of size n, denoted by $H(n)$:

$$H(n) = \mathrm{E}\left(H(X^{(n)}\right)$$

A result in [Vap98] states that (3.1) is equivalent to

$$\frac{H(n)}{n} \xrightarrow{n\to\infty} 0$$

This however is of little use if μ is unknown and does not guarantee *fast* (i.e. exponential) convergence.

Definition 3.4. The *growth function* is defined by

$$G(n) = \ln \sup_X N(X)$$

where the supremum is taken over all samples of size $n \geqslant 1$. It is independent of μ and choice of sample X.

There are two cases to consider for an upper bound for the growth function [Vap98]:

- for all n, $G(n) = n \ln 2$

- or, for the largest Δ such that $G(\Delta) = \Delta \ln 2$,

$$G(n) \begin{cases} = & n \ln 2 \quad \text{if } n \leqslant \Delta \\ \leqslant & \Delta(1 + \ln(n/\Delta)) \quad \text{if } n > \Delta \end{cases}$$

Which says that G can be either linear in n or logarithmic after some point Δ. This Δ is the so-called *VC dimension* and it turns out that its existence is precisely a necessary and sufficient condition for (3.1). Terminology in the literature also deems existence of Δ the case of *finite* VC dimension while if no such number exists $\mathcal{A}$ is said to be of infinite VC dimension.

So the challenge of going from (2.4) to (2.3) can be reduced to the calculation of this VC dimension for some $\mathcal{A}$. Since equations (2.4) and (2.3) are statements about sets $\mathcal{C} = \mathcal{C}_{q,p_1 \ldots p_{k(n)},r(n)}$, we would like to estimate the VC dimension of the collection of all sets of this form, for fixed k:

$$\mathcal{A} = \mathcal{A}_k = \{\mathcal{C}_{q,p_1 \ldots p_{k(n)},r(n)} | q \in \Omega, p_i \in \Omega, r > 0\} \tag{3.2}$$

Unfortunately this is a far from trivial exercise as it involves figuring out what possible forms can of all of these sets take.

3.1 Calculating a VC dimension

To show an easy example of calculating a VC dimension, we will come back to the $\mathcal{A}$ from Glvenko-Cantelli:

$$\mathcal{A} = \{(-\infty, r] \,|\, r \in \mathbb{R}\}$$

The calculation in this case can be done with "one's bare hands":
For $X = X_1 \in \mathbb{R}$, a sample consisting of a single observation, only two colourings have to be realized. The element $(-\infty, X_1 - 1] \in \mathcal{A}$ will paint X_1 as 0, while $(-\infty, X_1]$ will paint it as 1.

This elementary observation can be phrased as "any sample of size 1 is shattered by the class $\mathcal{A}$".

However if one takes a two element sample where without loss of generality $X_1 < X_2$ it is quite clear that no element of type $(-\infty, r]$ will contain X_2 but not X_1. In other words the colouring 0, 1 cannot be realized.

Therefore if we denote an n-size sample by $X^{(n)}$ we have that $N(X^{(1)}) = 2$ and $N(X^{(2)}) < 4$ for any choice of samples. The biggest size of the shattered sample is then 1, i.e. the VC dimension of $\mathcal{A}$ is 1.

More clever arguments (e.g. [Dud84], [Vap98], [Pes02]) are needed to prove that:

- The VC dimension of half-spaces in $\mathbb{R}^d$ is $d + 1$. Recall that a general hyperplane in $\mathbb{R}^d$ is defined as

$$\{x \in \mathbb{R}^d | (x, v) = b\}$$

for some $v \in \mathbb{R}^d$, $b \in \mathbb{R}$. Hence a general half-space is of the form

$$\{x \in \mathbb{R}^d | (x, v) \geqslant b\}$$

with the other "half" specified by multiplying v by -1.

- The VC-dimension of all open (or closed) balls in $\mathbb{R}^d$

$$\left\{ \{x \in \mathbb{R}^d | \ \|x - v\| < r\} \quad , \text{ where } v \in \mathbb{R}^d, r \in \mathbb{R} \right\}$$

is also $d + 1$.

- axis-aligned rectangular parallelepipeds in $\mathbb{R}^d$, i.e. sets of form

$$[a_1, b_1] \times [a_2, b_2] \times \ldots \times [a_d, b_d]$$

have a VC dimension of $2d$ [Dev97]

Given the existence of these results for $\mathbb{R}^d$, it is relatively easy to attempt to calculate the VC dimension of $\mathcal{A}_k$ from the previous section.

First note that

$$\mathcal{C} = \{\omega : \sup_i | \ \|\omega - p_i\| - \|q - p_i\| \ | > r\} = \left(\bigcap_i \{\omega : \ | \ \|\omega - p_i\| - \|q - p_i\| \ | \leqslant r\} \right)^c$$

This allows us to proceed through several simple steps:

- A set of the form

$$\{\omega : \ | \ \|\omega - p_i\| - \|q - p_i\| \ | \leqslant r\}$$

is a spherical shell, i.e. the intersection of one ball with the complement of a smaller ball having the same center. A coconut shell comes to mind as a physical example. We note that an intersection of shells is an intersection of sets from $\mathcal{A} \cup \mathcal{A}^c$ where $\mathcal{A}$ is the collection of all balls.

- Given a collection $\mathcal{A}$ the complement collection

$$\mathcal{A}^c = \{A^c | A \in \mathcal{A}\}$$

has the same VC dimension. For assume $\mathcal{A}$ shatters X and take any colouring of X. The opposite colouring, by putting 1 instead of 0 and 0 instead of 1, is produced by some $A \in \mathcal{A}$. Then $A^c \in \mathcal{A}^c$ produces the original colouring. The same argument can be applied in the other direction.

- The VC dimension of balls was quoted above as $d + 1$, hence the VC dimension of complements of balls is $d + 1$ as well. The VC dimension of the *union* of the two collections is

$$(d + 1) + (d + 1) + 1 = 2d + 3$$

This is a consequence of a general result [Vid03]:

Lemma 3.5. *If a collection $\mathcal{A}$ has VC dimension Δ_a and a collection $\mathcal{B}$ has VC dimension Δ_b (i.e. both are finite), the union $\mathcal{A} \cup \mathcal{B}$ has VC dimension at most $\Delta_a + \Delta_b + 1$*

We note that the above is for a union of the two *collections*. Another result, this time for intersection of sets is mentioned in [Blu89]:

Lemma 3.6. *For $(\Omega, \rho) = (\mathbb{R}^d, L^2)$, an upper bound on the VC dimension of $\mathcal{A}_{\cap_k}$, composed of k-fold interesections of elements of $\mathcal{A}$ of VC dimension Δ is*

$$2\Delta k \ln(3k)$$

Hence at last we can conclude that the VC dimension of $\mathcal{A}_k$ for the case $\Omega \subset \mathbb{R}^n$ is bounded by

$$2(2d+3)(2k)\ln((3)(2k)) = k(8d+12)\ln(6k) \tag{3.3}$$

where k is the number of pivots. This also allows us to conclude that convergence like in equation (3.1) takes place. Of course we only considered the case of $\mathbb{R}^d$ with the normal Euclidian metric.

We have already mentioned that the VC dimension of axis-aligned "boxes" is $2d$. It so happens that all balls with respect the L^∞ metric are such boxes, so we can obtain a bound on $\mathcal{A}_k$ for this metric as well.

Another example comes from considering the Hamming cube. In a similar argument to section 1.3, we observe that are 2^d points in a d-dimensional Hamming cube, and at most d different radii, so at most $d2^d$ different balls exist. We know from e.g. [Blu89] an upper bound on the VC dimension of finite collections:

Lemma 3.7 (Finite $\mathcal{A}$)**.** *If the class $\mathcal{A}$ is finite, its VC dimension is bounded by $\log_2 |\mathcal{A}|$.*

Disregarding the small leftover term, the VC dimension for balls in the Hamming cube is about d.

Summarizing our three examples:

Theorem 3.8. *Let us denote by Δ the VC dimension of collection $\mathcal{A}_k$ as defined in equation (3.2). Then various upper bounds on Δ, depending on the space, are as follows:*

- *For $(\mathbb{R}^d, L^2)$, $\Delta \leqslant k(8d+12)\ln(6k)$*
- *For $(\mathbb{R}^d, L^\infty)$, $\Delta \leqslant k(16d+4)\ln(6k)$*
- *For (Σ^d, ρ) (normalized or not), $\Delta \leqslant k(8d + 8\log_2 d + 4)\ln(6k)$*

The case of the general metric space is harder, for the VC dimension of balls is dependent on the "intrinsic" dimension of the space. In [Bou] various capacity measures are calculated for general balls in general metric spaces. Crucially, the capacity measures are stated in terms of the covering numbers of the space (Ω, ρ) which in turn relate to the Assouad dimension discussed above. What is lacking is a computationally feasible procedure of estimating this dimension for an arbitrary Ω, through a random sample X. This underscores why the Chávez intrinsic dimension is so attractive, as its easy formulation allows us to estimate it accurately with moderate sample sizes (using for example the Hoeffding inequality in section 3.2). What is lacking however is structural results either linking capacity measures to the Chávez intrinsic dimension directly or somehow relate it to the Assouad dimension.

Even if we are willing to sacrifice the generality of our Ω and stick to $\mathbb{R}^d$ (after all a lot of spaces can be made to sit even if unnaturally in *some* $\mathbb{R}^d$) a problem remains. A finite VC dimension gives us convergence, but nothing about the rate of convergence has been said so far. A worrying sign comes from trying out some reasonable values for d and k, e.g. $d = 20$, $k = 50$. The bound on VC dimension then becomes 49,053 – a much larger number than say the VC dimension of spheres of the underlying Euclidian space. How large does n have to be for the convergence result to have any value?

3.2 Rates of convergence

In addition to the convergence specified by the Law of Large Numbers various inequalities exist to describe the rate of this convergence. An example of that is :

Theorem 3.9 (Hoeffding inequality). *Suppose we have a sequence of i.i.d random variables* $X_i \in [0, 1]$, $1 \leqslant i \leqslant n$. *Then for all* $\varepsilon > 0$,

$$P\left(\left|\bar{X} - \mathrm{E}\right| \geqslant \varepsilon\right) \leqslant 2\exp(-2n\varepsilon^2)$$

where E *is the expected value of any* X_i.

This is more precise than the usual statement of the Law of Large numbers:

$$P\left(\left|\bar{X} - \mathrm{E}\right| \geqslant \varepsilon\right) \overset{n\to\infty}{\longrightarrow} 0$$

We shall go back to Glivenko-Cantelli once more. In terms of $\mathcal{A}$ the theorem tells us:

$$\text{for all } \varepsilon > 0,\ P\left(\sup_{A\in\mathcal{A}} \mid \mu_n(A) - \mu(A) \mid > \varepsilon\right) \overset{n\to\infty}{\longrightarrow} 0$$

With $\varepsilon > 0$ considered fixed, we can ask how fast the expression on the left goes to zero. A consequence of the Kolmogorov-Smirnov Law [Vap98] is :

$$P\left(\sup_{A\in\mathcal{A}} \mid \mu_n(A) - \mu(A) \mid > \varepsilon\right) < 2\exp(-2n\varepsilon^2)$$

This is perhaps unsurprisingly very similar to the Hoeffding inequality. This convergence is what we will call exponential or fast in keeping with [Vap98].

The extension ([Vap98] p.148) to the case of any $\mathcal{A}$ of finite VC dimension Δ is as follows:

Theorem 3.10. *[Generalization of Glivenko-Cantelli] For a collection $\mathcal{A}$ of subsets of Ω, of finite VC dimension Δ, and any measure μ on Ω, we have that for any $\varepsilon > 0$,*

$$P\left[\sup_{A\in\mathcal{A}} \mid \mu_n(A) - \mu(A) \mid > \varepsilon\right] < 4\exp\left[\left(\frac{\Delta(1+\ln(2n/\Delta))}{n} - \left(\varepsilon - \frac{1}{n}\right)^2\right) n\right].$$

The convergence is eventually like

$$\exp(-\varepsilon^2 n),$$

which is again a fast rate of convergence.

A somewhat different form is quoted in [Dev97]:

$$P\left[\sup_{A\in\mathcal{A}} \mid \mu_n(A) - \mu(A) \mid > \varepsilon\right] < 8\exp\left(\Delta(1+\ln n/\Delta)\right)\exp\left(\frac{-n\varepsilon^2}{32}\right).$$

Except the assumption that $\mathcal{A}$ is of finite VC dimension no other information is used. In fact since no information about the measure μ is incorporated the left side can be replaced by its supremum taken over all possible probability measures on the underlying space Ω.

Depending on the specific case, tighter bounds may be possible, using other capacity concepts than the VC dimension and a priori knowledge about the measure μ [Vap98], [Men03].

As an example, we will go back to covering numbers. We can turn any $\mathcal{A}$ into a metric space by using the *exclusive or* under some measure. Using the counting measure, an exclusive or is just:

$$A \oplus_n B := \mu_n(A \cup B - A \cap B)$$

As a result we can talk of the minimal ε-net $\mathcal{B}_n(\mathcal{A}, \varepsilon)$ induced by μ_n, or the minimal ε-net $\mathcal{B}(\mathcal{A}, \varepsilon)$ induced by μ. If the VC dimension of $\mathcal{A}$ is known to be finite, the following bound holds (cf e.g. [Pes07b]):

$$P\left[\sup_{A\in\mathcal{A}} |\mu_n(A) - \mu(A)| > \varepsilon\right] < 8\mathrm{E}_\mu\left[\mathcal{B}(\mathcal{A}, \varepsilon/8)\right]\exp\left(\frac{-n\varepsilon^2}{128}\right)$$

Then perhaps it is unsurprising that there is a relationship between covering numbers of $\mathcal{A}$ and its VC dimension (cf e.g. [Pes07b]):

$$\log \mathcal{B}_n(\mathcal{A}, \varepsilon) \leqslant \Delta(\log n + 1 - \log \Delta)$$

To free ourselves from the particular choice of sample (which depends on the unknown μ), we can introduce a new concept

Definition 3.11. The *metric entropy* of $\mathcal{A}$ is [Dev97] the function of ε defined as:

$$N(\mathcal{A}, \varepsilon) = \sup_n \sup_{\mu_n} \mathcal{B}_n(\mathcal{A}, \varepsilon)$$

It then folows that

$$P\left[\sup_{A \in \mathcal{A}} |\mu_n(A) - \mu(A)| > \varepsilon\right] < 8N(\mathcal{A}, \varepsilon/8) \exp\left(\frac{-n\varepsilon^2}{128}\right)$$

A natural restatement of these results is to ask how big does n have to be for the expression on the left to be less than some $\eta > 0$. Solving for η we get

$$n \geqslant \frac{128}{\varepsilon^2}\left(\log N(\mathcal{A}, \varepsilon/8) + \log\frac{8}{\eta}\right)$$

The use of some technical inequalities (cf e.g. [Pes07b]) yields a similar result in terms of VC dimension:

$$n \geqslant \frac{128}{\varepsilon^2}\left(\Delta \log\frac{2\mathrm{e}^2}{\varepsilon} + \log\frac{8}{\eta}\right). \tag{3.4}$$

Chapter 4

Linking theoretical and empirical observations

In this chapter we are able to tie everything together for a conclusion about indexability. Essentially, we we are now able to go from the theoretical observation described by (2.4) to (2.3). Let us be clear that we are not proving that for a given dataset all queries are linear for all pivot based indexes. There are several points to keep in mind:

- Convergence in (2.3) is in probability: it is conceivable that for some rare datasets indexing is easy.
- The analysis is asymptotic, i.e. for large n and d, so we do not dispute that datasets of small dimension are indexable.
- We assume certain properties of our underlying spaces (Ω_d) : concentration of measure has to take place, and the VC dimension of balls is small. We think it is not unreasonable to assume these are commonly present, but again they are probably not universal.

Let us walk through to the main result, on the way establishing a bound for k. From the previous section, we know that only for large values of n will empirical measures be close (up to ε) to actual measures with high likelihood (1-η). The lower bound on n then naturally depends on ε, η but also on the VC dimension Δ of the collection of sets $\mathcal{A}_k$ we are trying to measure.

Let us fix ε and η, e.g. for our purposes both can be 1/2. Then by pooling all constants, including ε and η but not Δ we can rewrite expression (3.4) as:

$$n \geqslant M_1 \Delta \tag{4.1}$$

where $M_1 > 1$. What we would like to avoid is to have the right part of this expression grow linearly in n. We know an upper bound on Δ depends on k and d as established in Theorem 3.8. As our concern is for asymptotic behaviour we

will simplify this bound to $kd \ln k$. We will generalize to define (an) acceptable asymptotic behaviour for the VC dimension of balls: linear in d.

Now we would like to be able to conclude that the database sizes we encounter are large enough for Theorem 3.10 to hold (that is the right side of 4.1 is asymptotically below n). The number of pivots k can potentially ruin the day. There are indexing schemes, like AESA [Zez05] or the above-mentioned Orchard's algorithm 1.1 where $k = n$. In such cases, what we have shown above is that samples much bigger than n are needed for Theorem 3.10 behaviour. Alas (or fortunately?) our sample is all of X and not more – so for large k we are not able to make the desired conclusions.

There are good arguments as to why n^2 storage is not practical: in a database consisting of an enormous amount of records is there really space for storing an index that is a *square* of that very large number? (keeping Google and the Internet as a case in point). It has even been argued that under certain assumptions the optimal number of pivots is on the order of $\ln n$ and that even this number is rarely reached in practice [Cha01]. Therefore it would not be unrealistic to restrict the analysis to pivot size much smaller than n. For example we could require

$$k = O(\log n)$$

We also have to keep in mind that the query algorithm we have described requires at least k distance computation so if $k = n$ the query is linear in n which for our purposes is no better than a linear search.

A similar discussion applies to dimension d: although it is possible to consider situations when the growth of d is on the order of n, more reasonable are situations in which it grows somewhat slower, as argued in section 1.3.

Combining $d = o(n)$ with the even stronger condition on k, we can conclude that:

$$\Delta = o(n)$$

and hence asymptotically we know that the right side of expression (4.1) falls (much) under n. Since the aim is only for $\Delta = o(n)$, the requirement on k can be weakened somewhat:

$$k = o(\frac{n}{d})$$

This will be our standing assumption for k.

Hence we have a result of the form:

$$P(\sup_{\mathcal{C}} |\mu_{\#}(C) - \mu(\mathcal{C})| > \varepsilon) < \eta \tag{4.2}$$

This is equivalent to saying that if (2.4) holds then so does (2.3).

As we have not made it precise in chapter 2 as to why (2.4) holds, we will come back to the concentration of measures in spaces Ω_d. Restating (2.1) using an exponential concentration of measure function and applying the reasoning of Section 2.2:

$$\mu(\mathcal{C}) \leqslant M_2 k \mathrm{e}^{-d\varepsilon^2}$$

We will sacrifice a certain number of $\mathcal{C}$ so that r can be considered a constant (see section 2.3): we will proceed with at least half the queries having radius r above a constant independent of d. Hence the quantities that vary in d are n and k. Since d is superlogarithmic in n,

$$\begin{aligned} & \forall c > 0,\ d > c \log n \\ \Rightarrow \quad & \forall c > 0,\ \exp(-d) < \exp(-c \log n) \\ \Rightarrow \quad & \forall c > 0,\ \exp(-d) < cn \end{aligned}$$

So $e^{-dr^2} = o(n)$, and hence

$$\mu(\mathcal{C}) = o(n)$$

so not only does $\mu(\mathcal{C}) \to 0$, we have a bound on the convergence as well. In fact this holds for at least half the queries simulataneously so:

$$\text{median} \sup_{\mathcal{C}} \mu(\mathcal{C}) = o(n)$$

This, combined with equation (4.2) gives us the main result.

4.1 Main result

Theorem 4.1. *Consider a sequence of metric spaces (Ω_d, ρ_d) where $d = 1, 2, 3, \ldots$ and the VC dimension of closed balls in space (Ω_d, ρ_d) as a function of d is $O(d)$. Furthermore the metric spaces come equipped with Borel probabilities μ_d such that for fixed $C, c > 0$ the sequence of concentration functions of $(\Omega_d, \rho_d, \mu_d)$ satisfies*

$$\forall \epsilon > 0, \quad \alpha_d(\epsilon) \leqslant C e^{-c\epsilon^2 d}$$

For each d an i.i.d. sample X_d of size n_d is selected from Ω_d, according to μ_d. The sample size n_d is such that $d = \omega(\log n_d)$ and $d = n_d^{o(1)}$ if d is expressed as a function of n_d. We treat X_d as a dataset on which to build an index for similarity search. The index built is a pivot index using k pivots, where

$$k = o(n_d/d).$$

We fix any small $\varepsilon, \eta > 0$ as desired. Suppose we only ask queries whose radius is equal or greater to the distance to nearest neighbour of query centre $q \in \Omega_d$ in X_d.

Then there exists a D such that for all $d \geqslant D$, the probability that* at least half *the queries on dataset X_d take less than $(1 - \varepsilon)\mu_d$ time is less than η.

Furthermore, if we allow the likelihood η to depend on d, we can pick η_d so that the above holds true and

$$\lim_{D \to \infty} \prod_{d=D}^{\infty} (1 - \eta_d) = 1$$

We emphasize that this result is independent of the selection of pivots.

We have done all the calculations to show this theorem except demonstrate the limit. We will nevertheless make a quick summary. First of all, we will refer to Theorem 3.8 and Chapter 2 to provide us examples of spaces that meet all of the above requirements. It must be said that these families of spaces are not peculiar counterexamples but are commonly used in indexing today - except of course that only the low dimensional ones are successfully being indexed.

We showed above how the results of Section 2.2 use the Concentration of Measure to deduce the "theoretical version" of linear querying. We could only show it for all range queries above some fixed $r > 0$, but this is not an obstacle as in all spaces that we talk of distances are scaled appropriately. In particular the expected distance to nearest neighbour approaches a constant as $d \to \infty$. So by adding this requirement of minimum radius we are merely disregarding queries that return no elements except perhaps the query center itself. To link this to querying in a discrete structure that is a dataset, we used the previous chapter's results where the assumptions included that the VC dimension of balls is small. The result then is that querying in these spaces is linear for all high dimensions.

We ignored k from calculating the runtime as it becomes asymptotically insignificant with increasing dimension. A k larger than that stipulated in the main result will matter, but may trivially give linear runtime in n_d (e.g. if k is linear in n_d). What remains unanswered here is the behaviour for k in between $o(n_d/d)$ and $\Theta(n_d)$.

To derive the asymptotic probability, equation 3.4 can be used to obtain the lower bound on η:

$$\eta \geqslant \exp\left(\Delta \log\left(\frac{2e^2}{\varepsilon}\right) + \log 8 - \frac{\varepsilon^2 n}{128}\right)$$

which as an asymptotic function of d transforms into

$$\eta = \exp(-d^{\omega(1)})$$

Assuming independent choices of the datasets X_d, and assuming that for each d the probability of an event is at least $1 - \eta_d$, we will estimate the quantity

$$\prod_{d=D}^{\infty}(1-\eta_d)$$

As η_d goes to 0 at least as fast as e^{-d}, it is enough to show that

$$\lim_{D\to\infty}\prod_{d=D}^{\infty}(1-e^{-d}) = 1$$

to have

$$\lim_{D\to\infty}\prod_{d=D}^{\infty}(1-\eta_d)=1$$

as well.

Observing [Ash71] that for any sequence $0 \leqslant \eta_d \leqslant 1$,

$$1-\sum_{d=1}^{N}\eta_d \leqslant \prod_{d=1}^{N}(1-\eta_d) \leqslant \exp\left(\sum_{d=1}^{N}-\eta_d\right)$$

we can extend this, for any D to:

$$1-\sum_{d=D}^{\infty}\eta_d \leqslant \prod_{d=D}^{\infty}(1-\eta_d) \leqslant \exp\left(\sum_{d=D}^{\infty}-\eta_d\right)$$

We know that as a geometric series,

$$\sum_{d=D}^{\infty}\mathrm{e}^{-d}=\frac{\mathrm{e}^{-d}}{1-\mathrm{e}}$$

Hence we can conclude that

$$\lim_{D\to\infty}\prod_{d=D}^{\infty}(1-\mathrm{e}^{-d})=1$$

□

The examples given above of the various spaces exhibiting normal concentration of measure should convince the reader that it is real and widespread, though of course not universal. For the VC dimension of balls to depend linearly on d is a more vague requirement as the definition of dimension for metric spaces for the purposes of similarity search is an unresolved problem. It is however clearly impossible to take out dimension from a discussion of the curse of dimensionality, which is after all what we have shown, caveats notwithstanding.

This is not the first lower bound result for pivoting algorithms or indexes in general. There has been research for some time into lower bounds for various cases, though most often for approximate algorithms e.g. [Chak]. A specific lower bound for pivot-based indexing already mentioned above is that of [Cha01b]:

$$\tilde{d}\log n$$

It is not asymptotic, and assumes that $k=\Theta(\log n)$. Furthermore the pivot selection is assumed to be random, as opposed to our bound that is applicable to *any* pivot selection technique.

If we are to apply our restrictions on d (assumed to be asymptotically equivalent to $\tilde{d}$), the result is that

$$\tilde{d}\log n=\omega(\log^2 n)$$

$$\tilde{d}\log n=n^{o(1)}$$

So if we are to use these results asymptotically they do not provide strong lower bounds on the cost of similarity search.

Chapter 5

Indexing experiments

We have borrowed basic code libraries from [SISAP] and followed up on the techniques and datasets in [Bus03] to simulate variations of the *incremental pivot selection* building algorithms. The aim of this section is to test-drive different approaches to selecting pivots to demonstrate the challenges that arise from the curse of dimensionality.

The construction of a pivot index is essentially a matter of finding the "right" pivots. It is well known that a non-random selection of pivots can improve similarity search. The most commonly used technique [Bus03] is called incremental selection. The broad aim is to maximize the average distance $\rho_{p_1,\ldots,p_i}$ for each $1 \leqslant i \leqslant k$. That is, we are doing a greedy search: at each step we find a pivot that maximizes average $\rho_{p_1,\ldots,p_i}$ and add it to our pivot list.

To estimate the average value of $\rho_{p_1,\ldots,p_i}$, that is:

$$\mathrm{E}_{\Omega\times\Omega}\left(\rho_{p_1,\ldots,p_i}(\omega_1,\omega_2)\right)$$

we use a sample of A pairs from X, and by averaging $\rho_{p_1,\ldots,p_i}(x,y)$ over all the pairs we calculate an estimate. In practice then we are not maximizing distance $\rho_{p_1,\ldots,p_i}$ but an A-sample version of it. Since our spaces have bounded diameter, calculating sample size in principle is not complicated, e.g. using (3.9). The curse of dimensionality interferes however by requiring the estimation to be ever more precise for higher dimensions. Due to space and time limitations tradeoff between large values of A and the other sample - that of pivots - must be found experimentally.

The sample of pivots is made at each step – N random choices for the next candidate pivot are made, and the one that maximizes the (A-sample) distance gets added. Once we reach k pivots, we calculate the distances from X to the set of pivots. That distance matrix, together with the pivot list constitute the index which can be used for similarity search as already described in Algorithm 2. This construction procedure is summarized in Algorithm 3.

A single experiment consists of constructing this index for a given dataset, and then running thousands of queries to calculate the average number of distance computations required to answer a query. As queries with larger radii are

Data: dataset, k
Result: pivot-list, distance-matrix
Sample A pairs from dataset;
pivot-list: {} ;
for *i in 1 to k* **do**
    Select N candidates for a pivot;
    Find pivot maximizing distance based on pairs;
    Add selected pivot to pivot-list;
end
calculate distance-matrix;
return pivot-list, distance-matrix;

Algorithm 3: constructing a pivot-based index

very slow, the experimentation, as in [Bus03], is restricted to queries that return on the order of 10 to 100 elements out of a typical dataset size of 100,000.

We have expanded on incremental pivot selection procedure by selecting pairs of points from X that lie close to each other. The underlying idea is to concentrate on points that are hard to separate because they are so close together.

We will not explicitly analyze the costs of doing a random sample versus this "smart" sample since it is clearly very expensive to arrive at the smart sample: if no such list exists a priori one would need to compile it by making an index (which incidentally was our original goal) and then making small radius range or nearest neighbour queries. Situations are however conceivable where such a list can be obtained at low cost.

In Figure 5.1 we have plotted results for a dataset coming from compressed image data from NASA [DIMACS]. This dataset has 40000 20-dimensional vectors, equipped with the Euclidian L^2 metric. Incidentally its estimated Chávez intrinsic dimension is 3.9. This dataset presented certain challenges as results lacked the stability of other datasets, e.g. for pivot size 20 the difference between worst and best result for the same parameters can be as high as 14 distance computations. This uncertainty remains even after varying the pivot sample size N. In our experiments we leave N fixed at 40 since varying it doesn't change results either, not just their variance. On the other hand this difference of 14 represents only about 4% so it doesn't affect the broad conclusions. The experiment was run using different sample sizes for the incremental selection technique: 5000, 50000, and 10000. Testing was done by averaging 5000 queries that return an average of 40 points (that is 0.1% of the dataset). The overall result is that indexes based on the different samples all achieve about the same performance, with the optimal number of pivots around 60. The conclusion is that significant savings in construction time can be achieved by reducing sample size. Query time and index size however remain the same for all 3 cases.

Figure 5.2 illustrates what happens when we select "smart" samples instead of random A pairs as before. All sample sizes are 5000, testing done by averag-

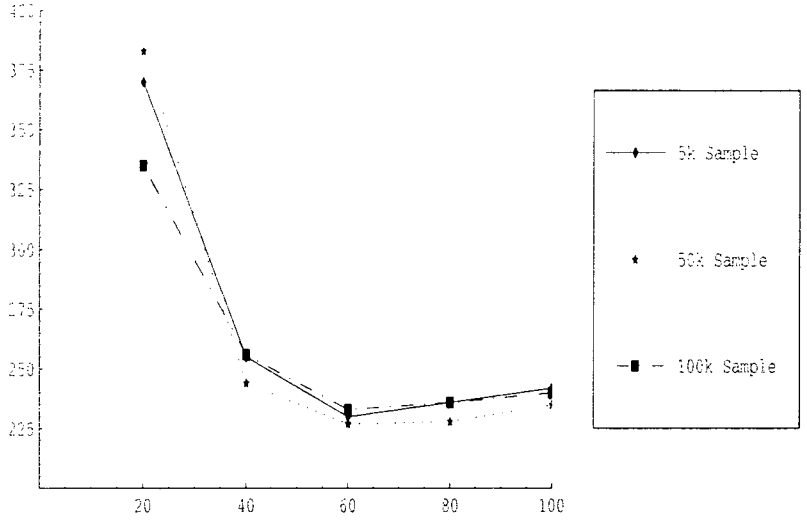

Figure 5.1: Distance computations vs number of pivots for the NASA dataset

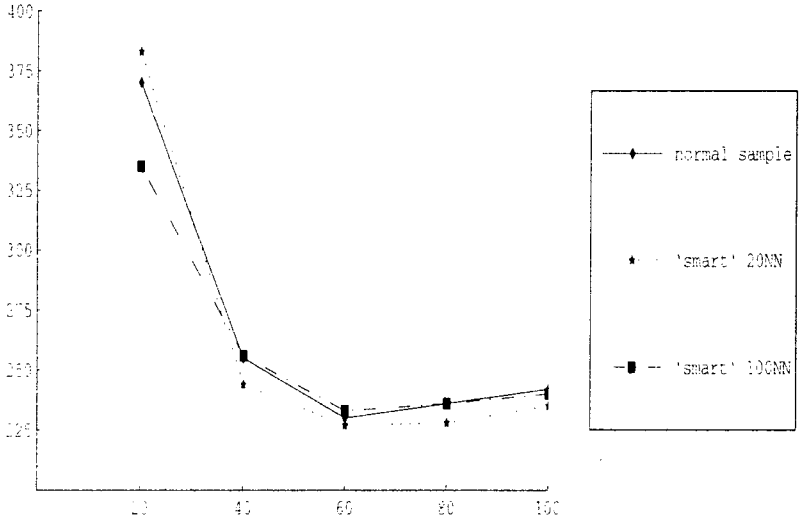

Figure 5.2: Distance computations vs number of pivots for the NASA dataset

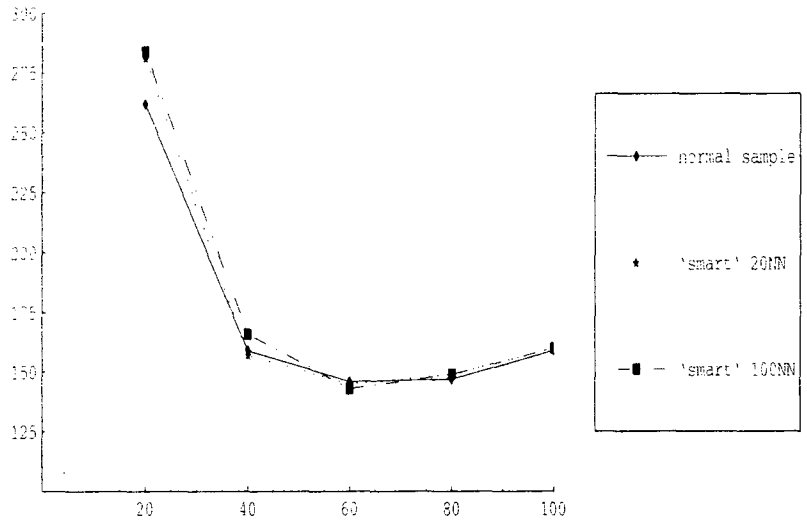

Figure 5.3: Distance computations vs number of pivots for an 8-dimensional uniform cube dataset

ing 5000 queries. We produce two sets of smart samples: one based on running 20-nearest-neighbour (20NN) queries, where 5000 random queries are made and their 20th nearest neighbour is selected as the second element of the pair, thus producing 5000 pairs. The second sample is produced similarly using 100NN queries. Here we notice that the choice of which smart sample to select matters somewhat, as the 20NN leads to a slightly better performing index. The difference is however small and must be balanced with the expense of construction for as we mentioned before we need to build a preliminary index to create the smart sample.

We repeat the smart sample experiments on a synthetic dataset: the uniform cube of dimension 8. The results here show even less difference between random and smart samples of A pairs. In addition variation is very low, with repeat experiments giving virtually identical results. Perhaps more interesting is to do simulations where the dimension d rises. Unfortunately hardware and software limitations severely limit the number d, with our setup running out of memory at $d = 14$. We can point however to results by [Bus03] where performance for optimal number of pivots is plotted against various values of d from 2 to 14, for uniformly distributed cubes. The results clearly demonstrate exponential growth in d no matter the pivot selection technique. In other words, so far, no matter how contrived the building algorithm for pivot-based indexing we have yet to find one that breaks the curse of dimensionality. Our proposal to vary the sample selection for the incremental technique performs just like the rest in that respect.

In summary, it is impossible to prove the curse of dimensionality by simulations alone as we could always ask ourselves maybe another index building algorithm exists. This is why our theoretical result is so powerful: we are able to make the conclusion for *any* pivot-index building method. We are leaving thus several options for the practitioner: to try better indexes, control intrinsic dimension, or limit the size of the dataset.

The solution so far seems to have been to simply mostly avoid high-dimensional datasets. Bar some breakthrough in indexing technology the advice then for those dealing with truly high-dimensional data is to limit the dataset size in relation to available processing power: this means no exponential growth in size is to be allowed.

Bibliography

[And76] Anderssen, R. S., Brent R.P., Daley, D.J., Moran, P. A. P. (1976) Concerning $\int_0^1 \cdots \int_0^1 x_1^2 + \cdots + x_k^2 \, dx_1 \cdots dx_k$ and a Taylor series method SIAM J. Applied Mathematics, Vol. 30, No. 1, January 1976

[Ash71] Ash, Robert B. (1971) Complex Variables. Academic Press

[Bel61] Bellman, Richard (1961) Adaptive Control Processes: A Guided Tour Princeton University Press

[Ben00] Benyamini, Y., Lindenstrauss, J. (2000) Geometric Nonlinear Functional Analysis: V. 1 AMS Colloquium publications

[Bey99] Beyer, K. Goldstein, J., Ramakrishnan, R., Shaft, U. (1999) When is "Nearest Neighbour" meaningful? Lecture Notes in Computer Science, vol.1540, pp217–235

[Blo00] Blockeel, H., Raymond, K. (2000) Web Mining Research: A Survey SIGKDD: SIGKDD Explorations, ACM, Vol. 2

[Blu89] Blumer, A., Ehrenfeucht, A., Haussler, D., Warmuth, M.K. (1989) Learnability and the Vapnik-Chervonenkis dimension *Journal of the ACM*, Volume 36, Issue 4. 929–965

[Bou] Bousquet, O., Luxburg, U. (2004) Distance-Based Classification with Lipschitz Functions. The Journal of Machine Learning Research, v.5, pp 669-695.

[Boz] Bozkaya, T., Ozsoyoglu, M (1999) Indexing large metric spaces for similarity search queries. ACM Transactions on Database Systems, Volume 24 , Issue 3, 361 – 404

[Bri95] Brin, S. (1995) Near neighbor search in large metric spaces Proceedings of the 21st Conference on Very Large Databases (VLDB95), 574584

[Bus03] Bustos, B., Chávez, E., Navarro, G (2003) Pivot selection techniques for proximity searching in metric spaces. Pattern Recognition Letters, 24-14, 2357 2366.

[Cap99] Capra, L. (1999) Il problema del dimensionality curse nelle base di dati multi-dimensionali. Master's Thesis, University of Bologna

[Cia97] Ciaccia, P., Patella, M., Zezula, P. (1997) M-tree: An efficient access method for similarity search in metric spaces. Proc. VLDB 1997: pp426-435

[Cia98] Ciaccia, P., Patella, M., Zezula, P. (1998) A cost model for similarity queries in metric spaces. Proc. 17th ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems

[Chak] Chakrabarti, A., Chazelle, B., Gum, B., Lvov, A. (1999) A lower bound on the complexity of approximate nearest-neighbor searching on the Hamming cube. Proc. 31th Annual ACM Symposium on Theory of Computing, pp 305–311

[Cha01] Chávez, E., Navarro, G. Baeza-Yates, R. & Marroquín, J. L. (2001). Searching in metric spaces. ACM Computing Surveys, 33, 273–321.

[Cha01b] Chávez, E., Navarro, G. (2001). Towards measuring the searching complexity of metric spaces. Proceedings of ENC'01, 969–978.

[Cha03] Chávez, E., Navarro, G. (2003). Probabilistic proximity search: Fighting the curse of dimensionality in metric spaces. Information Processing Letters 85 pp. 39-46

[Cha05] Chávez, E., Navarro, G. (2005). A compact space decomposition for effective metric indexing. Pattern Recognition Letters, v.26, issue 9, pp 1363 1376

[Cha07] Chávez, E., Navarro, G., Paredes, R. (2007). t-Spanners for metric space searching. Data & Knowledge Engineering, Volume 63 , Issue 3, pp 820-854

[Cla05] Clarkson, K.L. (2005) Nearest-Neighbor Searching and Metric Space Dimensions. Nearest-Neighbor Methods for Learning and Vision: Theory and Practice, MIT Press, 2006.

[Dev97] Devroye, L., Györfi, L., Lugosi, G. (1997) A Probabilistic Theory of Pattern Recognition. Springer

[DIMACS] Sixth DIMACS Implementation Challenge: Available Software. http://www.dimacs.rutgers.edu/Challenges/Sixth/software.html

[Don00] Donoho, D. L. (2000) High-Dimensional Data Analysis: The Curses and Blessings of Dimensionality. AMS Conference: Math Challenges of the 21st Century

[Dud84] A Course on Empirical Processes. Lecture Notes in Math. Dudley, R.M. (1984) Springer, New York. 1984

[Dun97] Dunbar, S.R. The Average Distance between Points in Geometric Figures. The College Mathematics Journal. Vol. 28, No. 3, (May, 1997), pp. 187-197

[Fal03] Faloutsos, C., Papadias, D., Tao, Y. (2003) The Power-Method: A Comprehensive Estimation Technique for Multi-Dimensional Queries. 12th International Conference on Information and Knowledge Management (CIKM'03)

[Fri01] Friedman, J., Hastie, T., Tibshirani, R. (2001) The Elements of Statistical Learning: Data Mining, Inference, and Prediction. Springer

[Goy08] Disorder inequality: A combinatorial approach to nearest neighbor search. Goyal, N., Lifshits, Y., Schütze, H. (2008) WSDM 2008

[Gro83] A topological application of the isoperimetric inequality. Gromov, M., Milman, V.D. (1983) Amer. J. Mathematics

[Heg01] Hegland, M. (2001) Data mining techniques. Acta Numerica, 2001 pp313-355

[Ind04] Piotr Indyk (2004) ch 39 of Handbook of Discrete and Computational Geometry. Goodman, J.E., O'Rourke, J. CRC Press

[Kim] Kimball, R., Ross, M. (2002) The Data Warehouse Toolkit: The Complete Guide to Dimensional Modeling, 2nd Edition . Wiley

[Kle97] Kleinberg, J. (1997) Two algorithms for nearest-neighbor search in high dimensions. Proc. 29th ACM Symposium on Theory of Computing

[Lad05] Johnson, M. H., Ladner, R., Zatloukal, Z. Nearest neighbor search for data compression. Data Structures, Nearest Neighbor Searches, and Methodology: Fifth and Sixth DIMACS Implementation Challenges. AMS. (M. Goldwasser, D. Johnson, and C. McGeoch)

[Led01] Ledoux, M. (2001). The concentration of measure phenomenon. Math. Surveys and Monographs **89**, Providence: Amer. Math. Soc.

[Lif07] Lifshits, Y. (2007) Similarity search: a web perspective. Google Tech Talk 18 October 2007

[Mal03] Maltoni, D., Maio, D., Jain, Anil K., Prabhakar, S. (2003) Handbook of Fingerprint Recognition. Springer

[Men03] Mendelson, S. (2003) A few notes on statistical learning theory. Advanced Lectures in Machine Learning LNCS 2600, 1-40. Springer

[Mil86] Milman, V.D., Schechtman, G. (1986) Asymptotic theory of finite dimensional normed spaces. Springer Lecture Notes in Mathematics.

[Pes00] Pestov, V. (2000). On the geometry of similarity search: dimensionality curse and concentration of measure. *Inform. Process. Lett.* 73, 47–51.

[Pes02] Pestov, V. (2002). Elements of Asymptotic Geometric Analysis. Lecture notes, Victoria University of Wellington.

[Pes07] Pestov, V. (2007). Intrinsic dimension of a dataset: what properties does one expect? In: Proc. of the 22-nd Int. Joint Conf. on Neural Networks (IJCNN'07), Orlando, FL (pp. 1775-1780).

[Pes07b] Pestov, V. (2007) Elements of statistical machine learning: a mathematical introduction. Lecture notes, University of Ottawa.

[Sed98] Sedgewick, R. (1988) Algorithms. Addison-Wesley

[Sha06] Shaft, U., Ramakrishnan, R. (2006) Theory of nearest neighbors indexability. ACM Transactions on Database Systems (TODS).

[She07] Sheskin, D.J (2007) Handbook of Parametric and Nonparametric Statistical Procedures. Fourth Edition. Chapman & Hall/CRC

[SISAP] International Workshop on Similarity Search and Applications. http://sisap.org/

[Tay06] Taylor, Michael E. (2006) Measure Theory and Integration. AMS Bookstore

[Vap98] Vapnik, V. (1998) Statistical Learning Theory. Wiley series on adaptive and learning systems for signal processing, communications and control

[Vid03] Vidyasagar, M. (2003) Learning and Generalisation: With Applications to Neural Networks. Springer.

[Yia93] Yianilos, P. (1993) Data structures and algorithms for neares neighbour search in general metric spaces. SODA '93: Proceedings of the fourth annual ACM-SIAM Symposium on Discrete algorithms.

[Zez05] Zezula, P., Amato, G., Dohnal, V., Batko, M. (2005) Similarity search: the metric space approach. *Springer series: advances in database systems*, vol 32.