Universiteit Leiden
The Netherlands

Opleiding Informatica

Compact descriptors for (near) duplicate image detection

Lisette de Schipper

Supervisors:
Dr. M. S. K. Lew & Dr. E. M. Bakker

BACHELOR THESIS

Leiden Institute of Advanced Computer Science (LIACS)
www.liacs.leidenuniv.nl

22/08/2017

Abstract

(Near) duplicate detection underpins many different applications, like the detection of copyright infringement and the reduction of redundancy of an image dataset. This is usually achieved by representing an image as a feature descriptor and comparing this to a database. However, with the increasing vastness of the internet and the advance of mobile devices, it becomes essential to associate each descriptor with a small storage load. This thesis tries to answer the question whether it is too ambitious to pose a hard constraint on the storage load of a descriptor. The evaluation of multiple methods is preceded by a study to the different ways compact descriptors can be obtained. The evaluation itself focuses on novel, clear-cut methods, of which nine are handcrafted of 8 bytes or less, one is obtained via a convolutional neural network of 6 bytes, and one from the combined discriminative power of four handcrafted descriptors. The performance of the descriptors is measured by their accuracy, sensitivity to homogeneity, and applicability on a well-rounded, large dataset that contains images that have been altered by multiple kinds of transformations, including but not limited to changes in brightness, saturation, scale, and orientation, and combinations of transformations. The performance analysis clearly demonstrates the power of the clear-cut descriptors. The nature of this thesis is essentially exploratory, since it is one of the first attempts to find (near) duplicate images with such a hard constraint. The underlying research can be consulted for the design of future descriptors for this purpose.

Acknowledgements

I am very grateful for the support and patience of my supervisor, Dr. M. S. K. Lew, and the unconditional support of my family. During this process I was able to bounce a lot of ideas off Memsiepemsie who was always there to listen to my ramblings, despite not understanding them. I would also like to show my gratitude to those that have read this thesis and provided many valuable comments.

Contents

Chapter 1

Introduction

The World Wide Web has become saturated with trillions of images. Additionally, with the simplicity with which images can be produced and distributed, billions are added to the circulation every day. This includes copyrighted images that have been distributed illegally and images that have been used by multiple sources. The specific problem that concerns the duplication and often, alteration, of images, is the concern of this thesis. For different reasons, it has become necessary to detect all the images that have the exact same subject matter. The scale has driven us to attack the problem in such a way that promises both performance as well as a low memory usage.

1.1 Motivations

(Near) duplicate detection allows one to search an image database for all images that have the exact same subject matter. Typically, such a system ”describes” an image and uses this description to select candidate images. This is useful for situations in which someone only has a small sized version of a picture and requires the image in its original dimensions, but similar systems can be utilized for multiple purposes.

The same system can be used for the detection of illegally redistributed imagery. This problem not only violates copyright, but may also cause profit-losses. The descriptors may even be utilized by a traitor tracing scheme, that aims to identify the illegal redistributor.

Another application is the reduction of redundancy in a dataset. A high percentage of the content online consists out of mere (near) duplicates, which results in a huge waste of storage space and a longer retrieval time for image searches. By reducing this redundancy, storage and retrieval time can be optimized.

The prime example of this application is Google Image search. Whenever that search machine is queried with certain keywords many of the results are similar. It can thus be argued that it is useful to group actual (near) duplicates under a single image, as to not overwhelm the user with redundant information.

(Near) duplicate detection can also be used as an analytical tool for both online trends and the behaviour of the redistributors, as the dispersal of a certain subject matter can be monitored. For instance, an online newspaper may function as the prime dispersal vector for an image to gain attention among a community.

In "Challenges of automating the detection of paedophile activity on the internet" [PCM05] another real-life application was presented. The police keeps a collection of images related to paedophilia. This collection can be compared to the contents of a computer owned by a suspect in order to quickly asses his culpability.

The scale of the World Wide Web does not allow for storing high dimensional descriptions in floating-point representations, as this consumes significant amounts of memory in addition to a long matching time. Also, the widespread use of mobile devices capable of making high-resolution photographs suffer from power-constrained CPUs and limited storage, which also necessitate compact descriptions that can be computed quickly. The solution to this is using compact, binary descriptions.

1.2 Main contributions

The main contributions of the work presented in this thesis are summarised below.

- A compendium of the most common techniques to achieve compact, binary encodings for the purpose of (near) duplicate detection.
- An evaluation of a wide spectrum of clear-cut methods to ascertain which methods are most promising.
  - Among these methods is a self-learning method that uses a convolutional neural network.
  - A method is evaluated that combines multiple descriptors into a composite descriptor.
- A new dataset that consists out of thousands of images and their near duplicates generated by typical transformations that include (but are not limited to) changes in brightness, saturation, contrast, resampling, and even images that have been altered with multiple transformations, such as zooming in and the rotation of a resampled image.

1.3 Thesis overview

The thesis is structured as follows. Chapter 2 presents works that have a similar focus. Underlying concepts are explained in chapter 3. In chapter 4 methods are presented used by feature descriptors to obtain compact feature descriptor values, focusing on recent advances in the field of content-based retrieval. Eleven methods were developed based on the insights gained from these methods. These are presented and evaluated in chapter 5. The thesis is concluded in chapter 6. Finally, the final chapter points out the limitations of the current work and outlines directions for future work.

Chapter 2

Related Work

This work is inspired by the paper "Large Scale Image Copy Detection Evaluation" [THBL08], which compares methods for the purpose of detecting similar visual content on a large scale. The objectives are quite similar, except that a hard constraint is imposed on the research carried out here by limiting the storage load available for each method.

To our knowledge, aside from the study mentioned previously, there are no other studies dedicated to finding the essence of a suitable algorithm in a similar setting. However, a plethora of methods have been proposed to search and index the images in an attempt to shorten the total time of description and searching. These are usually compared to state-of-the-art methods. Despite all efforts, their substance is usually lost in the grand scheme of things, due to the absence of a standard model to evaluate them by.

Among the evaluated descriptors, one methods is proposed that consist out of multiple descriptors. However, other descriptors have also been proposed that successfully improve the distinctive power by combining multiple descriptors. Both Hu et al. [HCC+09] and Battiato et al. [BFPR14] propose methods that exploit the correspondence of multiple descriptors to capture different aspects of local regions for the purpose of (near) duplicate detection. In "Bundling features for large scale partial-duplicate web image search" [WKIS09] multiple SIFT features that appear in the same elliptical region are combined in a bundled feature. Fedorov and Kacher [FK16] propose a feature descriptor that combines triples of feature points with their geometric layout.

Chapter 3

(Near) duplicate detection

Different definitions and solutions exist, because the need to discriminate possible alterations and duplicates of images stems from different fields. This chapter will present important concepts and provide the necessary definitions in sections 3.1 and 3.2. The two main strategies to detect (near) duplicates, namely watermarking and content-based detection, are explained in sections 3.3 and 3.4 accordingly.

3.1 (Near) duplicates

The duplicate of an image is its exact copy. A near duplicate is an altered version of an original image that keeps a similar visual value. This altered version can be linked back to the original version through one or more transformations, like compression, brightness change or cropping. The properties of this relationship are as follows:

- Transitive: if an image A is a (near) duplicate of an image B and an image C is a (near) duplicate of an image B, then image C is in turn a (near) duplicate of image A.
- Symmetric: if an image A is a (near) duplicate of an image B, then image B is in its turn a (near) duplicate of an image A.
- Reflexive: obviously every image is a duplicate of itself.

In other words, it is an equivalence relationship, and thus every image can be split into equivalence classes. This apprehension is fundamental to the actual detection of duplicates, which is the focus of this thesis. However, the precise definition of near duplicate depends on the degree of variability (photometric and geometric) that is considered acceptable for each particular application.

3.2 (Near) duplicate detection

The goal of (near) duplicate detection is to search a collection of images for (near) duplicates of a query image. Ideally, the method returns the entire equivalence class the query image belongs to. There are two approaches suitable for this task. The oldest one watermarks the original image before publishing it. Duplicates of the original image can then be detected by checking the presence of the watermark within the images of the dataset. The second method relies on the analysis of the content of an image in order to extract relevant visual information. (Near) duplicates are then identified when their features are similar to those of the original image. In the next two sections we will expose these methods in more detail and present their advantages and drawbacks.

3.3 Digital Watermarking

Watermarking techniques embed information that can be traced back to the copyright holder into the image by modifying the content slightly. This information can be extracted and used for authentication. There are two apparent kinds of watermarks: one that is perceptible and one that is not.

The principle is as follows: a signature is embedded within the original image in a subtle yet robust way. This signature, or watermark, should hinder the illegal use of that image, for it can be easily detected to proof the authenticity. The watermark can also serve additional purposes, as it allows people to identify the copyright holder, and can be employed for source tracking by which different recipients receive differently watermarked content. If one of these is used illicitly, it can be traced back to the source.

Watermarking is not widely used in practical services, because all the perks are circumvented when the watermark is removed while keeping the integrity of the subject of the original image intact, rendering all watermarked versions useless. In a lot of cases, it is also just impractical to corrupt an image with a watermark, because whether the method results in an imperceptible watermark or not, the original image data will be degradated. The general consensus of the technique is therefore that it serves purpose in low-security related applications, like the logo of a tv network on their channel.

3.4 Feature description

Instead of corrupting an image with a watermark, the alternative option is to extract features from an image and describe these by a feature descriptor. A feature is a distinctive characteristic of information from an image. To find (near) duplicates of an image in a dataset, one only needs to compare its descriptor to the descriptors of the dataset. Note that, commonly the term ”feature descriptor” is used both for the method of extraction and the output (the description) of one.

When designing methods to extract descriptors a lot of options have to be considered. To mention a few, the selection of features, the scale of the actual extraction, and the dimension of the output. These choices result in a trade-off between robustness, discriminability, storage load and computation time.

The extraction of descriptors for (near) duplicate detection can be considered as a restricted kind of content-based image retrieval (CBIR). CBIR is widely used to retrieve images similar to the query image. CBIR does not only retrieve (near) duplicates, but also images that share the same or similar semantics. This means that it is not feasible to use existing CBIR techniques directly for our purpose, since they will result in false positives.

Chapter 4

Low bit-rate encodings

To determine whether a pair of images consists of (near) duplicates of each other, it is necessary to obtain descriptors of these images that can be compared. For the purpose of this research, they need to be compact and binary.

If feature vectors are obtained via local feature descriptors, it is very common to aggregate these in a single, global feature vector. Local descriptors extract descriptions from points of interests as determined by a so called detector algorithm. Section 4.1 provides more detail on this.

To facilitate retrieving concise, binary descriptors, it might be desirable to convert a D-dimensional vector to a code of B bits. Multiple paradigms enabling this conversion are explained in section 4.2. Feature descriptors can also be compressed and extracted via hashing, as explained in section 4.3. Finally, designs are imaginable that do not depend on existing descriptors or projections, but compare intrinsic data via binary tests. These are discussed in section 4.4.

4.1 Aggregates of local descriptors

For very large databases, it is unrealistic to store a set of local or regional features per image, which is why methods have been proposed to aggregate these into unique, global presentations.

The most widely used methods are based on the "Bag-of-Features" model (BoF). The key principle is that each image can be represented as a histogram that keeps track of the amount of times certain features are used within that image. Each feature is an image patch with a certain characteristics (a local feature). The BoF representation is then binarized by means of a paradigm as described in sections 4.2 and 4.3. To compare a query image to the images in the database, only the distances between the histograms need to be computed.

Other well known aggregation methods are the Fisher Kernel [PD07] and the simplified representation of these, the Vector of Locally Aggregated Descriptors, usually referred to as VLAD [JDSP10]. Performance-wise,

these representations have been thoroughly evaluated and compared in the context of image classification [CLVZ11] [BS09] [JPD+12].

It been shown that the difference in overall performance of the Fisher Kernel and VLAD is negligible. The advantage FV and VLAD have over BoF is that they can be computed from much smaller vocabularies [JDSP10]. However, they both have two major drawbacks, which is their memory and computational cost. Fortunately, these can be alleviated using compression techniques [PLSP10].

4.2 Compression techniques

Compression techniques minimize or reduce the number of variables. The main paradigms are explained here. They are often used in conjunction with each other and with hashing, and along with these hybrids, a lot of derived methods have also been proposed.

4.2.1 Entropy encoding

Entropy encoding is a lossless compression technique that aims to find the smallest number of bits needed to represent a symbol on average. This is achieved by deducing the statistical occurrence (probability) of each symbol. The two most common used entropy coding techniques are Huffman coding and arithmetic coding.

The Huffman coding assigns a smaller number of bits to symbols that occur more frequently and a longer number of bits to symbols that occur less frequently. Huffman coding is a prefix code, which means that there is no code word possible that is a prefix of another codeword in the same system.

Arithmetic coding assigns a code word to the input in its entirety, as opposed to Huffman which assigns a codeword to a symbol. Arithmetic coding takes a message and converts it to a floating point number greater than or equal to zero and less than one.

4.2.2 Vector quantization

Vector quantization (VQ) is a lossy type of data compression, as it aims to construct a codebook of representative vectors. VQ reduces the size of a vector by compressing the range of values, and is therefore only based on the bias of values, while ignoring the spatial structure, as opposed to the methods mentioned before. The most common technique is product quantization proposed by Jégou, Douze and Schmid [JDS11]. The idea behind this method is to split each input vector into subvectors that can be quantized seperately.

4.2.3 Statistical analysis

The two flagships that both utilize the variance by maximizing scatter are Principal Component Analysis (PCA) and Linear Discriminant Analysis (LDA). These are both methods that aim to reduce redundancy by analyzing all the data.

The most common, unsupervised, approach is the Principal Component Analysis (PCA) and its variations, which have been exploited wide and beyond to reduce the dimensionality of vectors [KS04] [WHB09]. PCA aims to maximize the variance in the dataset by projecting the data into a lower dimensional manifold. Principal components are the variables introduced by PCA to differentiate between the groups in the data.

Linear Discriminant Analysis (LDA) is its supervised counterpart. Whereas PCA aims to maximize the total scatter, LDA aims to minimize the within-class scatter while maximizing the between-class scatter. By computing the ideal directions (linear discriminants) it seeks to maximize the separation between multiple classes. The major drawback of LDA is that it may encounter the so-called small sample size problem [Fuk13]. This problem occurs when the number of training samples is less than the feature dimension. The classical solution to overcome this was to apply PCA on the raw data and performing LDA on this PCA subspace, but other solutions have been proposed [CCL05] [DB12] [TD11].

Studies are conflicting when it comes to the performance of these two methods to reduce dimensionality. As one might expect, given a large and representative learning dataset, LDA should have the upper hand [MK01].

PCA and LDA are both linear methods, which limits them in their applicability, since many high dimensional data sets have a nonlinear nature. In these cases the high-dimensional data lie on or near a nonlinear manifold as opposed to a linear subspace. Both PCA and LDA are not capable of modelling the variability of the data correctly. However, multiple methods have been proposed to overcome this limitation [TDSL00] [RS00] [SSM98] [MRW$^+$99].

4.3 Similarity-sensitive hashing

Nearest neighbour search addresses the problem of finding the most similar data from a database. It is therefore fundamental for (near) duplicate detection. The linear search is impractical to be applied in big databases, so a lot of research has been devoted to investigate methods to reduce search complexity. Recent work shows that learning binary projections is a powerful way to index a database. The basic idea behind this is to formulate the projections so as to approximately preserve similarity. They were first introduced by Salakhutdinov and Hinton [SH09] for text retrieval, and then introduced to computer vision by Torralba et al [TFW08]. The basic idea is to map data points to a finite number of hash points, so that similar data points have a larger probability of having the same hash code.

Since these methods are still relatively young, they have not been extensively employed for (near) duplicate detection, but they have become a mainstay for many other problems in computer vision, such as object

recognition, image retrieval and local descriptor compression.

Because of the recent success of descriptors produced by convulutional neural networks (CNN's) for large-scale image classification problems [KSH12] [STE13], it is also interesting to look at their potential to extract discriminative descriptors for (near) duplicate detection. In general, a deep convolutional network consists of two these layers: the convolution layers, the max-pooling layers, the fully connection layers and the output layers. The weight sharing in the convolutional layers and the appropriate pooling schemes allow for some translation and scale invariance. Despite being theoretically attractive, they are relatively expensive to apply in a large scale setting.

There are two ways to employ CNN's, the first method is build on top of existing handcrafted descriptors. As a result, the performance of this method is limited by the representation power of these descriptors. The second method learns a set of non-linear projection functions to map an image to a binary string [LJC+13] [TCLF12] [ZK15] [SSTF+15] [OLYL14] [WSL+14].

Hashing methods can be independent of the data available, making use of random projections, or learning-based. Previous work has shown that learning-based hashing is, in general, superior to the data-independent hashing [AI06] [WKC10] [LLCZ16].

4.3.1 Random projection based hashing

The hashing methods that do not make use of any training data, usually integrate a random factor. The path most travelled by seems to be the algorithms based around locality-sensitive hashing (LSH) [GIM99] [AI08] [DIIM04] [KG09]. In LSH, binary codes are generated by using a random projection matrix and tresholding using the sign of the projected data. This group of algorithms usually require longer codes to attain the same kind of accuracy as the methods that do make use of training data [GL11].

4.3.2 Learning-based hashing

These methods are dependent on the data and can be categorized as: unsupervised, supervised and the hybrid, semi-supervised methods. Unsupervised learning does not have a set objective and has therefore no parameters, whereas supervised learning does. In layman's terms, supervised learning will actually learn in terms of characteristics why a certain image is not like another image and unsupervised learning will aim to model the underlying structure. The data for supervised learning for (near) duplicate detection often consists of pairwise labels of similar and dissimilar image pairs. Semi-supervised algorithms use the information from both labelled and unlabelled samples to learn hash functions. Typically, a larger amount of unlabelled data is used.

4.4 Binary tests

Instead of aggregating local descriptors and reducing the dimensionality of the resulting vectors or mapping the data to a fixed size via projections by hashing, it is also possible to design descriptors that produce low bit-rate descriptor values by binary tests. Thomee et al. [THBL08] presented a straightforward method to obtain discriminative descriptor values, namely the median method. The median method compares the intensity of image patches to the overall median intensity to form a vector. The idea to perform binary tests has also been used by local feature descriptors, such as Binary Robust Independent Elementary Features (BRIEF) [CLSF10] and Local Difference Binary (LDB) [YC12]. As opposed to the median method, the binary tests performed here have two image patches as input as opposed to an image patch and a global median for a certain feature.

Chapter 5

Evaluation

The design of the evaluation is based on the goal described in section 5.1 and the hypotheses that support it. These hypotheses are stated in section 5.2. The insights gained by the first half of the thesis allowed us to adopt several methods for this purpose. Section 5.3 discusses image features that were used by the implemented handcrafted descriptors. The construction and an overview of all descriptor methods are described in section 5.4. In section 5.5 the datasets used for the evaluation of the feature descriptor are described and section 5.6 elaborates on the used set-up, and implementations. Section 5.7 explains the methods we have chosen to evaluate the descriptors by. Finally, section 5.8 presents the experiments and their results.

5.1 Goal

The main goal of this thesis is to analyze the discriminatory power of compact methods for the purpose of (near) duplicate detection on a large scale. The feature descriptors we will be testing will have to meet the following requirements:

1. The descriptor values needs to be equal or less than 8 bytes.
2. The descriptor value must be binary.

The maximum storage load of the composite descriptor, which is composed of other descriptors, amounts to 32 bytes. This means that the footprint of each descriptor value is very small, and allows them to be loaded into memory very quickly. The third, fourth and fifth requirements will be investigated during the evaluation.

The choice for the second requirement is based on the fact that binary nature allows us to compare all descriptors by the Hamming distance, which is extremely fast as it can perform millions of comparisons per second on standard computers [WTF09]. Plus, the calculation of the Hamming distance is natively supported on modern processors (XOR operation followed by bit count) [LCS11].

The tests will then determine to what extent the descriptors satisfy the following quality standards:

1. The descriptor value needs to be computed in a reasonable amount of time.
2. The descriptor is discriminative enough to distinguish (near) duplicates from the rest of the collection.
3. The descriptor is robust enough to resist commonly used image transformation.

The combination of the scale of concern and the first quality standard, an emphasis is placed on primitive descriptors. This will ensure a faster extraction time. Because of the performance of the median method as described by Thomee et al. [THBL08], similar methods were implemented, and as a result of the promising results shown in other computer vision problems for learning-based hashing, a method was implemented that extracts descriptors via a convolutional neural network.

5.2 Hypotheses

It is hypothesized that methods that extract descriptors of no more than 8 bytes and their combined efforts can efficiently detect (near) duplicates from a dataset. Based on the research in Chapter 4 the following hypotheses were developed:

1. Based on the performance of neural networks for other computer vision problems, it is expected that descriptors based on self-learned features will be just as good, if not better, than handcrafted features.
2. It is expected that a a descriptor created by multiple descriptor methods is more robust and precise than individual descriptors.
3. The median method [THBL08] can be improved by using of different features.
4. The median method can be improved by the use of a different approach that does not compare each cell to a global value, but to the values of neighbouring cells, especially for certain photometric transformations, since not all cells need to be affected.

5.3 Features

For many years in computer vision, raw pixel values of image statistics as color, gradient and filter responses have been the most elementary choices for image features. The descriptor methods that were implemented also make use of these features.

5.3.1 Image gradient

An image consists out of pixels that have three attributes: an x-coordinate, a y-coordinate, and a certain intensity. The derivative of an image can be taken with respect to x or with respect to y at a given pixel and

together they make up the image gradient. The image gradient is often used for edge detection, as it lays down the directional change in an image's intensity.

We approximate the horizontal derivative, G_x, and the vertical derivative, G_y as follows

$$G(x,y)_x = A \times I(x,y) \tag{5.1}$$

$$G(x,y)_y = B \times I(x,y) \tag{5.2}$$

$$\tag{5.3}$$

where $I(x,y)$ is the source image and A and B are the 3×3 kernels that I convolves with to calculate approximations of the derivatives.

$$A = \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix} \tag{5.4}$$

$$B = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} \tag{5.5}$$

The magnitude of image gradient specifies the rate of change at pixels (x,y). If the image vector $I(x,y)$ is defined as

$$\nabla I(x,y) = \begin{bmatrix} \frac{\partial I(x,y)}{\partial x} \\ \frac{\partial I(x,y)}{\partial y} \end{bmatrix} = \begin{bmatrix} I_x \\ I_y \end{bmatrix} \tag{5.6}$$

then the magnitude of the gradient can be simply calculated with the formula:

$$|\nabla I(x,y)| = \sqrt{I_x^2 + I_y^2} \tag{5.7}$$

5.3.2 Image moments

An image moment is a weighted average or the sum of the pixels' intensities of a region of an image. These moments can be used to derive properties from images.

A geometric moment is defined by the following formula:

$$m_{pq} = \sum_{i=0}^{N} \sum_{j=0}^{N} f(i,j) i^p j^q \tag{5.8}$$

where $N \times N$ is the image size, and $f(i,j)$ expresses the image gray level. We can use this formula to derive certain properties, such as the total sum of pixel values via m_{00} and the center of mass. The center of mass is the points where the mass of the entire investigated region could be concentrated without changing the first moment of the image about any axis. In other words, the coordinates of the center of mass would be:

$$\tilde{x} = \frac{m_{10}}{m_{00}} \tag{5.9}$$

$$\tilde{y} = \frac{m_{01}}{m_{00}} \tag{5.10}$$

Moments can be used to derive invariants with respect to specific transformation classes. Sets of these moments can then be used to uniquely describe the information contained an image or an image patch. However, an infinite number of moment values is required to obtain all of the information an image can contain. Out of the necessity to select the most meaningful moments, Hu [Hu62] derived a subset of seven invariant moments which are invariant under image translation, scaling, and rotation. These are easy to compute, but due to the restraint put on the footprint only the first two invariants were used:

$$I_1 = \eta_{20} + \eta_{02} \tag{5.11}$$

$$I_2 = (\eta_{20} + \eta + 02)^2 4\eta_{11}^2 \tag{5.12}$$

Where η_{pq} is the central moments of order $(p+q)$, which can simply be seen as weighted moments, assuming the data follows a certain distribution.

Flusser and Suk [FS93] also derived a set of invariants based of statistical features. These Affine moment invariants (AMI's) were derived by the theory of algebraic invariants and are invariant under the general affine transformation.

$$u = a_0 + a_1 x + a_2 y \tag{5.13}$$

$$v = b_0 + b_1 x + b_2 y \tag{5.14}$$

where (x,y) and (u,v) are coordinates in the image plane before and after the transformation respectively. We decided to use the following three AMI's:

$$I_1 = \frac{1}{\eta_{00}^4}(\eta_{20}\eta_{02} - \eta_{11}^2) \tag{5.15}$$

$$I_2 = \frac{1}{\eta_{00}^{10}}(\eta_{30}^2\eta_{03}^2 - 6\eta_{30}\eta_{21}\eta_{12}\eta_{03} + 4\eta_{20}\eta_{12}^3 + 4\eta_{03}\eta_{21}^3 - 3\eta_{21}^2\eta_{12}^2) \tag{5.16}$$

$$I_3 = \frac{1}{\eta_{00}^7}(\eta_{20}(\eta_{21}\eta_{03} - \eta_{12}^2) - \eta_{11}(\eta_{30}\eta_{03} - \eta_{21}\eta_{12}) + \eta_{02}(\eta_{30}\eta_{12} - \eta_{21}^2) \tag{5.17}$$

where η_{pq} is, again, the central moments of order $(p+q)$.

5.3.3 Gabor filter

The Gabor filter is often used to describe the texture of an image. Among the wavelet functions, it was determined that it has the best discriminatory power [MM96]. The function of the filter is appraised for its similarities to the human visual system. We extract our feature by taking the magnitude of the Gabor filter, since its more informative than the imaginary and real part [LKF05] .

A Gabor filter is defined by its scale parameter σ, wave frequency λ, and orientation θ, phase offset ψ and aspect ratio γ. The general form of the symmetrical 2d Gabor kernel $G_R(x,y)_{\sigma,\lambda,\theta,\psi,\gamma}$ and the anti-symmetrical 2d Gabor kernel $G_I(x,y)_{\sigma,\lambda,\theta,\psi,\gamma}$ can be expressed as follows:

$$G_R(x,y)_{\sigma,\lambda,\theta,\psi,\gamma} = exp(-\frac{x'^2 + \gamma y'^2}{2}\psi^2)cos(2\pi\frac{x'}{\theta} + \theta) \tag{5.18}$$

$$G_I(x,y)_{\sigma,\lambda,\theta,\psi,\gamma} = exp(-\frac{x'^2 + \gamma y'^2}{2}\psi^2)sin(2\pi\frac{x'}{\theta} + \theta) \tag{5.19}$$

where

$$x' = xcos(\theta) + ysin(\theta)$$
$$y' = -xsin(\theta) + ycos(\theta)$$

Let $I(x,y)$ denote a grayscale image and $G_{\sigma,\lambda,\theta.\psi}(x,y)$ represent a Gabor kernel. The filtered images would are then created by convoluting these two kernels with the original image, as follows:

$$C_R(x,y)_{\sigma,\lambda,\theta,\psi,\gamma} = I(x,y) \times G_R(x,y)_{\sigma,\lambda,\theta,\psi,\gamma} \tag{5.20}$$

$$C_I(x,y)_{\sigma,\lambda,\theta,\psi,\gamma} = I(x,y) \times G_I(x,y)_{\sigma,\lambda,\theta,\psi,\gamma} \tag{5.21}$$

In our implementation 2D Gabor filters were used with the eight different orientations $\Theta = \{0, \frac{\pi}{8}, \frac{2\pi}{8}, \frac{3\pi}{8}, \frac{4\pi}{8}, \frac{5\pi}{8}, \frac{6\pi}{8}, \frac{7\pi}{8}\}$,

scale parameter $\sigma = 1$, wavelength $\lambda = 15$, phase offset $\psi = \frac{\pi}{2}$ and aspect ratio $\gamma = 0.5$.

An average image was formed out of the different orientations, resulting into

$$C_{R_{average}}(x,y)_{\sigma,\lambda,\theta,\psi,\gamma} = \frac{\sum_{\theta\in\Theta} C_R(x,y)_{\sigma,\lambda,\theta,\psi,\gamma}}{8} \tag{5.22}$$

$$C_{I_{average}}(x,y)_{\sigma,\lambda,\theta,\psi,\gamma} = \frac{\sum_{\theta\in\Theta} C_I(x,y)_{\sigma,\lambda,\theta,\psi,\gamma}}{8} \tag{5.23}$$

Based on these results, the magnitude response $M(x,y)_{\sigma,\lambda,\theta,\psi,\gamma}$ of the Gabor filter can be computed as follows:

$$M(x,y)_{\sigma,\lambda,\theta,\psi,\gamma} = \sqrt{C^2_{R_{average}}(x,y)_{\sigma,\lambda,\theta,\psi,\gamma} + C^2_{I_{average}}(x,y)_{\sigma,\lambda,\theta,\psi,\gamma}} \tag{5.24}$$

5.3.4 Amount of interest points detected by DOG

The Scale-invariant feature transform, better known as SIFT,is a well-known histogram-based descriptor that computes local descriptors [Low99]. It describes the minima and maxima of a difference of Gaussian function applied in scale space. These are computed by building an image pyramid with sampling between each level. Interest points are then selected at regions and scales of high variation. A pixel is considered highly variable if it is a minimum or maximum compared to its eight neighbours at the same level in the pyramid. If it is a maximum or minimum, then the closest pixel location is calculated at the next lowest level of the pyramid. If the outcome is the same, the test is repeated for the level above. If the result is the same for all three levels, it is a potential interest point. At this point low-contrast and edge interest points are eliminated by setting a threshold for intensity and analyzing the eigenvalues.

5.4 Methods of extraction

All implemented feature descriptor methods are described in this section. They are categorized by the fashion with which they extract the binary descriptor. The first two kinds of descriptors are handcrafted and the final descriptor does the feature engineering automatically. For convenience all implemented descriptors have been summarized in Table 5.1.

5.4.1 Median descriptors

The original median method was introduced by Thomee, Huiskes, Bakker and Lew [THBL08]. The clear-cut descriptor occupies only 8 bytes. The original descriptor is extracted as follows: After resampling and converting an image I into grayscale, the image is divided into an 8×8 grid. Then the value of the feature from

each cell is compared to the entire grid, as shown by binary test in Expression 5.25. Every cell is assigned 1 if its value is greater than the median value and 0 if it is smaller. The bit errors between two feature descriptor values determine whether two images are (near) duplicates of each other.

$$L_{med}(Func(i))) := \begin{cases} 1 & \text{if } Func(i) > Func(image) \\ 0 & \text{else} \end{cases} \tag{5.25}$$

where $Func(\cdot)$ is the function for extracting information from a grid cell.

The original method used the first image moment, m_{00}, the intensity of the gray value, to evaluate the cells against the grid. It was tested against image overlaying, cropping, zooming in, contrast, sharpening, downsampling and compression, and the implementation that resampled the image to 8×8 bits was the most successful at this. Aside from the original descriptor, which is henceforth referred to as Med_m_{00}, three other descriptors were extracted with the same sampling pattern.

Med_Hu compared the first Hu invariant between cells and the global median.

Med_SIFT extracted SIFT interest points for the entire image, cropped the edges off that contained less than 8 interest points in order to capture more information, and compared the number of interest points to the average number of interest points in the image.

Med_Gabor was extracted by comparing the output from the magnitude of the average filtered image composed from the original image with 8 Gabor filters applied to it of the entire grid to the cells.

5.4.2 Self similarity descriptors

Inspired by LDB [YC12], a method was developed that divides an image I into $n \times n$ non-overlapping, equal sized grids. Cross-correlations between adjoining cells are evaluated by the binary test shown in Expression 5.26.

$$L_{ss}(Func(i), Func(j)) := \begin{cases} 1 & \text{if } (Func(i) - Func(j)) > 0 \text{ and} i \neq j \\ 0 & \text{else} \end{cases} \tag{5.26}$$

where $Func(\cdot)$ is the function for extracting information from a grid cell and i and j are a pair of grid cells.

Two descriptors are computed with a 63 bit descriptor, by dividing the image in a 4×4 grid. $Ss4\_Hu$ compares the first and second Hu invariant between adjoining cells and $Ss4\_m_{00}mag$ compares the values of $m_0 0$ and magnitude of Sobel filters between cells.

Three descriptors are computed by dividing the images into 3×3 grids. $Ss3\_m_{00}dydx$ uses the first image moment and the derivatives in the y and x direction. $Ss3\_AMI$ uses the first three AMI's and $Ss3\_m_{00}m_{02}m_{10}$ compares the image moments m_{00}, m_{01} and m_{10} between cells. All of these descriptors occupy 60 bits each.

Name	#bits	Extraction Method	Paradigm	Features
Med_Gabor	64	Median	Linear filter	Magnitude of the average of the Gabor output
Med_Hu	64	Median	Statistical features	first Hu invariant
*Med_*m_{00}	64	Median	Intensity	Image moment m_{00}
Med_SIFT	64	Median	interest point detector	amount of SIFT keypoints
*Ss*3*_AMI*	60	Self similarity 3×3	Statistical features	first three AMI
*Ss*3*_*$m_{00}dydx$	60	Self similarity 3×3	Intensity & change in intensity	Image moment m_{00}, & x & y derivative
*Ss*3*_*$m_{00}m_{01}m_{10}$	60	Self similarity 3×3	Intensity & center of Mass	Image moment m_{00}, m_{01} and m_{10}
*Ss*4*_Hu*	63	Self similarity 4×4	Statistical features	First two Hu Invariants
*Ss*4*_*$m_{00}mag$	63	Self similarity 4×4	Intensity and magnitude	Image moment m_{00} & magnitude
DHN [KSH12]	48	Supervised Learning	Machine Learning	Self-learned

Table 5.1: The representative methods selected for evaluation.

5.4.3 Composite descriptors

To improve the performance of single handcrafted descriptors a scheme is proposed that extracts multiple descriptors. The aim is to increase both precision and recall by using a set of feature descriptors. Matching two images will be a 4 step process, where each step should both complete and fine-tune the results. The advantage over a single large feature descriptor is the gained flexibility, as the descriptors the scheme is composed of are able to partially match images. Two matched feature descriptors are allowed to have a large overlap in error, since further down the procedure another feature can determine whether a candidate is a feasible (near) duplicate or not.

5.4.4 Deep learning of hash codes

We used a CNN to extract domain specific image representations via supervised hashing without the use of handcrafted descriptors. The used CNN is called the Deep hashing Network (*DHN*) and was published in 2016 [ZLWC16].

The model accept input images in a pairwise form (x_i, x_j, s_{ij}) and process them through a deep hashing pipeline. When $s_{ij} = 1$ it is implied that x_i and x_j are similar and when $s_{ij} = 0$ it is implied that x_i and x_j are dissimilar. The network is based on CNN AlexNet [KSH12]. Alexnet has five convolutional layers with max-pooling operations(F1–5), followed by two fully connected layers(F6-7) and an output layer. There are two layers added to Alexnet to the top of F7 to transform the output of F7 into a K-dimensional hash-coding.

5.5 Datasets

There is no sense in using classical datasets that have been used for image retrieval to evaluate the methods by, since our aim is to retrieve the query image that has been affected by one or multiple photometric and geometric transformations. To prevent underfitting for DHN while still getting a decent scale for the testset, we have combined and altered four datasets, namely the INRIA Copydays dataset [JDS08], the University of Kentucky Recognition Benchmark Images dataset [NS06], the Leeds Butterfly Dataset [WME09], and the MIRFLICKR 1M dataset [HL08].

The INRIA Copydays dataset was created for the purpose of (near) duplicate detection. It contains 157 various images featuring all sorts of subjects, among which are nature, environments, buildings and people, and their variants subjected to rotation, cropping and 'strong attacks'.

The University of Kentucky Recognition Benchmark Images dataset consists of sets of four images all featuring the same subject from a different angle. In total it consists of 10200 images. It was created in 2006 and is frequently used in studies.

The Leeds Butterfly Dataset is comprised of 832 images of 10 different species of butterflies. Per species up to 100 images are present, which may result in false positives, as will the different viewpoints of the University of Kentucky Recognition Benchmark Images dataset.

These datasets were subjected to the transformation shown in Table 5.2 to generate the (near) duplicates. These transformations are a good representation of the most common ones encountered on the internet [QMC05] [THBL13] [FZST07].

To ensure our combined dataset is representative of the real-world we are also using the MIRFLICKR 1M dataset. The images from this set will serve as the noise that shouldnt be retrieved when gathering the (near) duplicates and will allow us to take into account the impact of large scale on the performance of the feature descriptors.

The total size of the dataset amounts to 1 760 850 images (a couple of images got corrupted and were not used). Figure 5.1 shows 11 images from the dataset. They depict the original and 10 altered versions to illustrate the transformations as described in Table 5.2.

ID	Categories	Transformations
1	Reducing the color depth	Reduce the color palette to 256 colors
2	Changing brightness	Change the brightness to 70%, to 90%, to 110%, and to 130%
3	Changing saturation	Change the saturation value to 70%, to 90%, to 110%, and to 130%
4	Sharpening	Sharpen with 5%, with 10%, with 20%, with 30%, with 40%, and with 50%
5	Blurring	Applying a gaussian blur with radius of 1.0px, 1.5px, and 2.0px, and Applying an iris blur with radius 1.5px
6	Changing the contrast	Increase the contrast by 10%, and by 20%
7	JPG compression	with quality factor 100, 90, 80, 70, 60, 50, 40, 30, 20, and 10
8	Image overlaying	Add small logo Add big logo Add text Add an outer frame of 10% of the image size
9	Cropping (center)	Crop by 5%, by 10%, by 20%, and by 30%
10	Cropping and rotating	Crop by 20% and rotate 20 degrees, and Crop by 20% and rotate 40 degrees
11	Resampling	Downsample by 20%, by 40%, by 50%, by 60%, and by 80% Upsample by 20%, by 40%, by 50%, by 60%, by 80%, and by 100%
12	Resampling and rotating clock-wise	Downsample by 20% and rotate by 10 degrees, and Downsample by 50 and rotate by 20 degrees
13	Mirroring	Mirror the image
14	Rotating clock-wise	Rotate by 2 degrees, by 5 degrees, by 10 degrees, by 20 degrees, by 90 degrees, by 180 degrees, and by 270 degrees
15	Rotating clock-wise and cropping	Rotate by 10 degrees and crop 10%, and Rotate by 20 degrees and crop 40%
16	Zooming in	Upsample to 120% and crop to original size, Upsample to 150% and crop to original size, and Upsample to 200% and crop to original size

Table 5.2: The transformations of interest.

(a) Original (b) Changed brightness to 130% (c) Changed saturation to 130%

(d) Sharpened with 50 (e) Blurred with the iris blur (f) Increased the contrast by 20%

(g) With a big logo (h) Cropped by 30% (i) Cropped by 20% and rotated 40 degrees

(j) Mirrored (k) Rotated by 20% and cropped 40% (l) Zoomed in 200%

Figure 5.1: A sample for 10 out of the 16 transformation categories from Table 5.2. The original image is from the Leeds Butterfly Dataset.

5.6 Set-up, configurations and implementations

All handcrafted methods were implemented in the C++ language with the use of the OpenCV 3.2 library [Braoo]. They all convert the image to grayscale and determine the dominant orientation. The conversion to grayscale discards all color information and solely stores the intensity. This ensures that changes in the hue of a certain image do not affect the performance of the methods.

In a couple of renown descriptors rotation invariance is achieved by determining the dominant orientation of the image before extracting the descriptor. We decided to go this route as well to ease the load on the extraction methods. Before computing the descriptor,the dominant orientation was determined via the first and second order central moments of patterns of the binary version of the image.

For the DHN method the ILSVRC2012 network for classification was finetuned. According to several papers [RASC14] [YCBL14] [WL16], pre-trained networks still carry their weight when challenged with tasks different from what they were trained. The ILSVRC2012 that was originally trained on 1.2 million images from the ILSVRC2012 for classification, was finetuned on 402776 random images from UKBench from each image category and 286852 images from the MIRFLICKR 1M dataset. Before DHN passes an image through its network, it is resampled to 277×277 pixels to reduce computation time. It is however not converted to grayscale.

All tests were were performed on an Intel® Core™ i7 CPU 930 @ 2.80GHz $\times$ 8 machine with 16 GB RAM running Ubuntu 14.04.5 LTS. There was no GPU used for computations.

Adobe Photoshop was used to create all transformed variants of the original images. To resample the images the bilinear method was used, which sets the color of each pixel according to the pixels surrounding it. All proportions were restrained for each resample.

All cropping operation were performed with the anchor in the center.

To reduce the color depth we converted each image to indexed colors. We used the local selective palette type which favours broad areas of color and the preservation of web colors. There is no forced inclusion of certain colors and absent colors are simulated by dithering the available colors. 75% of the colors is dithered. The conversion does not prevent colors that are present in the image table from being dithered. The end-product is saved as an uncompressed PNG file.

The sharpening operation was performed with the Smart Sharpen filter. The surrounding 3.4px of each edge pixel are affected by this operation. Lens blur was the algorithm of choice to sharpen each image.

All interpolation was linear.

The blur radius of the gaussian filter determines how far the filter searches for pixels to blur.

The compression quality of JPEG compression was bicubic.

Sharpening was achieved via the smart sharpen filter. The amount of sharpening was 5% and the radius was 3.4%.

5.7 Evaluation methods

The goal is to determine to what extent each method is able to detect (near) duplicates of a query image and their efficiency at this. The hypotheses of Section 5.2 will be tested by the five experiments have devised here. The methods we have used to capture the performance of the methods are described here

The precision of a method is defined as the fraction of the correct (near) duplicates retrieved.

$$precision = \frac{|\{detected\ (near)\ duplicates\} \cap \{retrieved\ images\}|}{|\{retrieved\ images\}|} \tag{5.27}$$

The recall is defined as the fraction of retrieved (near) duplicates over all (near) duplicates

$$recall = \frac{|\{detected\ (near)\ duplicates\} \cap \{retrieved\ images\}|}{|\{all\ relevant\ (near)\ duplicates\}|} \tag{5.28}$$

A method to get an impression of the difficulty of each transformation category is by calculating the average normalized ranking performance. This evaluation method was measured in a manner similar to the method described in "Large Scale Image Copy Detection Evaluation" [THBL08], and is defined as follows:

$$rank_{normalized}(i) = 1 - \frac{rank_{average}(i)}{|\{dataset\ images\}|} \tag{5.29}$$

$$rank_{average}(i) = \frac{1}{|Q|} \sum_{j \in Q} rank(j) \tag{5.30}$$

$$Q = \forall x \in \{dataset\ images\} \wedge T(x, i) \tag{5.31}$$

$$T(x, t) : x \text{ is a (near) duplicate of the query images} \wedge x \text{ belongs to the transform category } t \tag{5.32}$$

where i is a transformation category from Table 5.2 and $rank(j)$ returns the position in the list returned by the method of interest for image j. $|Q|$ results in the number of (near) duplicates that belong to category i for however many images have been used to test the method of interest. For example, there are 4 transformations within the "image overlaying" category. The $|Q|$ for one query image would equate to four, for two query images eight. This set-up allows $rank_{normalized(i)}$ to award a (near) perfect method for a certain transformation category with 1.

method	Copydays	Butterfly	UKBench
Med_Gabor	0.808	0.753	0.737
Med_Hu	0.794	0.794	0.796
Med_m_{00}	0.794	0.779	0.780
Med_SIFT	0.764	0.623	0.452
$Ss3\_AMI$	0.783	0.756	0.771
$Ss3\_m_{00}dydx$	0.672	0.627	0.606
$Ss3\_m_{00}m_{01}m_{10}$	0.743	0.708	0.580
$Ss4\_Hu$	0.789	0.773	0.770
$Ss4\_m_{00}mag$	0.832	0.803	0.798

Table 5.3: The area under curve (AUC) for each method per type of query images.

5.8 Experiments

To get an accurate grasp of the potential of the methods, multiple experiments were conducted. The first three are focused on the handcrafted descriptors, focussing on the sensitivity to homogeneity and subject matter, sensitivity to the different transformations, and applicability. The fourth and fifth experiment investigate the potency of a composite descriptor and a self-learning descriptor respectfully. The last experiments investigates the computation time of descriptors by the self-learning and handcrafted descriptors.

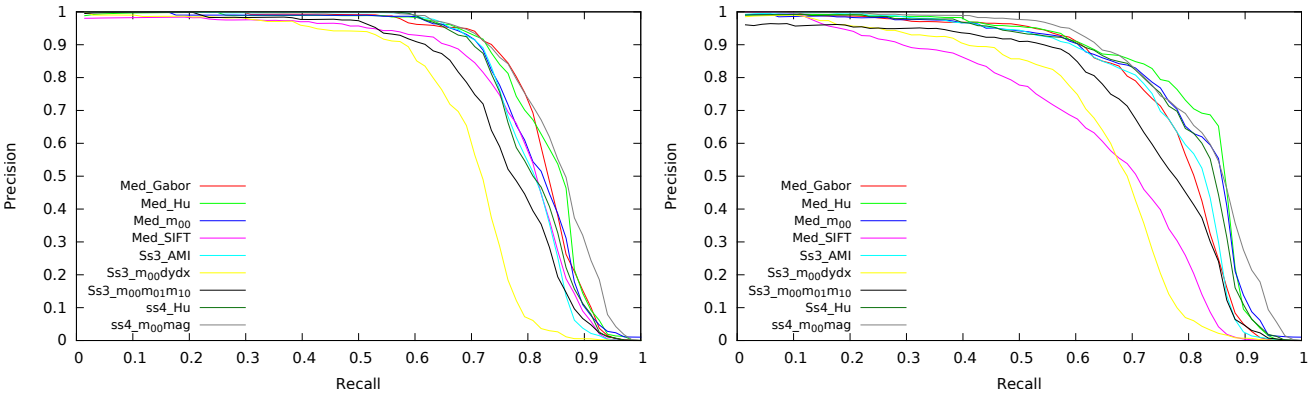
5.8.1 Accuracy and sensitivity to homogeneity/subject on handcrafted descriptors

As described in section 5.5 the dataset used for testing is composed of four different datasets, of which one acts as noise. The other three have a varying degree of homogeneity. Copydays is a very varied set. Leeds Butterfly is less varied and can be described as a lot of close-ups of ten kinds of butterflies. UKBench is not very varied at all, as it contains subjects photographed from different angles. To see how the performance is affected by the different kinds of dataset, we let the methods enumerate all the duplicates of a given query image. We measured the performance with the precision and recall curve (P-R curve).

Figure 5.2 shows how the methods performed per dataset. According to the figure there is no clear method outperforming other methods. The curves are crossing quite often, which is why we listed the Area Under Curve in Table 5.3. From this table, it appears that $Ss4\_m_{00}mag$ outperforms the other methods. Between the three datasets, all methods perform best on the Copydays dataset, which is the most heterogeneous dataset. Especially Med_SIFT, $Ss3\_m_{00}dydx$ and $Ss3\_m_{00}m_{01}m_{10}$ appear to be affected by the nature of the dataset.

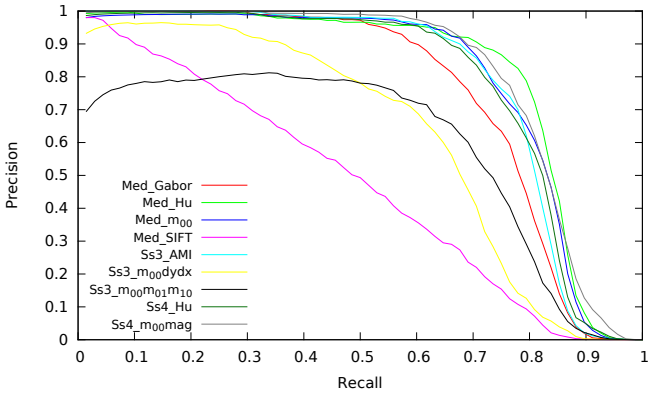
5.8.2 Ranking performance of handcrafted descriptors

Figure 5.3 shows the normalized ranking performance per transformation category for each handcrafted descriptor method. It is noticeable that there is no method that is the overall best. It clearly shows all methods have an easier time detecting real duplicates, versions with a reduced color depth,versions with an altered brightness, saturation or sharpness, and versions with logos, text or an outer frame.



(a) INRIA Copydays Dataset.

(b) Leeds Butterfly Dataset.



(c) UKBench Dataset.

Figure 5.2: The average precision-recall curves when queried 100 images per dataset over the entire collection of (near) duplicate images and noise images.

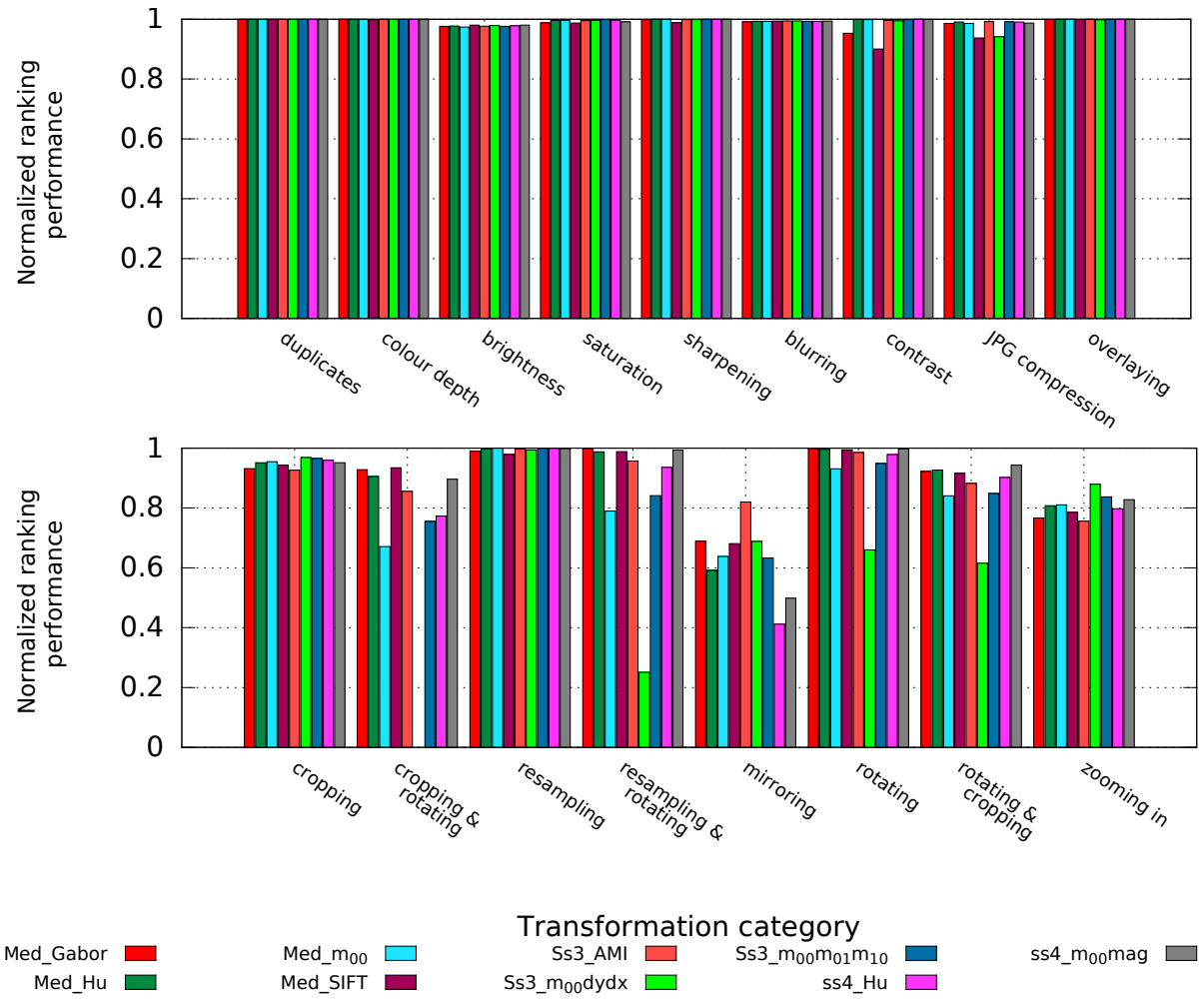


Figure 5.3: The average performance of the descriptor per transformation category for 100 query images from the Copydays dataset.

All handcrafted descriptors seem to perform poorly when describing a mirrored image. This is as expected due to the construction of the handcrafted descriptors. Images that have been cropped and rotated, rotated and cropped, and zoomed in on also prove to be more challenging for all methods.

$Ss4\_m_{00}mag$ is arguably the best performing self similarity descriptor. $Ss4\_Hu$ and $Ss3\_AMI$ only outperform $Ss4\_m_{00}mag$ slightly when detecting images that have an altered contrast or scale and the performance of $Ss3\_m_{00}dydx$ is simply not up to par with $Ss4\_m_{00}mag$, $Ss4\_Hu$ and $Ss3\_AMI$.

5.8.3 Applicability

The ideal method would list the n (near) duplicates of a query image I as the top n results. To be more precise, it would assign all $n+1$ images the exact same signature. By doing so, all (near) duplicates are grouped together. With a fixed descriptor size of 64, this paragon would thus allow for 2^{64} groupings and maintain the distinctiveness of $1.84e+19$ kinds of images, which is way more than the $1.8e+6$ images that were used for this thesis.

The aim of this experiment is to see how close the methods get to that ideal. Their applicability is measured by computing the average amount of retrieved images per hamming distance. The hamming distance between 2 binary descriptors is the exact number of positions at which the corresponding symbols between two descriptors are different.

By showing the average amount of images described per hamming distance, Figure 5.4 provides context for the results in Figure 5.3. Now, it becomes clear that some methods only seem to perform well in Figure 5.3, because they are not discriminative enough. *Med_SIFT* is an unsuitable descriptor for a dataset of this scale, since it lists an average of 812 candidates to be a (near) duplicate for a query image at hamming distance 0. We would also like to emphasize that not all descriptor methods extract descriptors of the same size.

When looking at hamming distance 0 in Figure 5.3 $Ss3\_m_{00}m_{01}m_{10}$ is one of the best performing descriptor for detecting all near duplicates generated by a photometric operation, cropping, and resampling. However, this is achieved by describing 0.00005% of the dataset, which is a lot more than most other descriptors and means that it is already including false positives. Despite its nonperformance in 4 categories, $Ss4\_Hu$ places second for a lot of categories at hamming distance 0. *Med_Hu* and *Med_m*00 hold up relatively well as well. *Med_Gabor* has relatively bad retrieval rates, but also retrieves just 23 images on average.

Med_Gabor, *Med_Hu* and Med_m_{00} do extremely well compared to other methods at a hamming distance of 0. However, *Med_Gabor* detects noticeably less near duplicates generated by JPEG compression, zooming in, or tampering with the contrast or size. *Med_Hu* and *Med_m*00 have a better recall at hamming distance 0 than *Med_Gabor*. However at a hamming distance of two these two methods are describing more than 300 images, whereas *Med_Gabor* describes 73 images. This is why it outperforms all the other median descriptors. The precision at K for any K retrieved images is simply better than the other median descriptors.

At a hamming distance of five $Ss3\_Ami$ and $Ss4\_m_{00}mag$ describe around 70 images on average, whereas the other methods have described an average between 111 and 5666 images. What was said for *Med_Gabor* is also the case for these methods. At this hamming distance not a single handcrafted descriptor manages a recall higher than 0.25 in categories with near duplicates generated by cropping and rotating, mirroring, and zooming in.

When looking at the photometric transformations, all methods barely gain any performance between hamming distances 10 and 15, with the noticeable exception of *Med_SIFT* and *Med_Gabor* when detecting near duplicates with a tampered contrast. For the first two methods, this is to be expected based on their features.

It is unfortunate to see how many performance gain is made between hamming distance 10 and 15 by all descriptors for the remaining transformations. It is at hamming distance 15 that a recall of 0.5 is finally achieved for the transformations that correspond to the categories that were determined the most challenging by the previous experiment, namely mirrored versions, cropped and rotated versions, rotated and cropped versions, and zoomed in versions.

Despite the good precision at K retrieved images for the descriptors *Med_Gabor*, $Ss4\_m_{00}mag$, $Ss3\_AMI$ and $Ss4\_Hu$, it can also be argued that these descriptors are actually very bad. As stated at the beginning of this

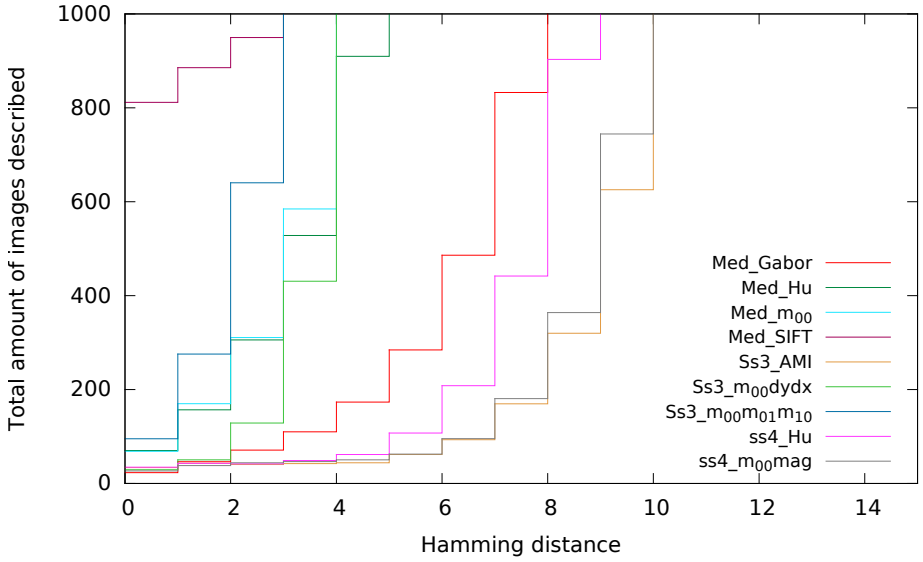


Figure 5.4: The total amount of images retrieved per hamming distance for all handcrafted descriptors.The averages were taken of 100 queried images from the Copydays dataset.

section, ideally, an image and its near duplicates are all assigned the same descriptor. Obviously, for a dataset of the size this experiment is run on, there is room for error, but if the dataset would have been larger, there is less leeway. This means that *Med_Hu* and *Med_*m_{00} are more fit descriptors for a larger scale than the other methods.

5.8.4 Composite descriptor

Multiple ways are imaginable to combine the forces of multiple descriptor methods. An composition could be based the proverb "many hands make light work", in which each method is a specialist in certain transformations. The composition tested in this experiment scores all images based on the output by 4 descriptor methods. This way there is some room for error for each individual method. The hypothesis is that they will increase robustness collectively.

The composite descriptor is conceptually very simple. The descriptor method outputs a vector composed out of the descriptors *Med_Gabor*, *Med_*m_{00}, *Med_Hu* and *Ss*4*_*m_{00}*mag*. To get the (near) duplicates, the top 300 results are retrieved per method. Then all results are assigned a score based on their abundance within the total number of results. If a certain image is returned by all 4 methods, it receives a score of four. Then these results are ordered and the top 100 are collected along with the top 100 results by some of the top descriptors from the previous experiments.

Figure 5.5 shows the average recall rate for the descriptor methods. Since there are 68 (near) duplicates per query image, there is a leeway of 32 results. The composite descriptor almost always performs better than the individual handcrafted descriptors. The composite descriptor underperforms detecting near duplicates generated by cropping and rotating, mirroring and zooming in, because it is composed of methods that also underperform in these categories. Only for the resampled and rotated, mirrored, and rotated images, it is

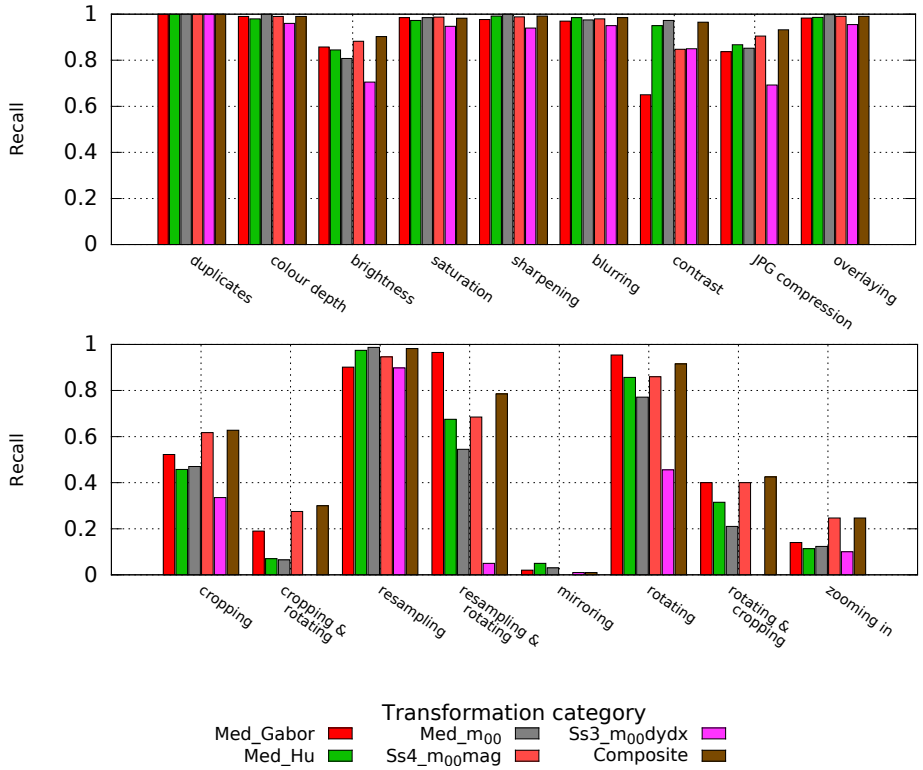


Figure 5.5: The average recall for the top 100 images returned by several methods. The queried images were 100 images that originated from the Copydays dataset.

Copydays	Butterfly	UKBench
0.777	0.767	0.772

Table 5.4: The area under curve (AUC) for the composite descriptor.

outperformed by *Med_Gabor* and $Ss4\_m_{00}mag$, which is reflected by the AUC score that is shown in Table 5.4.

5.8.5 Self-learning descriptor

The fifth test compares the performance of the self-learning descriptor, DHN, to the handcrafted descriptors on a small scale. This test was performed on solely the Copydays dataset and its (near) duplicates. The neural network extracts descriptors of 48 bits, which is far less than the handcrafted descriptors. This is why the recall of the descriptors is plotted against the ratio of the hamming distance to the length of the descriptors in Figure 5.6. This image shows that DHN detects all the near duplicates of a query image on a lower hamming distance than the handcrafted descriptors. It also shows that it is less discriminative than the handcrafted descriptors.

Figure 5.7 shows the amount of the dataset that is examined per ratio. At around ratio 0.8 DHN has examined 100% of the dataset. This is not an issue for a dataset of this size, but should be taken into account when examining datasets that are larger. However, this can be solved by finetuning the method. It was finetuned on a relative small portion of the UKBench and MIRFLICKR 1M dataset and tested on the Copydays dataset.

It is also interesting to compare this figure to Figure 5.4 which displays the average amount of images described

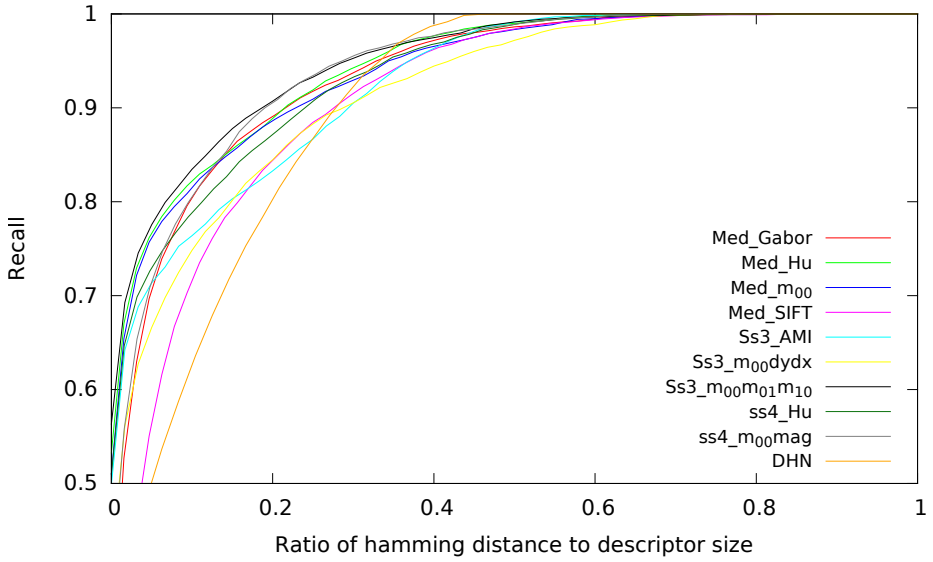


Figure 5.6: The average recall for the top 100 images returned by several methods. The queried images were 100 images that originated from the Copydays dataset.

per hamming distance for a very large scale. Figure 5.7 shows that *Med_SIFT* is more suitable for a dataset of a smaller size.

5.8.6 Computational performance

This experiment will give an indication of the time it takes per method to extract a descriptor from an image. The preconditions of this experiment were the same for all methods. All methods, but the composite descriptor, were tested on the same images of size 277×277 pixels. Note that this experiment is just a rough indicator of the performance, and does not consider the fine-tuning time of DHN.

The results of this experiments are shown in Figure 5.8. *Med_SIFT* is the worst performing method in this test as well, as it takes at least thrice as most other methods (DHN being the exception). Aside from the relatively long time it needed to describe an image, it also required excessive amounts of RAM memory. It was the only method to use more than 8 GB of RAM. It is also unfortunate that DHN takes an disproportionate amount of time to extract a descriptor.

Needless to say, the matching time for all the descriptors was very fast. It took mere seconds to scour the entire dataset, even when the composite descriptor was used.

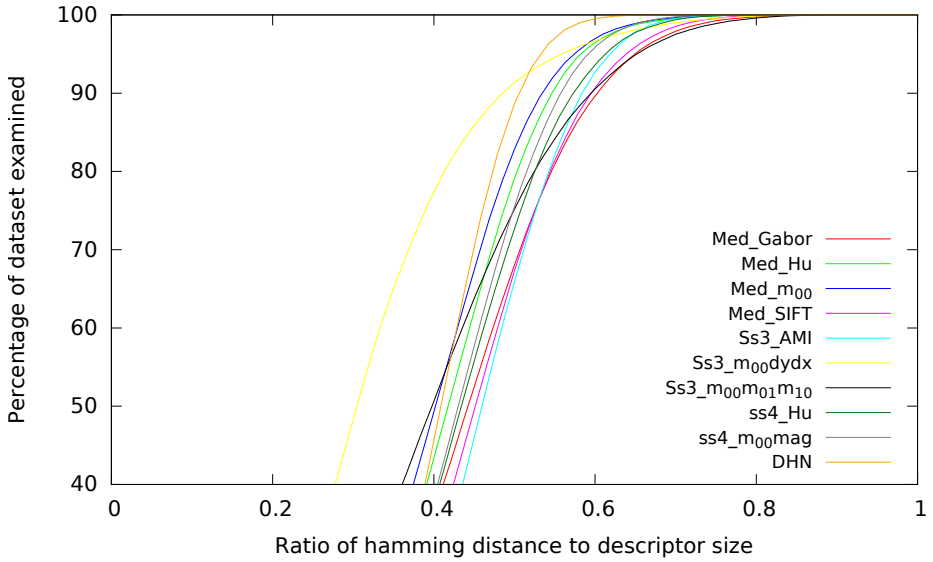


Figure 5.7: The average recall for the top 100 images returned by several methods. The queried images were 100 images that originated from the Copydays dataset.

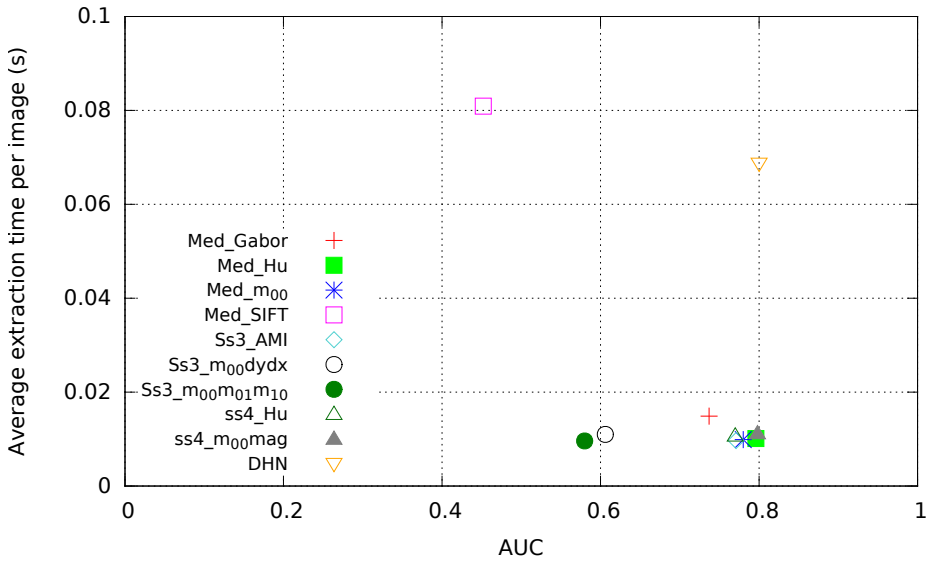


Figure 5.8: The average extraction time per image plotted against the AUC for each method. The average description time was calculated by extracting a descriptor for 500 images from the UKBench set resampled to 277×277 pixels.

Chapter 6

Conclusion and discussion

This thesis dealt with feature-based detection of (near) duplicate imagery via descriptors in such a way that is congruous with the demands stated in section 5.1. (Near) duplicates are forged versions of images, that may be the result of acts of copyright infringement or/and contribute to redundancy in a database.

After having evaluated existing methods that have been applied in other computer vision tasks, 11 methods were developed and a dataset of almost 1.8 million images was put together. Most of the methods applied a sliding window protocol in which values were compared to a global median or among grid cells. One method employed a convolutional neural network and 4 methods were combined into a composite descriptor. This selection of descriptor methods was based on the preceding analysis, but many other feasible methods are conceivable. The analysis may benefit the future development of (near) duplicate detection methods in making more informed decisions.

The current implementation of CNN-based descriptor was probably overfitted and utilized just 48 bits. The obvious downside of this method was the time required to finetune the network. The extraction time was acceptable. This method definitely deserves more attention for this purpose. However, this method might be too computationally complex for mobile devices. The performance gain of this implementation over the handcrafted descriptors was also limited, which was not conform expectations, as we hypothesised that the automatic feature engineering of the method would surpass the primitive handcrafted feature engineering.

In overall, based on the results from the 6 experiments, in cases were a low memory usage of 8 bytes or less and fast matching are of importance, our suggestion is to use $Ss4\_m_{00}mag$, as it seems to be the most well-rounded, handcrafted descriptor implemented. If the database is however several order of magnitudes bigger than the dataset tested on in this thesis or very homogeneous (such as a dataset of only MRI scans of a brain), we suggest the use of Med_Hu. The performance of the composite descriptor does not weigh up to the fact that it is 4 times the size of $Ss4\_m_{00}mag$. However, many more compositions are imaginable for a composite descriptor, and are also interesting for future research.

This thesis investigated whether it was possible to pose such a hard constraint on the storage load of a

descriptor while still getting good results. The answer is yes. However, the average rank at which all kinds of duplicates were detected was never perfect. There are still too many false positives, and cropped near duplicates pose a major problem. Fortunately, there is a lot of room for improvement.

Chapter 7

Limitations and future work

This thesis demonstrated the potential of a multitude of descriptors to efficiently extract very compact descriptors for the purpose of (near) duplicate detection. There are still many opportunities for extending the scope of this thesis. This chapter presents some directions and possible improvements on the current work.

For the handcrafted descriptors, we aimed to achieve a lot of invariance in the preprocessing phase. It is this phase of the implementation that falls short, as we suspect a lot more performance gain can be achieved by focussing more on this. When looking solely at the results for rotated images, not all rotations are found by the methods, but the preprocessing did account for brightness, saturation, sharpening, blurring, contrast and JPG compression. We would like to add that in the first stages of testing multiple color spaces were examined ad hoc, and the brightness component of the YUV color space and the saturation component of the HSV color space did not result in performance gain over the grayscaled image.

It should be relatively easy to improve the performance of all handcrafted descriptors for mirrored versions in the preprocessing phase, for example by flipping the image if the value of a certain metric on the left half of the image is higher than the value on the right half of the image.

The results from the test for the resistance against homogeneity and from the test for applicability were crudely combined in Figure 7.1 to show the ability of the descriptors to adapt to a bigger scale and to more homogeneity. Knuckling down on the adaptability of the descriptors is an interesting topic for future research. Note that the already fast matching time could be improved by advanced indexing and data structures.

Obviously there is also lot of performance to be gained for self-learning methods for the purpose of (near) duplicate detection with compact descriptors. This would require more research. The self-learning method implemented here was only tested on a small scale, and is promising to look at for bigger scales. Semi-supervised learning is also of great interest for this purpose since this might give an even higher accuracy, as it can be trained on unlabeled data as well. In practice, this might allow the method to detect (near) duplicates generated by transformations not present in the labeled dataset. It might also be a good idea to combine a neural network with a handcrafted descriptor.

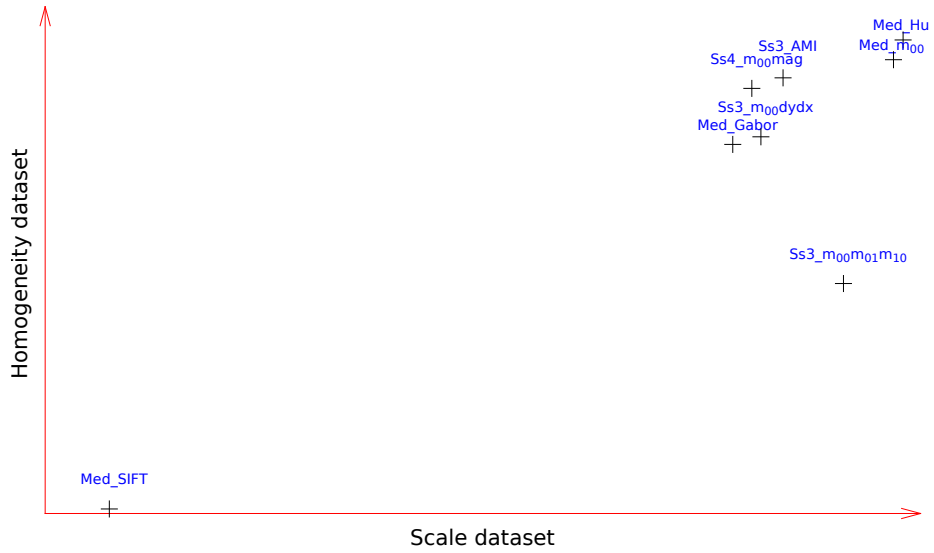


Figure 7.1: A crude diagram of the adaptability against the scale and the homogeneity of a dataset of all descriptors except the composite descriptor. The x-coordinate is determined my the difference of the maximum and minimum recall in Table 5.3 and the y-coordinate is determined by the absolute value that is the result from the total number of transformations, 68, minus the average of all the recall rates for hamming distance 0 times the amount of images retrieved for this hamming distance. The higher a method is, the more robust it is against more homogeneity. The more right a descriptor is, the more robust it is hypothesized to be against a bigger dataset.

Bibliography

[AI06] A. Andoni and P. Indyk. Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions. In *2006 47th Annual IEEE Symposium on Foundations of Computer Science (FOCS'06)*, pages 459–468, Oct 2006.

[AI08] A. Andoni and P. Indyk. Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions. *Commun. ACM*, 51(1):117–122, January 2008.

[BFPR14] S. Battiato, G. M. Farinella, G. Puglisi, and D. Ravì. Aligning codebooks for near duplicate image detection. *Multimedia Tools and Applications*, 72(2):1483–1506, 2014.

[Bra00] G. Bradski. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 2000.

[BS09] L. Bo and C. Sminchisescu. Efficient match kernel between sets of features for visual recognition. In *Advances in neural information processing systems*, pages 135–143, 2009.

[CCL05] H. Chen, H. Chang, and T. Liu. Local discriminant embedding and its variants. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 2, pages 846–853 vol. 2, June 2005.

[CLSF10] Michael Calonder, Vincent Lepetit, Christoph Strecha, and Pascal Fua. *BRIEF: Binary Robust Independent Elementary Features*, pages 778–792. Springer Berlin Heidelberg, Berlin, Heidelberg, 2010.

[CLVZ11] K. Chatfield, V. Lempitsky, A. Vedaldi, and A. Zisserman. The devil is in the details: an evaluation of recent feature encoding methods. In *British Machine Vision Conference*, 2011.

[DB12] F. Dornaika and A. Bosaghzadeh. Generalized local discriminant embedding for face recognition. In *Proceedings of the 9th International Conference on Image Analysis and Recognition - Volume Part II*, ICIAR'12, pages 90–97, Berlin, Heidelberg, 2012. Springer-Verlag.

[DIIM04] M. Datar, N. Immorlica, P. Indyk, and V. S. Mirrokni. Locality-sensitive hashing scheme based on p-stable distributions. In *Proceedings of the Twentieth Annual Symposium on Computational Geometry*, SCG '04, pages 253–262, New York, NY, USA, 2004. ACM.

[FK16] S. Fedorov and O. Kacher. Large scale near-duplicate image retrieval using triples of adjacent ranked features (TARF) with embedded geometric information. *CoRR*, abs/1603.06093, 2016.

[FS93] J. Flusser and T. Suk. Pattern recognition by affine moment invariants. *Pattern Recognition*, 26(1):167 – 174, 1993.

[Fuk13] K. Fukunaga. *Introduction to statistical pattern recognition*. Academic press, 2013.

[FZST07] J. J. Foo, J. Zobel, R. Sinha, and S. M. M. Tahaghoghi. Detection of near-duplicate images for web search. In *Proceedings of the 6th ACM International Conference on Image and Video Retrieval*, CIVR '07, pages 557–564, New York, NY, USA, 2007. ACM.

[GIM99] A. Gionis, P. Indyk, and R. Motwani. Similarity search in high dimensions via hashing. In *Proceedings of the 25th International Conference on Very Large Data Bases*, VLDB '99, pages 518–529, San Francisco, CA, USA, 1999. Morgan Kaufmann Publishers Inc.

[GL11] Y. Gong and S. Lazebnik. Iterative quantization: A procrustean approach to learning binary codes. In *CVPR 2011*, pages 817–824, June 2011.

[HCC+09] Y. Hu, X. Cheng, L. T. Chia, X. Xie, D. Rajan, and A. H. Tan. Coherent phrase model for efficient image near-duplicate retrieval. *IEEE Transactions on Multimedia*, 11(8):1434–1445, Dec 2009.

[HL08] M. J. Huiskes and M. S. K. Lew. The mir flickr retrieval evaluation. In *Proceedings of the 1st ACM International Conference on Multimedia Information Retrieval*, MIR '08, pages 39–43, New York, NY, USA, 2008. ACM.

[Hu62] M. Hu. Visual pattern recognition by moment invariants. *IRE Transactions on Information Theory*, 8(2):179–187, February 1962.

[JDS08] H. Jégou, M. Douze, and C. Schmid. Hamming embedding and weak geometric consistency for large scale image search. In *Proceedings of the 10th European Conference on Computer Vision: Part I*, ECCV '08, pages 304–317, Berlin, Heidelberg, 2008. Springer-Verlag.

[JDS11] H. Jégou, M. Douze, and C. Schmid. Product quantization for nearest neighbor search. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(1):117–128, Jan 2011.

[JDSP10] H. Jégou, M. Douze, C. Schmid, and P. Prez. Aggregating local descriptors into a compact image representation. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 3304–3311, June 2010.

[JPD+12] H. Jégou, F. Perronnin, M. Douze, J. Snchez, P. Prez, and C. Schmid. Aggregating local image descriptors into compact codes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(9):1704–1716, Sept 2012.

[KG09] B. Kulis and K. Grauman. Kernelized locality-sensitive hashing for scalable image search. In *2009 IEEE 12th International Conference on Computer Vision*, pages 2130–2137, Sept 2009.

[KS04] Y. Ke and R. Sukthankar. Pca-sift: a more distinctive representation for local image descriptors. In *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004. CVPR 2004.*, volume 2, pages II–506–II–513 Vol.2, June 2004.

[KSH12] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 1097–1105. Curran Associates, Inc., 2012.

[LCS11] Stefan Leutenegger, Margarita Chli, and Roland Y. Siegwart. Brisk: Binary robust invariant scalable keypoints. In *Proceedings of the 2011 International Conference on Computer Vision*, ICCV '11, pages 2548–2555, Washington, DC, USA, 2011. IEEE Computer Society.

[LJC+13] Y. Lin, R. Jin, D. Cai, S. Yan, and X. Li. Compressed hashing. In *2013 IEEE Conference on Computer Vision and Pattern Recognition*, pages 446–451, June 2013.

[LKF05] C. L. Liu, M. Koga, and H. Fujisawa. Gabor feature extraction for character recognition: comparison with gradient feature. In *Eighth International Conference on Document Analysis and Recognition (ICDAR'05)*, pages 121–125 Vol. 1, Aug 2005.

[LLCZ16] K. Lin, J. Lu, C. S. Chen, and J. Zhou. Learning compact binary descriptors with unsupervised deep neural networks. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1183–1192, June 2016.

[Low99] D. G. Lowe. Object recognition from local scale-invariant features. In *Proceedings of the International Conference on Computer Vision-Volume 2 - Volume 2*, ICCV '99, pages 1150–, Washington, DC, USA, 1999. IEEE Computer Society.

[MK01] A. M. Martinez and A. C. Kak. Pca versus lda. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(2):228–233, Feb 2001.

[MM96] B. S. Manjunath and W. Y. Ma. Texture features for browsing and retrieval of image data. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(8):837–842, Aug 1996.

[MRW+99] S. Mika, G. Ratsch, J. Weston, B. Scholkopf, and K. R. Mullers. Fisher discriminant analysis with kernels. In *Neural Networks for Signal Processing IX: Proceedings of the 1999 IEEE Signal Processing Society Workshop (Cat. No.98TH8468)*, pages 41–48, Aug 1999.

[NS06] D. Nistér and H. Stewénius. Scalable recognition with a vocabulary tree. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages 2161–2168, June 2006. oral presentation.

[OLYL14] X. Y. Ou, H. F. Ling, L. Y. Yan, and M. L. Liu. Convolutional neural codes for image retrieval. In *Signal and Information Processing Association Annual Summit and Conference (APSIPA), 2014 Asia-Pacific*, pages 1–10, Dec 2014.

[PCM05] L. Penna, A. Clark, and G. Mohay. Challenges of automating the detection of paedophile activity on the internet. In *First International Workshop on Systematic Approaches to Digital Forensic Engineering (SADFE'05)*, pages 206–220, Nov 2005.

[PD07] F. Perronnin and C. Dance. Fisher kernels on visual vocabularies for image categorization. In *2007 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, June 2007.

[PLSP10] F. Perronnin, Y. Liu, J. Snchez, and H. Poirier. Large-scale image retrieval with compressed fisher vectors. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 3384–3391, June 2010.

[QMC05] A. Qamra, Y. Meng, and E. Y. Chang. Enhanced perceptual distance functions and indexing for image replica recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(3):379–391, March 2005.

[RASC14] A. S. Razavian, H. Azizpour, J. Sullivan, and S. Carlsson. CNN features off-the-shelf: an astounding baseline for recognition. *CoRR*, abs/1403.6382, 2014.

[RS00] S. T. Roweis and L. K. Saul. Nonlinear dimensionality reduction by locally linear embedding. *science*, 290(5500):2323–2326, 2000.

[SH09] R. Salakhutdinov and G. Hinton. Semantic hashing. *Int. J. Approx. Reasoning*, 50(7):969–978, July 2009.

[SSM98] B. Schölkopf, A. Smola, and K. Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Comput.*, 10(5):1299–1319, July 1998.

[SSTF+15] E. Simo-Serra, E. Trulls, L. Ferraz, I. Kokkinos, P. Fua, and F. Moreno-Noguer. Discriminative learning of deep convolutional feature point descriptors. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 118–126, Dec 2015.

[STE13] C. Szegedy, A. Toshev, and D. Erhan. Deep neural networks for object detection. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, pages 2553–2561. Curran Associates, Inc., 2013.

[TCLF12] T. Trzcinski, M. Christoudias, V. Lepetit, and P. Fua. Learning image descriptors with the boosting-trick. In *Proceedings of the 25th International Conference on Neural Information Processing Systems*, NIPS'12, pages 269–277, USA, 2012. Curran Associates Inc.

[TD11] Ü. Ç. Turhal and A. Duysak. An algorithm to minimize within-class scatter and to reduce common matrix dimension for image recognition. *Turkish Journal of Electrical Engineering & Computer Sciences*, 19(6):929–939, 2011.

[TDSL00] J. B. Tenenbaum, V. De Silva, and J. C. Langford. A global geometric framework for nonlinear dimensionality reduction. *science*, 290(5500):2319–2323, 2000.

[TFW08] A. Torralba, R. Fergus, and Y. Weiss. Small codes and large image databases for recognition. In *2008 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, June 2008.

[THBL08] B. Thomee, M. J. Huiskes, E. Bakker, and M. S. K. Lew. Large scale image copy detection evaluation. In *Proceedings of the 1st ACM International Conference on Multimedia Information Retrieval*, MIR '08, pages 59–66, New York, NY, USA, 2008. ACM.

[THBL13] B. Thomee, M. J. Huiskes, E. M. Bakker, and M. S. K. Lew. An evaluation of content-based duplicate image detection methods for web search. In *2013 IEEE International Conference on Multimedia and Expo (ICME)*, pages 1–6, July 2013.

[WHB09] S. Winder, G. Hua, and M. Brown. Picking the best daisy. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 178–185, June 2009.

[WKC10] J. Wang, S. Kumar, and S. F. Chang. Semi-supervised hashing for scalable image retrieval. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 3424–3431, June 2010.

[WKIS09] Z. Wu, Q. Ke, M. Isard, and J. Sun. Bundling features for large scale partial-duplicate web image search. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 25–32, June 2009.

[WL16] S. Wu and M. S. K. Lew. Image correspondences matching using multiple features fusion. In *Computer Vision–ECCV 2016 Workshops*, pages 737–746. Springer, 2016.

[WME09] J. Wang, K. Markert, and M. Everingham. Learning models for object recognition from natural language descriptions. In *BMVC*, 2009.

[WSL+14] J. Wang, Y. Song, T. Leung, C. Rosenberg, J. Wang, J. Philbin, B. Chen, and Y. Wu. Learning fine-grained image similarity with deep ranking. In *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1386–1393, June 2014.

[WTF09] Yair Weiss, Antonio Torralba, and Rob Fergus. Spectral hashing. In D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, editors, *Advances in Neural Information Processing Systems 21*, pages 1753–1760. Curran Associates, Inc., 2009.

[YC12] X. Yang and K. Cheng. Ldb: An ultra-fast feature for scalable augmented reality on mobile devices. In *2012 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pages 49–57, Nov 2012.

[YCBL14] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson. How transferable are features in deep neural networks? In *Proceedings of the 27th International Conference on Neural Information Processing Systems*, NIPS'14, pages 3320–3328, Cambridge, MA, USA, 2014. MIT Press.

[ZK15] S. Zagoruyko and N. Komodakis. Learning to compare image patches via convolutional neural networks. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4353–4361, June 2015.

[ZLWC16] H. Zhu, M. Long, J. Wang, and Y. Cao. Deep hashing network for efficient similarity retrieval. In *AAAI*, 2016.