UPTEC X 08 035
Examensarbete 20 p
September 2008

Improving face recognition using textual annotation

Martin Norling

UPPSALA
UNIVERSITET

Bioinformatics Engineering Program

Uppsala University School of Engineering

UPTEC X 08 035	Date of issue 2008-09

Author

Martin Norling

Title (English)

Improving face recognition using textual annotation

Title (Swedish)

Abstract

In this thesis project I have designed an application combining a traditional search engine and one of the most well known face recognition algorithms to date. The purpose of this design is to attempt to improve search relevancy for queries related to celebrities and other well known persons. The application includes both face detection and face recognition. The classical eigenfaces algorithm is used in the face recognition. This algorithm turns out to perform quite badly as the algorithm requires faces to be aligned to its references which is often not the case with arbitrary images of the internet.

Keywords

Face Recognition, Face Detection, Eigenfaces, Search Engine.

Supervisors

Pablo Belin
PicSearch AB

Scientific reviewer

Ewert Bengtsson
Centre for Image Analysis, Uppsala University

Project name	Sponsors
Language **English**	Security
ISSN 1401-2138	Classification
Supplementary bibliographical information	Pages **31**

Biology Education Centre Biomedical Center Husargatan 3 Uppsala
Box 592 S-75124 Uppsala Tel +46 (0)18 4710000 Fax +46 (0)18 555217

Improving Face Recognition Using Textual Annotation

Martin Norling

Sammanfattning

I slutet av 1990-talet började internet växa sig för stort och utvecklas för snabbt för att en vanlig användare skulle kunna följa utvecklingen trots hjälp från portaler och länksidor. Detta gjorde att sökmotorer fick en allt större och viktigare plats för människors internetanvändande.

De flesta moderna sökmotorer har specialtjänster för sökningar efter bilder, men dessa tjänster är oftast baserade på text som hittas i samband med bilder samt på filnamn. I det här projektet undersöks om en variant där en klassisk ansiktsigenkänningsalgoritm kombinerad med den vanliga stödordssökningen ger bättre resultat än en sökmotor som bara är baserad på text.

Programmet som utvecklades använder Eigenfaces-algoritmen, en av de första ansiktsigenkänningsalgoritmerna som har fördelen att den kan beräknas mycket snabbt. För ansiktsdetektering användes OpenCV-paketets ansiktedetekterare. Som dataset användes 4940 bilder med nyckelord ur PicSearchs databas.

Tyvärr visade det sig att det inte gick att få tolkningsbar detekteringsinformation från ansiktsdetekteraren. Detta gjorde att ansiktsigenkänningen presterade dåligt, då det är ett krav för eigenfacesalgoritmen att ansikten skall vara normerade med avseende på ögon- och munkoordinater.

Om projektet utökades med en mer lämpad ansiktsdetekterare, helst en som kunde arbeta samtidigt med stödordsgenereraren istället för efter, skulle bättre resultat kunna erhållas.

Examensarbete 20p
Civilingenjörsprogrammet Bioinformatik
Uppsala Universitet september 2008

Contents

1 Summary of variables and measures

This section is only to easily be able to find the definitions of variables and measures used later in the text.

1.1 Variables

θ The variable θ is used in the face recognition to set the minimum euclidean distance between a detected face and a reference to classify the face as recognized.

α The variable α is used as a weight when the scores from the textual annotation is combined with the face recognition. This is done according to $FinalSearchIndex = (1-\alpha) \cdot TextualAnnotationSearchIndex + \alpha \cdot FaceRecognitionSearchIndex$. The formal definition can be found in equation 7

c The variable c is a cut-off constant that sets the minimum score value a result must have to be returned from a query. c affects the score according to equation 8

1.2 Measures

Precision The measure of precision is tied to the measure of recall. It describes the quality of the data returened from a query. A high precision means that mostly or only relevant data is returned from a query. Note that this might mean that only relevant data is returned but there may be further relevant data that is not returned. Precision is defined in equation 9.

Recall Recall is tied to the measure of precision. It describes how much of the relevant data in a corpus that is returned by a query. A high recall means that most of the data relevant to a query is returned. Note that it may also mean that non-relevant data is also returned. Recall is defined in equation 10.

F-measure The F-measure is a way of measuring overall performance in a system where precision and recall can be measured. The F-measure is the weighted harmonic mean of precision and recall. In this project the balanced F-measure is used, which assigns equal importance to precision and recall. The F-measure is defined in equation 11.

Sensitivity Sensitivity is a measure of how well a recognition system can detect a positive from a negative sample. Sensitivity is defined according to equation 12. Sensitivity is tied to the measure specificity.

Specificity Specificity is a measure of how well a recognition system can detect a negative sample from a positive sample. Specificity is defined according to equation 13. Specificity is tied to the measure sensitivity.

2 Introduction

At the end of the second millenia the internet grew rapidly, making it almost impossible for a user to navigate without the use of search engines or other kinds of index to help in locating information. The first search engine[1] appeared in 1990 and was called archie [2], a program that accesses and indexes anonymous ftp servers. From that spot searching continued to expand and improve with more and more sophisticated tools and more and more advanced computers. At the end of 2007 the most popular search engine was Google [6].

While most of todays search engines provide special search functions for images the search is still text based. Images are indexed by filename and by words associated with the image in the document where it was found. Ideally an image would instead be indexed from what is depicted. While research in the area of computer vision has been going on for many decades it is still far from the point where a computer can successfully analyze the content of a completely random image. In this project I will try to evaluate whether an approach where textual annotation is combined with face detection and recognition can result in greater search relevancy compared to purely textually annotated searching.

2.1 Similar Work

One interesting project published in 1995 was jacobs et. al "Fast Multiresolution Image Querying" [5] where a user could search a database of 20k images by drawing an image by hand and the program would give you a list of images that was similar to what you had drawn. While this is a very interesting approach it does not deal with the problem of converting a textual search string into image data but similar approaches could be used to refine a search by iteratively searching for the best result in previous searches.

Another project which was published in 2002, lienhart et. al "Classifying images on the web automatically" [12] did fairly well in classifying random images from the internet into three classes: photos, photo-like graphics and posters/slides/comics. An implementation like this could be used, just like the face recognition handled in this project, to refine search results in an ordinary search engine.

A project that is similar to mine in that it tries to identify people in arbitrary images of the internet is Polar Rose [3]. This company hands out a browser plug-in that scans images in pages you browse and if it contains faces it puts

[1] Whether Archie was indeed the first search engine is mostly a question of defining the concept of a search engine and that is outside the scope of this thesis.

a small rose where the pinhole of their shirt would be. If you click the rose the plug-in tries to identify the person using a database of known people. The project is currently in beta testing so no direct results of how well it performs is available at the moment.

3 Application Design

The first part of this project would be to design a program that would perform in a way that was theoretically equivalent to a real search engine. A real search engine is a huge system often spanning several servers with a large number of crawlers working simultaneously to index the web and keep the indexed parts up to date. The system must also have functions for continuously sorting and adding new data to, as well as removing out of date data from, the search index. Everything to keep the index in an ordered state to increase search relevancy and to speed up querying.

Such a system can of course not be designed and implemented by a single individual over the course of a thesis project. Still, an application design that would be equivalent to an actual search engine was needed. To keep the essence of a search engine the index generation needs to be completely separated from querying. In the case where a continuously updated search index is completely up to date it is equal to a search index that has been pre-generated and where the search space isn't changing. This can be used to design an evaluation application which performs equally to a search engine but can be run fast on a single computer.

Since the purpose of this project is to attempt to combine facial recognition with a textually annotated image search engine, computer vision technology becomes an important asset.

3.1 Face Detection

The Intel Open Source Computer Vision Library [7] includes a pre-trained face detection algorithm [1] that is used in this project. The face detection uses a gradually growing sliding window to detect faces in each part of an image using an ADAboosted haar-wavelet classifier. This face detection package were used since it performs as well as one can expect from a free software package and it is easy to implement into the application.

In this implementation the face detection is controlled by two constants. The first one is a window scaling constant set so that the sliding detection window is to grow by 5% each pass. The second constant is a counter that specifies the number of times a part of the image has to be classified as a

potential face by the sliding windows to be classified as a true face. In this implementation a minimum of four detections were used.

The main drawbacks of the face detection is that it can only detect faces that are mostly facing the camera and are upright in the image and as this is a pre-trained learning system it has some of the drawbacks often associated with learning systems. Primarily it does not give interpretable results. It detects faces due to being trained to do so but the classification rules are extremely abstract, being 212 decision tree stumps with different weights. One effect of this is also that it will not give eye- mouth- or nose-coordinates which generally simply face recognition.

Note Detection of rotated faces could be achieved by rotating the sliding window as well as expanding it but that would increase detection time by one degree as a rotation step would be added to the analysis. As most faces in images are in fact upright this did not seem cost effective and was thus left outside the implementation.

3.2 Face Recognition

The eigenfaces algorithm [10] is considered to be one of the first successful face recognition algorithms. Facial recognition has improved in many ways since then but the algorithm is fast and simple which allowed me to implement and evalutate the algorithm within the time offered by this project. The algorithm is also very scalable once it is initialized.

3.2.1 PCA

The eigenfaces algorithm is strongly related to principal component analysis (PCA) and this also means that it has the strengths and weaknesses of PCA. PCA is a way of finding patterns in data and trying to separate those patterns from each other. It can also be used to filter out the noise that is always present in empirical data. PCA finds the linear combination of the original basis that captures the most of the variance of the data in least square terms. This is useful for clarification of data and also for dimensionality reduction as most of the variance will be in the first principal components, making it possible to approximate the original data well in a lower dimensional space. In many cases the principal components containing the least variance is considered to be noise and with this assumption PCA can also be used for noise reduction. PCA is made simple and efficient by making a few assumptions about the data. If these assumptions are not fulfilled PCA will not give reliable results. [14, 13]

1. Linearity As PCA is basically a change of basis to a linear orthogonal base maximizing variance, it must assume that the data is linear to begin with. PCA can be extended to nonlinearity but is then known as *kernel PCA*.

2. The distribution can be described by mean and variance PCA only takes mean and variance in consideration and because of this only data belonging to a gaussian/normal distributions can be properly described. This also guarantees that the signal to noise ratio and covariance matrix fully characterize the noise and redundancies.

3. Important dynamics have large variance This assumption states that large variation in the data should mean that it is important and that noise should be the lower variances.

4. The important components are orthogonal This simplification makes PCA soluable with linear algebra decomposition techniques.

The PCA algorithm is quite straight-forward. First a variance-covariance matrix C is calculated. Then the eigenvectors and eigenvalues of C. These eigenvectors are the principal components in the new base and the corresponding eigenvalues are their lengths. The eigenvalues are directly proportional to how much of the variance is contained within each eigenvector [14, 13]. In this project it is important to chose enough eigenvectors to describe faces correctly and hopefully be able to reduce some of the noise (ie. white noise, non-uniform lighting and face orientation) that will be in many, if not all, of the images.

3.2.2 Eigenfaces

The theory of the eigenface approach is to take a large number of images that are normalized so that they are of equal size, equal lighting conditions and have mouth and eyes aligned and find the most important features of these faces using PCA. If the training set is representative of all the facial attributes that can be, then any face can be described as a linear combination of the principal components.

Another result of this is that the description of a face can be stored as an array of real values representing the weights of the eigenvectors, saving memory compared to storing a reference image. A summary of the algorithm

can be found in table 1. The fact that eigendecomposition of images is very fast is important since it helps preserve the scalability of the application.

1	Prepare a training set $T = \{t_1, t_2, ..., t_n\}$
2	Let M be the mean of T according to equation 1.
3	Let A be the mean centered training set according to equation 2.
4	Calculate C as the variance-covariance matrix of A according to equation 3
5	Calculate the eigenvalues and eigenvectors of C
6	Choose the number p of principal components ϵ that you wish to keep in your model

Table 1: A summary of the Eigenfaces algorithm.

$$M = \frac{i}{n}\sum_{i=1}^{n} t_i \tag{1}$$

$$A_i = T_i - M \tag{2}$$

$$C = \frac{1}{n}\sum_{i=1}^{n} A_i A_i^T \tag{3}$$

When the eigenfaces has been generated a reference matrix is made to hold information about faces the search engine should recognize. This matrix will have p cloumns, one for each principal component and one row for each face to describe. It will contain the weights of the eigenfaces used to describe the face. Once this "celebrity matrix" is generated the images in the search space will be indexed. To classify an image F_{new} it is transformed into an eigenface representation according to equations 4 and 5.

$$\omega_i = \epsilon_i^T (F_{new} - M), i = 1...p \tag{4}$$

$$\Omega_{new}^T = [\omega_1\ \omega_2\ ...\ \omega_p] \tag{5}$$

Where ω_i is the eigenface component corresponding to ϵ_i making Ω_{new} the weight vector of F_{new}. Ω_{new} can then be compared to the stored faces in the celebrity matrix using euclidean distance. To determine if this is a known face a threshold value θ is used. The exact value of θ can be altered to give a satisfying recall and precision depending on the application. The θ value

is often quite large compared to the other constants and values between 100 and 200 are common in classifying the image used in training[2].

3.3 The Textual Search Model

The textual search of this project resembles a traditional search engine. In this application the textual search is very simplified as real world data could be extracted and used from the PicSearch search engine. This means that no analysis of text needs to be done and instead pre-formated textually annotated images are used. Each image has a number of keywords describing it. These keywords are listed in order from most important to least important in how well they describe the image. This order is converted into weights as described in formula 6, where i is the keywords index and j is the number of keywords associated with keyword k. In a real world application some other method of generating weights would be used, but real world weights were not available during this project.

$$w(k_i) = \frac{1 + j - i}{\sum_{c=0}^{j} c} \tag{6}$$

All images, their keywords and the keywords' weights are used to generate an inverted search index that incorporates all the information of the textual annotation. To reduce the size of this index stop words (e.g. I, in, and, to, etc.) are removed and keywords are stemmed to avoid unnecessary repeats of the same words with different endings.

3.4 Combining the Models

In the evaluation application design proposed by this project a single program with two modes are used instead of two different programs, as would be the case in a live application. The two modes are initialization and querying. The initialization mode is in itself split further into two separate processes, textual initialization and image analysis. Figure 1 shows a graphic model of the project application design.

The evaluation application is controlled by three variables. α, θ and c. α is the weight constant used when the facial recognition results is combined with the textual annotation results. This index combination is performed

[2]That there is a difference is due to the reference images being scaled using bicubic interpolation for high image quality and the test image being scaled using bilinear interpolation for speed.

according to equation 7 and would in a final application be part of initialization. To allow α to be changed without re-initializing the index the index are now held separate and are combined for each query. θ controls how the facial recognition performs by setting the maximum euclidean distance between the reference image and the analyzed image in the search space. Finally c is a cut-off constant representing the minimum score in the interval a query needs to be classified as correct. The variable c affects the application according to equation 8. In this equation S is the score function and I_i is image i.

$$I_s = (1 - \alpha) \cdot I_t + \alpha \cdot I_f \tag{7}$$

$$S(I_i) = \begin{cases} I_s & I_s > c \\ o & I_s \leq c \end{cases} \tag{8}$$

Worth noting in the design is that stemming is not an active function in the program but as the queries and keywords are pre-formated the performance is equal to what it would be if stemming was used.

3.5 Scalability

While the implementation used in this project was optimised for changing variables and gathering data for statistical use, the application design itself should theoretically be very scalable. The actual querying is done using an inverse search index in the same way as without face recognition. The only part that has changed is the way the search index is generated.

Initialization using this algorithm is by far more time consuming than in the text only scenario though. The amount of data that needs to be processed is in many cases drastically increased as images need to be analyzed in comparison to text only.

The actual face recognition can be done in linear time using this algorithm as eigen decomposition is very simple once the eigenfaces have been calculated. The face detection is more time consuming. It works by analysing each part of an image using a sliding window that is gradually increased in size for each pass over the image. Each time an area is analyzed to see whether it is recognized as a potential face, and each time a potential face is found all previous potential faces are compared to see if they are in the same area of the image. This is the most time consuming step of this algorithm and should generaly be tweaked for speed rather than accuracy in a large scale application.

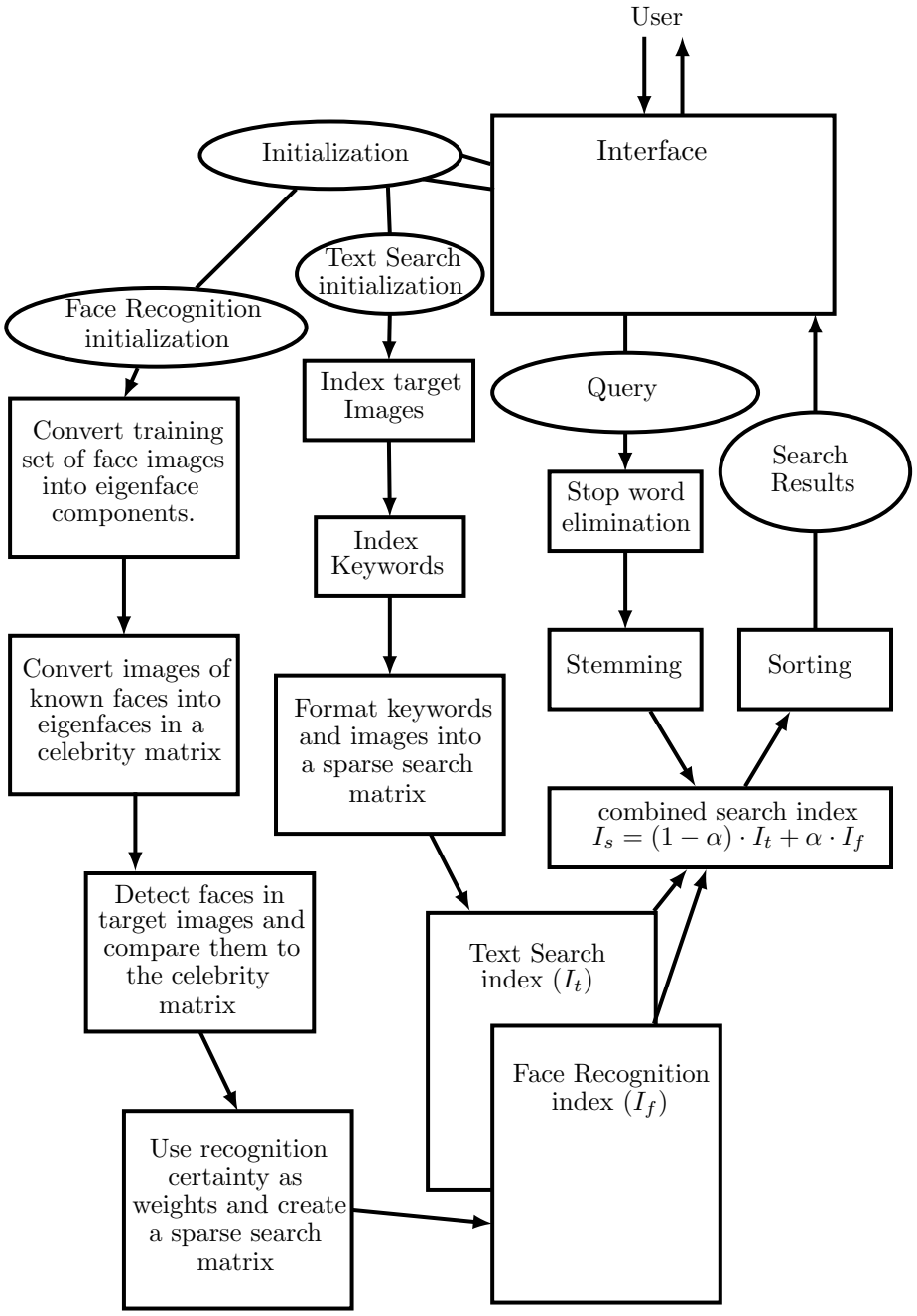

Figure 1: A flow chart describing the application design used in this project.

Note In a large scale application other kinds of both face recognition and face detection should be used as there are plenty of more efficient and more accurate algorithms that was outside the scope of this thesis project. See

section 5.6.3 for some examples of high-end products.

To evaluate further a number of test runs were done on a AMD Athlon^{TM}64 X2 4200+ Dual Core Processor with 2Gb of RAM. As expected indexing the images and generating the search matrices was by far the most time consuming step taking 20 minutes and 1 second. Once the search index have been created they can be loaded from disc in only 15 seconds. Once running the actual querying takes about 0.2 seconds.

Note These speeds are all far higher than optimal. For instance query time would immediately be halved if the index were concatenated into one instead of being recalculated every time. For the loading times, OpenCV saves matrices into xml, which is easy to handle but far from optimized for this kind of application. This implementation is done to easily change parameters and evaluate statistical performance, not primarily to measure speed.

4 Evaluation

Measuring performance in an application like this is of course dependent on what kind of performance a user would want from the application. In this case where a search engine is extended with face recognition one way of treating it would be to assume that only images containing faces would be interesting. Another would be to assume that all images can still valid even if they do not contain faces and that the face recognition is just an addition to increase performance in a part of the search space.

To measure performance one must first define the correct search results for every query to be evaluated. In doing this, a definition of what is correct must already be present. In this project I have made two classification rule sets.

Classification rule set A uses the wider definition of the examples above. An image can be correctly returned from a query even if it doesn't contain a detectable face. This definition was chosen since even though my test set is celebrities, not only images of themselves but also album covers, merchandise etc. can be considered interesting to an end user. This is usually how a real world search engine works but due to the way the test set is made in this project it becomes hard to draw hard conclusions from this rule set.

Classification rule set B uses the stricter definition and only images depicting the actual person is considered correct hits. This definition was chosen to see how well the face recognition performs in a test set where all correct images

could theoretically be identified using face recognition. This classification rule is more suited to evaluating this design than rule set A, but is less frequently used in a real world scenario.

4.1 Precision and Recall

To evaluate performance the measures of precision and recall are used. Both are measures which are commonly used in information retrieval and statistical classification. Precision is defined according to equation 9 and recall is defined according to equation 10 Where ξ is the returned information, ξ_{rel} is the relevant returned information and ω is the total relevant information in the corpus.

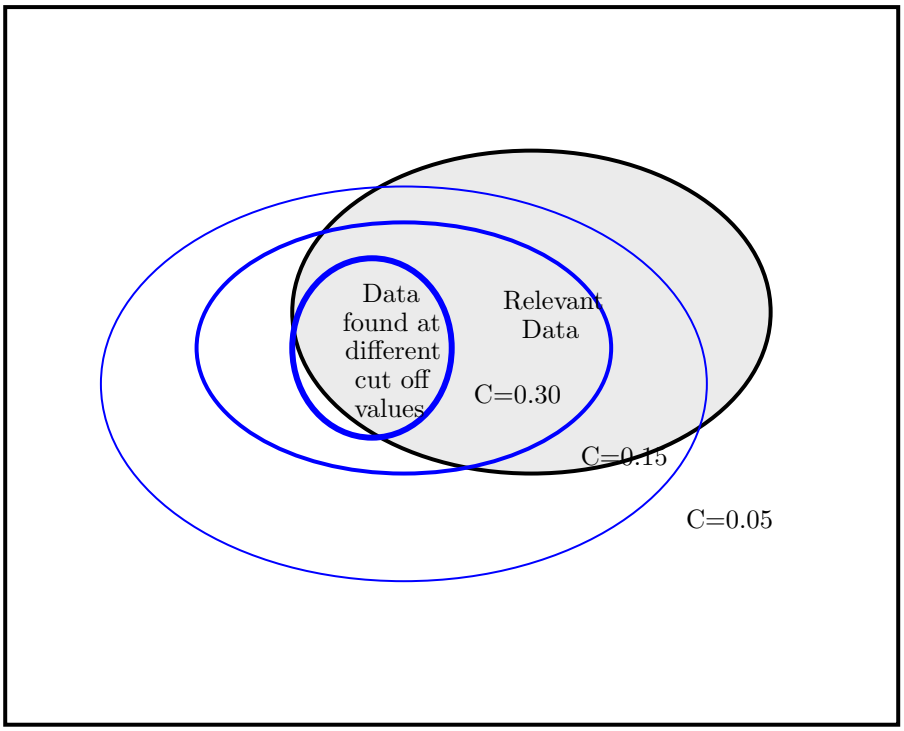

Figure 2: A simple illustration of the relationship between precision and recall.

$$\phi = \frac{\xi_{rel}}{\xi} \tag{9}$$

$$\rho = \frac{\xi_{rel}}{\omega} \tag{10}$$

The relationship between precision and recall is visualized in figure 2. The box represents the search space and the grey area represent the information of interest to a particular query and the blue ellipses represent the information returned to the user. As the cut-off value C gets higher, the blue circles become smaller, returning less information but generally more relevant information, thus increasing precision but lowering recall. In the same way recall is increased at the cost of lower precision when the cut-off value is lowered.

4.2 The F-measure

A system can be tuned to give high recall by allowing more data to be returned from each query and thus increasing the number of relevant images returned. This usually also increases the number of non-relevant images returned as well and in such it lowers precision. In the same way returning fewer images is likely to give higher precision at the cost of lower recall.

To measure overall performance the F-measure can be used. The F-measure is the weighted harmonic mean of precision and recall. I will use the balanced F-measure which assigns equal importance to both precision and recall. The balanced F-measure is defined according to equation 11.

$$F = \frac{2 \cdot \phi \cdot \rho}{\phi + \rho} \tag{11}$$

Note that the balanced F-measure is only interesting if a precision and recall is considered equally important. If one is considered more important than the other to a particular implementation, a weighted F-measure or another kind of performance measure should be used.

4.3 Sensitivity and Specificity

In the evaluation of face recognition and face detection the measures of sensitivity and specificity are used. Sensitivity is a measure of how well the system finds the positive cases in a study and specificity is a measure of how well it finds the negative cases. Sensitivity and specificity are defined according to equations 12 and 13 respectively.

$$Sensitivity = \frac{TP}{TP + FN} \tag{12}$$

$$Specificity = \frac{TN}{TN + FP} \tag{13}$$

4.4 Data sets

In this project three kinds of major data sets are used. The first one is a training set of representative face images used to train the eigenfaces algorithm, the second is a set of face references, chosen to represent the people who are to be recognized by the system. The third data set is a test set to evaluate the performance of the design.

Apart from these major data sets there are also two smaller, specialized data sets used. The first is a set of 100 non-face images and 100 face images used to evaluate the performance of the face detection. The other is a set of 20 correct face images and 100 incorrect face images used to evaluate the performance of the face recognition.

4.4.1 Training set

The eigenfaces training set used in this project is made out of images from the Yale Face Database [4]. The images in the data base are taken from nine different views but in this application only the frontal view images will be used. This is because the face detection performs best on frontal faces and the face reference data set is made from forward images. The system could easily be extended with profile or angled images as long as the appropriate classification rules and face references were added into the application. This would of course double the time used in detection and recognition. In [8, 9] they show that in many datasets the background plays a major role in classification. To avoid this error the images were cropped using the same face detection as is used in the final application so that only the face part is considered. The faces were also extracted from the reference images using the face detection algorithm in the same way as is used with the test set during evaluation. This is done to avoid errors due to differences in extraction.

The Yale face database has images from 64 different lighting conditions but in this application only uniformly lit images were used. An informal test was performed where all lighting conditions were used but it caused the application to be more sensitive to lighting conditions. As the application became better at approximating shadows, the shadows came to play a bigger part of the recognition. This caused lighting to be more important than facial attributes in many classification cases.

4.4.2 Face reference data set

The next kind of dataset needed is a set of reference images depicting people that the system should be able to recognize. These images were taken from the internet and care was taken to ensure that they were taken from a fronal angle, with no hair or clothing obscuring the face, neutral lighting and a neutral or smiling facial expression. The reference images were also compared to the test set to ensure that the same images was not used in training and classification as these images would be unfairly simple to classify correctly.

When reference images that fulfiled all the requirements had been found the face detection were used to extract the faces from the images, once again to ensure that no errors were introduced due to differences in extraction. These images were then resized to 50 x 50 pixels. This size was chosen to allow a good amount of detail while still keeping the design scalable. As many images on the internet, especially thumbnail images, are small in size, the faces found are often somewhere around 50 x 50 pixels. As the faces need to be scaled to the same size for the recognition to be performed using larger reference images would not have improved recognition.

4.4.3 Test Set

The test set is a set of textually annotated images. A list of 247 celebrities was assembled and for each of these celebrities the 20 highest ranked hits from the picsearch image search engine was extracted from their data base, including textual annotation but not including textual annotation weights. This resulted in a data set of 4940 images. When these images were downloaded 513 turned out to be unreachable and another 93 turned out to be non-images. The list of celebrities was also shortened to 244 due to three celebrities appearing twice with slight misspellings in their names. The resulting test set contains 4334 images.

Two classification rule sets were made to classify this data. Classification rule set A, where all images related to a query is considered correct and classification rule set B, where only images containing the correct person is considered correct hits. The test set was manually classified according to both classification rule sets with respect to 244 queries being the names of the celebrities that were used to query the PicSearch data base.

The evaluation of both rule set A and B is somewhat flawed in this implementation as it only works with images that has already been picked out by the textual annotation system. In a real world scenario the face recognition system would analyze all images that are found, just like the textual annotation generator does, not just those that has already passed. Because

of this the face recognition in this application does have a disadvantage in that if the textual annotator misses a relevant image the face recognition has no way to find it as it never gets the chance to analyze it.

Note This is especially true for rule set A. In this project rule set A is included since it describes how a modern search engine normally works, by returning information relevant to a query, not only images of the specific individual in the case of a celebrity search. Due to the fact that the test set used in this is picked from highest ranking results in a data base, not arbitrary images of the internet the main reason to include face recognition into such a system, to identify more relevant images, becomes void. In a real world scenario the face recognition and textual annotation systems would work together to classify images, in this case textual annotation classifies and face recognition tries to find flaws in that classification. In a rule set where images not containing face can be considered correct this step can not remove any images as they may still be relevant due to other criteria than faces and it can not add images since the test set is only images that are already accepted by the textual annotator. I decided to keep rule set A in the design since I feel that it is relevant due to being more what a real search engine would work than rule set B, but like this the only possibility to improve queries is to find images relevant to a query among the images that the textual annotator has assigned to another query. As the test set is taken from high ranking images this is of course very rare and the results from rule set A can thus not be expected to be improved significantly in this implementation.

When the data set and classification was finished the program was queried with each of the 244 queries using θ values of 200, 400, 600, 800, 1000, 1500, 2000, 3000, 4000, 5000, 6000, 7000, 8000, 9000, 10000, α values of 0.00 to 1.00 in 0.05 intervals and c-values of 0.00 to 0.30 in 0.05 intervals. This gives a total of 538.020 queries per classification rule set. Statistical results of the queries using rule set A and different values of c are presented in figures 11-14. Results from the queries using rule set B are presented in figures 16-20. Precision and recall was plotted in the same graphs to allow for more intuitive evaluation of the search engines behaviour resulting in four dimensional data in each plot. The fourth dimension is plotted using a color map[3].

Beside face recognition a series of tests was conducted where only face detection was used. Instead of comparing a detected face against a reference image it was considered to be any person the textual annotation claimed it to be. The face detection results are of course independent of the θ variable

[3]The decision to use four dimensional graphs was made due the large amount of graphs needed to properly describe the results in fewer dimensions.

since no comparison is done to the face references. Results from the face detection only tests can be found in figures 15 and 20. The detection only data is again plotted in four dimensions but uses C as color mapped dimension instead of θ.

Worth noting is that since this approach uses the textual annotation as a complement to the face detection only images with correct textual annotation can be returned from a query. This fact makes this approach more of a filter. With perfect performance this approach would turn the results from rule set A into rule set B, but it would not be able detect images classified with the wrong name as a true face recognition system could.

In this implementation face detection only performs just like it would in a real world scenario since this approach has to work on textually annotated data while face recognition is completely separate from textual annotation. This means that the face detection only results can not be compared directly to the face recognition results.

5 Performance

To measure the performance of this application, each of the individual parts that the system is built from is evaluated. First the face recognition and the face detection are evaluated using small controlled test sets to get a clearer image of how they perform compared to the much more complex image of their performance provided using the real world data in the test set. The performance of the textual annotation has not been tested in the same way as the indexing algorithm that has generated that data is not part of this project and is considered company property of PicSearch AB. An evaluation of how well the textual annotation performs on the test set using classification rules A and B can be found in figures 4 and 7 respectively. Lastly the performance of the combined method as well as the facial detection only method has also been evaluated.

5.1 Face recognition

To evaluate the performance of the face recognition software used in this project a simple study was made to see how well the program would classify images where a face could be detected. 20 images contained faces of the person that were to be recognized and 100 contained other faces. A total of 120 images were used. Results from this test can be seen in tables 2, 3 and 4 for θ values 600, 1200 and 2000 respectively. Note that in this test no face is recognized at $\theta = 600$ but in the runs using the test set $\theta = 600$ gives a noticable increase in recall and decrease in precision due to the many

faces being detected. That this effect cannot be seen here is because there are only one face to detect. With every face added to the recognition the chance of a face being recognised increases due to a larger area in the search space being considered a correct recognition. This stresses the importance of using a more controlled test to evaluate face recognition performance as well as the importance of understanding the dynamics of how the performance changes as the recognition task is altered by adding or removing reference faces.

	Face is in image	Face is not in image
Face recognized	0	0
Face not recognized	20	100

Table 2: Face detection performance at $\theta = 600$ on a data set of 20 correct face images and 100 incorrect face images.

	Face is in image	Face is not in image
Face recognized	6	6
Face not recognized	14	94

Table 3: Face detection performance at $\theta = 1200$ on a data set of 20 correct face images and 100 incorrect face images.

	Face is in image	Face is not in image
Face recognized	13	61
Face not recognized	7	39

Table 4: Face detection performance at $\theta = 2000$ on a data set of 20 correct face images and 100 incorrect face images.

Tables 2, 3 and 4 give a highest sensitivity of 0.65 (specificity 0.39) at $\theta = 2000$ and a highest specificity of 1.00 (sensitivity 0.00) at $\theta = 600$. As can be seen the face recognition tends to perform badly at all θ values used. This is likely due to the face recognition being dependent on having eyes and mouth coordinates normalized and as that can not be done it affects the recognition performance. In a normalized scenario performance should be expected to rise.

To illustrate performance in a normalized scenario another test was run using training data as true positives. performance from this test can be found in table 5.

Performance becomes ideal at a θ value as low as 200. This test is of course severely biased due to the use of the same images for training and recognition but it proves that the system will properly identify images that are normalized and similar to training data.

	Face is in image	Face is not in image
Face recognized	20	0
Face not recognized	0	100

Table 5: Face detection performance at $\theta = 200$ on a data set of 20 training set images and 100 incorrect face images.

5.2 Face detection

The face detection software was evalutated in a similar way to the face recognition. A data set of 100 images containing faces and 100 images not containing faces were used. Before an image was classified as a true positive the user was queried to see that the face part had been detected and not another part of the image. Faces were classified as true positives only if all faces in the image were detected and no other parts were incorrectly considered faces. Results from this test can be found in table 6. The results presented here are results using the same variables as were used in the application developed during this project, but by changing the variables the results can of course be changed. A good rule of thumb is that the smaller the window scaling variable the more sensitive the system gets and the more detections needed the more robust the system gets. Obviously every area will be detected more times with a smaller window scaling (as the window will pass the area more times) and as such the detection number should always be increased if the window scaling is decreased.

	Face in image	No Face in image
Face is found	69	8
No face is found	31	92

Table 6: Face detection performance on a data set of x face images and x non-face images.

This gives a sensitivity of 0.69 and a specificity of 0.92. The face detection performs a lot better than the face recognition in this test, but it is still far from perfect. Where images are noisy or containing many small details the face detection is prone to errors. The fact that it performs better than the face recognition is not surprising due to the more sophisticated algorithm and greater computational power used, but it still does not perform nearly as well as todays commercial products.

5.3 Classification rule set A

The test set is, as can be expected, very high quality with respect to the queries used when using classification rule set A . As can be seen in table

7 the application has almost perfect precision and recall using textual annotation only. By looking at the color mapping in figure 11 it can be seen that the algorithm gets a recall above 0.99 while precision drops under 0.10 as θ passes 3000, stressing the importance of keeping the facial recognition at high precision as it will easily drown the query in incorrect results. A highlight of this effect at $\alpha = 0.50$ can be seen in figure 3. This effect can be countered somewhat by using a higher cut-off value as can be seen in figures 12 to 14.

Details of how the application performs using textual annotation only and using face recognition only can be seen in figures 4 and 5. The textual annotation performs well in both precision and recall while the facial recognition only manages a high precision when combined with an extremely low recall and a high recall only at a very low precision. As the combined performance is a linear combination of these two figures (affected by c) performance over all values of α can easily be visualized. A summary of the original data results and the best results from the combined method considering the F-measure can be found in table 7.

While lots of interesting data about the behaviour of the application can be found in test set A, no real conclusions can be found in this data. As is discussed in section 4.4.3, the main purpose of using a design and classification rule set like this is lost when working with pre-textually annotated images while allowing non-face images to be classified as correct.

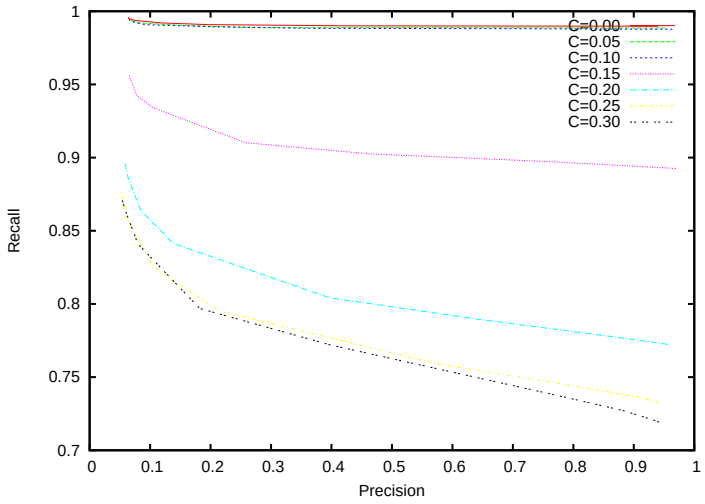

Figure 3: A graph highlighting performance when θ, the variable determining how strict the face recognition is, varies at $\alpha = 0.50$ using rule set A. Note that the graph is not a function y = f(x) but varies with the hidden variable θ between 1.00 (far left) and 0.00 (far right).

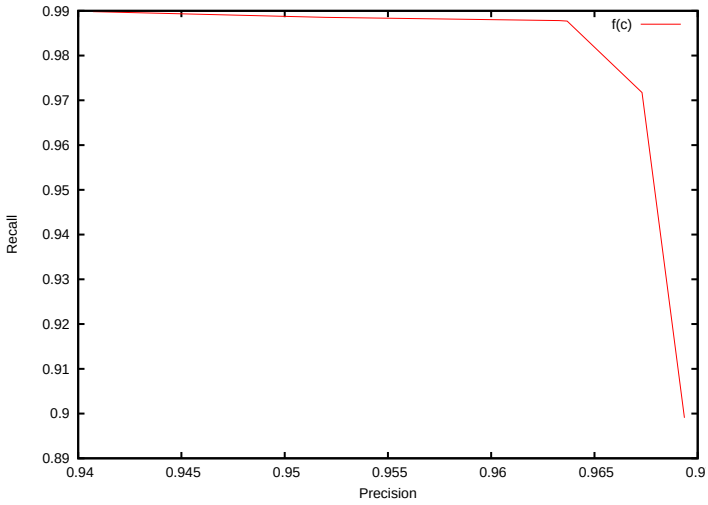

Figure 4: A graph showing how performance varies with c, the cut-off variable defining minimum score to be returned by a query, when only textual annotation is used and using rule set A. Note that the graph is not a funciton y = f(x) but varies with the hidden variable c between 0.00 (far left) and 0.30 (far right).

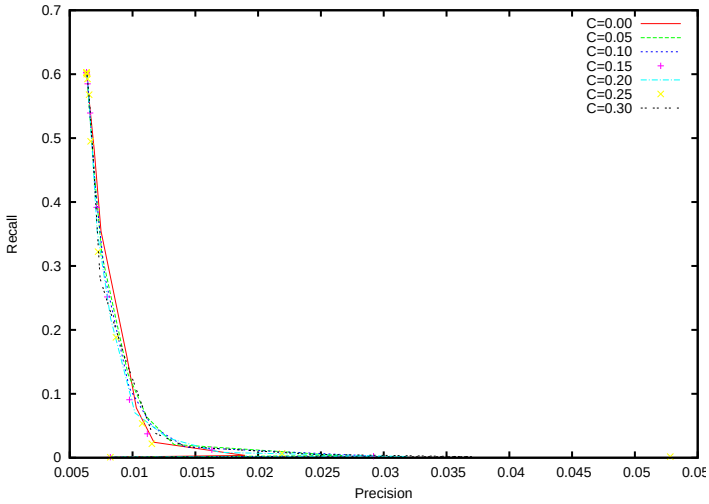

Figure 5: A graph showing how performance varies with θ, the variable determining how strict the face recognition is, and different values of c using rule set A and only facial recognition is used. θ varies between 1.00 (far left) and 0.00 (far right).

5.4 Classification rule set B

Classification rule set B was made to evaluate the performance of the face recognition on a test set where all correct results could be found using face

	Textual annotation			Best combined		
	Precision	Recall	F	Precision	Recall	F
AVG	0.9407	0.9899	0.9606	0.9637	0.9877	0.9721
VAR	0.0099	0.0018	0.0052	0.0077	0.0021	0.0044
STD	0.0994	0.0429	0.0720	0.0880	0.0457	0.0667

Table 7: Summary of results from the original data and the best results from the combined method considering the F-measure using classification rule set A.

recognition. A summary of the results from the textual annotation as well as the best combined method considering the F-measure can be found in table 8 and graphs of total performance can be found in figures 16-20 and a performance summary at $\alpha = 0.50$ can be found in figure 6. Worth noting is that the highest F-measures are found using the very same numbers as using classification rule set A. This is somewhat expected as the application performs in exactly the same way on the two data sets, the only thing that's changed is that a number of images has been considered incorrect results of some queries. As can be seen the average precision and F-measure is lower than using rule set A, but at the same time the standard deviation is higher, again making it hard to prove an actual improvement even if nearly perfect results could be accomplished.

Details of how the application performs using face recognition only and using textual annotation only can be seen in figures 7 and 8. As with rule set A the textual annotation performs well, although slightly worse as some of the images are now considered irrelevant due to not containing faces. The facial recognition is still performing as poorly though, resulting in overall lower numbers (as seen in table 8).

	Textual annotation			Best combined		
	Precision	Recall	F	Precision	Recall	F
AVG	0.9209	0.9900	0.9491	0.9431	0.9878	0.9606
VAR	0.0127	0.0019	0.0062	0.0104	0.0021	0.0053
STD	0.1126	0.0432	0.0790	0.1021	0.0460	0.0731

Table 8: Summary of results from the original data and the best results from the combined method considering the F-measure using rule set B.

Using this algorithm the best results I was able to get is still well within the standard deviation of the textual annotation data. This indicates that this facial recognition algorithm is probably not useful in a commercial application.

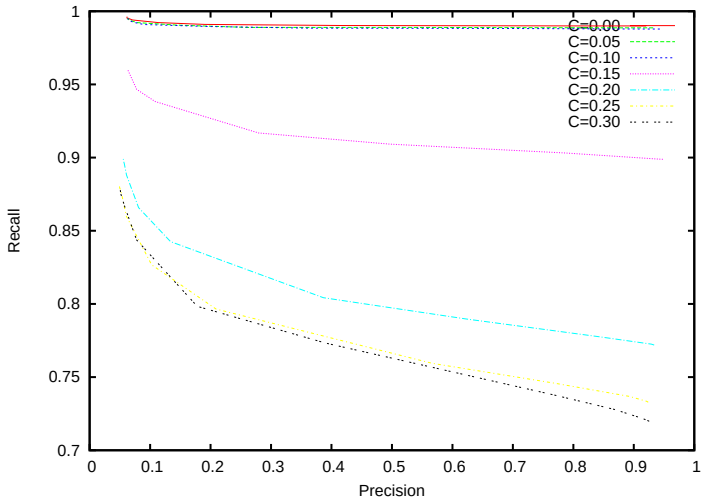

Figure 6: A graph highlighting performance when θ, the variable determining how strict the face recognition is, varies at $\alpha = 0.50$ using rule set B. Note that the graph is not a function y = f(x) but varies with the hidden variable θ between 1.00 (far left) and 0.00 (far right).

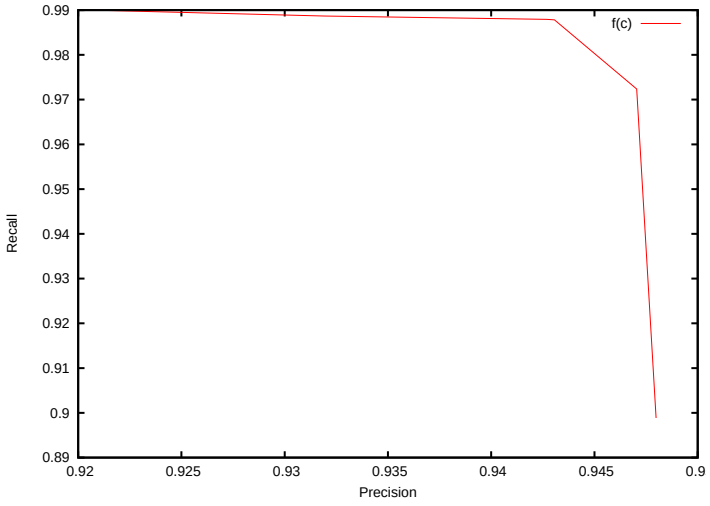

Figure 7: A graph showing how performance varies with c, the cut-off variable defining minimum score to be returned by a query, when only textual annotation is used and using rule set B. Note that the graph is not a funciton y = f(x) but varies with the hidden variable c between 0.00 (far left) and 0.30 (far right).

5.5 Detection only

The most reliable part of the program is the face detection algorithm which performs fairly well under the circumstances. The detection only results can

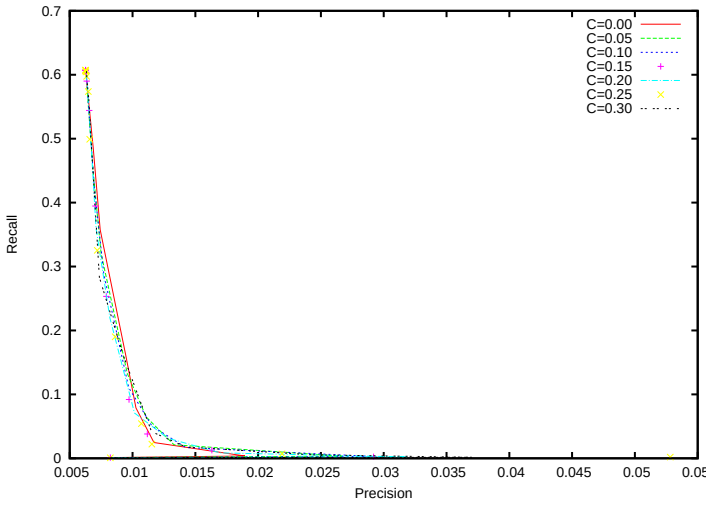

Figure 8: A graph showing how performance varies with θ, the variable determining how strict the face recognition is, and different values of c using rule set B and only facial recognition is used. θ varies between 1.00 (far left) and 0.00 (far right).

be found in figures 9 and 15 for rule set A and figures 10 and 20 for rule set B. An interesting thing to note is that the figures are almost perfectly identical, only a bit lower in precision for test set B. Worth noting is also that as alpha passes 0.5 precision drops down to about 0.63 for c values over 0.10 which is high compared to a drop under 0.10 when using face recognition.

My personal opinion is that this is the most reliable of the combined algorithms. Even though it doesn't show in the precision/recall calculations except when high cut-off values are used, face detection also sorts results so that facial images gets higher priority which may be considered useful in some features.

This is of course an evaluation of the application designed in this project. If the detected face could be normalized to fit the eigenfaces criteria and the face detection and recognition where allowed to work on all images, not just those already found using textual annotation another design might prove superior.

5.6 Results summary

As is apparent from tables 7 and 8 no results from the test sets used in this project could be statistically proven to be an improvement over the original textually annotated approach. Still, the application design remains valid

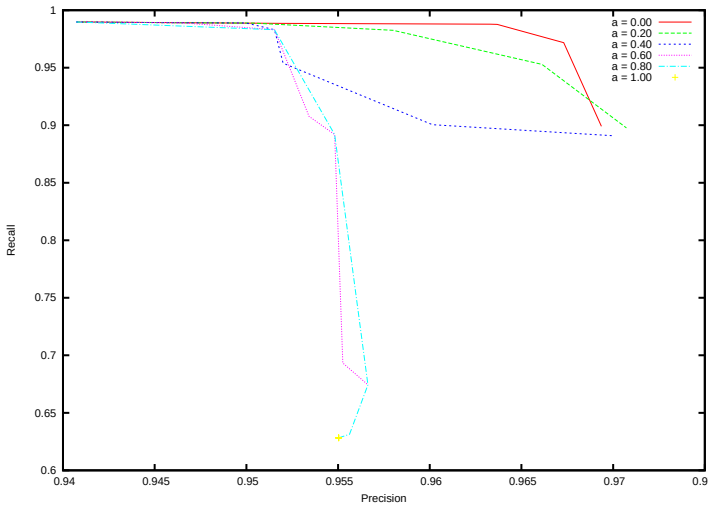

Figure 9: A graph showing how performance varies with c, the cut-off variable defining minimum score to be returned by a query, at different values of α when facial detection only is used and using rule set A. c varies between 0.00 (far left) and goes on to 0.30. Note that the graph is not a function y = f(x) but varies with the hidden variable c. Also not that when $\alpha = 0.80$ and $c > 0.20$ the system converges into face recognition only ($\alpha = 1.00$).

and can easily be altered to fit any face detection of recognition software, working on any data set. As can be seen in figures 11-14 and 16-19 the most important thing in an application like this is to get high precision results from the face recognition. In this implementation this could not be done as the eigenfaces algorithm requires faces that are oriented as the original training set to identify an individual. In the test sets used in this project the faces were in most cases not at all oriented in this way, and neither was the face detection quaranteed to supply the face recognition with the exact part of an image that would be required to fulfill these requirements.

The numbers presented in section 3.5 agree with the assumption that the design in itself is very scaleable but that the indexing becomes very time consuming with this method. As the textual annotation was done outside this project and the images used were indexed from a local computer no direct comparison can be made to show exactly how much more time consuming this approach is but considering that the graphics on a webpage often is several times larger than the source code this must be considered if further studies in this area is to be conducted.

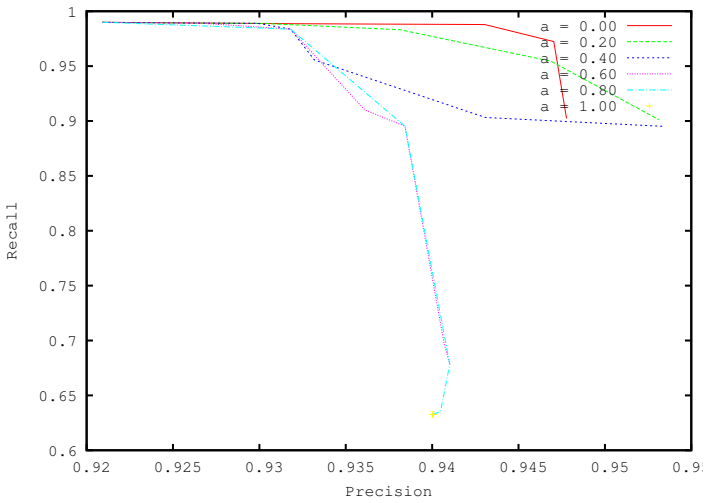

Figure 10: A graph showing how performance varies with c, the cut-off variable defining minimum score to be returned by a query, at different values of α when facial detection only is used and using rule set B. c varies between 0.00 (far left) and goes on to 0.30. Note that the graph is not a function y = f(x) but varies with the hidden variable c. Also not that when $\alpha = 0.80$ and $c > 0.20$ the system converges into face recognition only ($\alpha = 1.00$)

5.6.1 Weaknesses in the implementation

As can be seen from the results this approach to increase search relevance fails when it comes to reliability in the face recognition. The eigenfaces algorithm does not compensate for changes in lighting, viewing angle or facial expression [15].

Manual rearrangement of the detected faces could very well be attempted if the position of eyes, nose, mouth and/or other characteristics could be identified but as the face detection only defines an area as a possible face with no further details such an approach would require another kind of face detection.

The Yale face database is only 10 persons. Of these there are only one woman, noone is black, noone has a big beard or mustasche, noone wears glasses, noone is old and noone is a child. These factors make this training set an unrepresentative subset of the human population which will make it harder for the algorithm to successfully classify people who fall outside the norm of the Yale database images[4].

[4]Knowing this it is still very hard to aquire a larger, free, high quality, representative set of images.

The biggest flaw in the application design is that the face detection and recognition only works on the images already found by textual annotation, not on all images found when a website was indexed. This flaw makes the results from rule set A unrepresentative and it also affects rule set B in a similar way, although the results from rule set B can still be used to evalutate face recognition performance.

5.6.2 Possible improvements

While the face detection is the most advanced part of this project, it is also the thing that should first be discarded for another. One of the main reasons that the face recognition performs badly is that the faces are not corrected for rotation and aligned to the eigenfaces. The reason for this is that such a correction demands at least eye coordinates and preferably mouth and nose coordinates for a successfull correction.

An even greater improvement would be to build a system that would perform face detection and recognition in the same step and avoid potential errors where the detection and the recognition do not perform in the same way.

A problem with using very complex algorithms are that they are very time consuming. If a web crawler is to index images as well as text it already has to analyze a lot more data time really becomes an issue, as an index not only has to be high quality, it also has to be up to date. As eigenfaces can be calculated extremely fast when once the algorithm has been initialized it is a valid choice in an application that should be as scalable as a search engine.

5.6.3 Comparison to commercial products

Face recognition software has improved incredibly since the eigenfaces algorithm. As presented during the Face Recognition Vendor Test of 2006 [11] the false recognition rate (FRR) at a constant false acceptance rate (FAR) of 0.001 (1 per thousand) has gone from 0.79 in 1993 using semi automatic eigenfaces recognition (eye coordinates were manually found) to 0.01 using two different algorithms. Neven Visions NV1-norm algorithm using very high-resolution still images and Viisages V-3D-n algorithm using 3D images.

These performances cannot be directly compared to that of my program as their images could be normalized before the eigenfaces algorithm was used thanks to eye coordinates, and the best performing algorithms performed on 3D or very high-resolution images compared to low-res images of the internet. Still it is clear that face recognition has improved a lot since Turk and Pentlands Eigenfaces algorithm was first published.

A face recognition system with a FAR of 0.001 and FRR of 0.01 would likely give a combined recognition system with a precision and recall both over 0.99 using classification rule set B and likely almost as well using rule set A.

References

[1] Opencv library wiki - face detection.
http://opencvlibrary.sourceforge.net/FaceDetection.

[2] P. Deutsch A. Emtage. archie - an electronic directory service for the internet. In *Winter Usenix Conference Proceedings*, pages 93–110, 1992.

[3] Polar Rose AB. Polar rose.
http://www.polarrose.com, 2008.

[4] D.J. Kriegman A.S. Georghiades, P.N. Belhumeur. From few to many: Illumination cone models for face recognition under variable lighting and pose. *IEEE Trans. Pattern Anal. Mach. Intelligence*, 6(23):643–660, 2001.

[5] D.H. Salesin C.E. Jacobs, A. Finkelstein. Fast multiresolution image querying. Technical report, Department of Computer Science and Engineering, University of Washington, 1995.

[6] comScore Inc. Worldwide search top 10.
http://www.comscore.com/press/release.asp?press=2018, December 2007.

[7] Intel Corporation. Open source computer vision library.
http://www.intel.com/technology/computing/opencv/.

[8] J.C. Lin C.C. Han L.F. Chen, H.Y.M. Liao. Why recognition in a statistics-based face recognition system should be based on the pure face portion: a probabilistic decision-based proof. *Pattern Recognition*, 34(5):1393–1403, 2001.

[9] A. Pentland M. Turk. Eigenfaces for recognition. *Journal of Cognitive Neurosicence*, 3(1):71–86, 1991.

[10] A. Pentland M. Turk. Face recognition using eigenfaces. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 586–591, June 1991.

[11] Alice J. OToole Patrick J. Flynn Kevin W. Bowyer Cathy L. Schott Matthew Sharpe P. Jonathon Phillips, W. Todd Scruggs. Frvt 2006 and ice 2006 large-scale results. Technical report, National Institute of Standards and Technology, March 2007.

[12] A. Hartmann R. Lienhart. Classifying images on the web automatically. *Journal of Electronic Imaging*, 11(4), October 2002.

[13] J. Shlens. *A tutorial on Principal Component Analysis.* University of California, December 2005.

[14] L.I. Smith. *A tutorial on Principal Component Analysis.* University of Otago, February 2002.

[15] M. Turk. A random walk through eigenspace. *IEICE Transactions on Information and Systems*, E84-D(12):1586–1595, December 2001.

6 Appendix A: Graphs

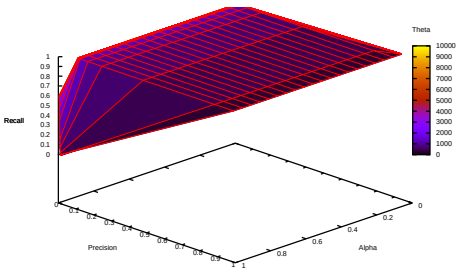

Figure 11: Results from test set A at c = 0.00.

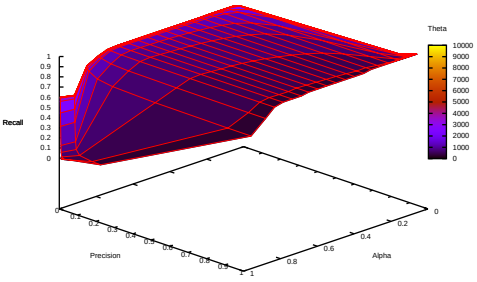

Figure 12: Results from test set A at c = 0.05.

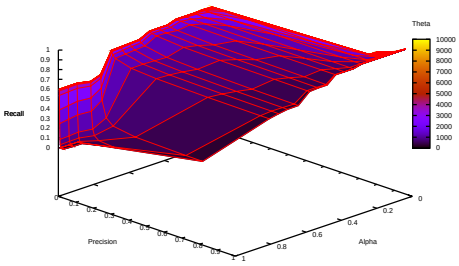

Figure 13: Results from test set A at c = 0.15.

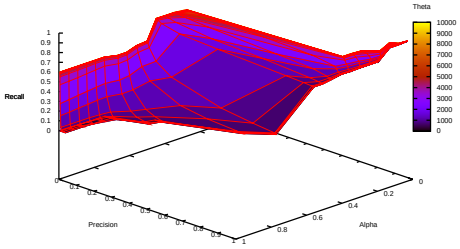

Figure 14: Results from test set A at c = 0.30.

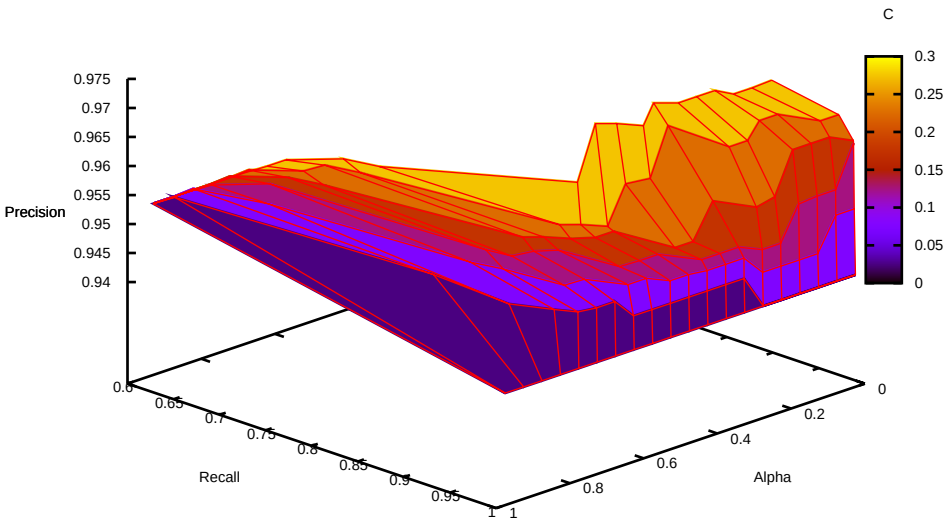

Figure 15: Results from test set A using face detection only.

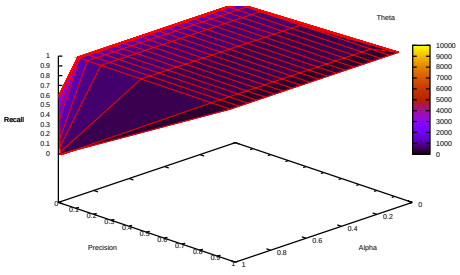

Figure 16: Results from test set B at c = 0.00.

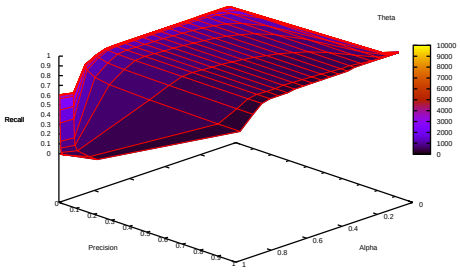

Figure 17: Results from test set B at c = 0.05.

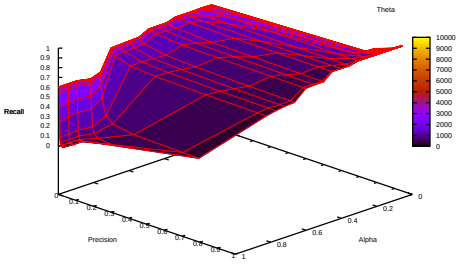

Figure 18: Results from test set B at c = 0.15.

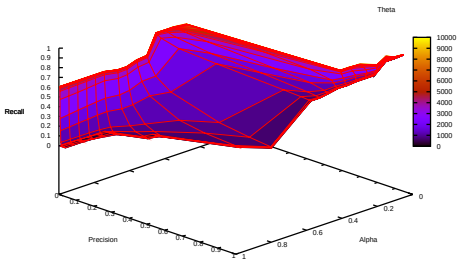

Figure 19: Results from test set B at c = 0.30.

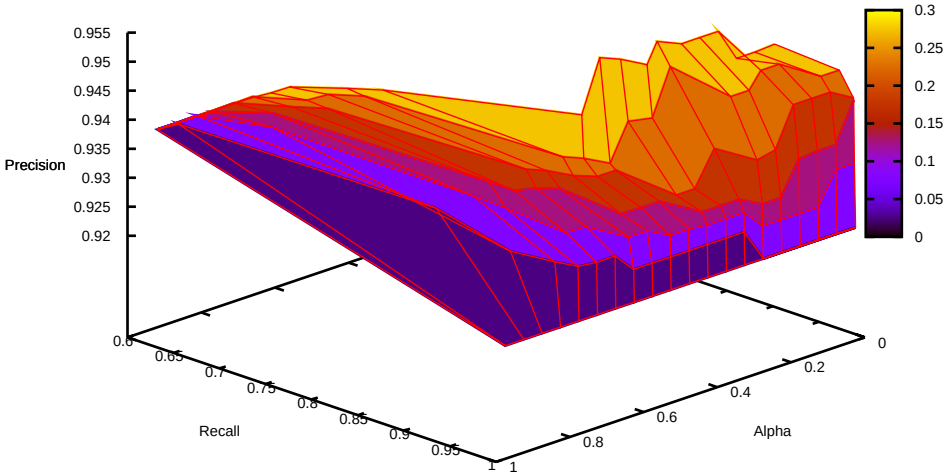

Figure 20: Results from test set B using face detection only.