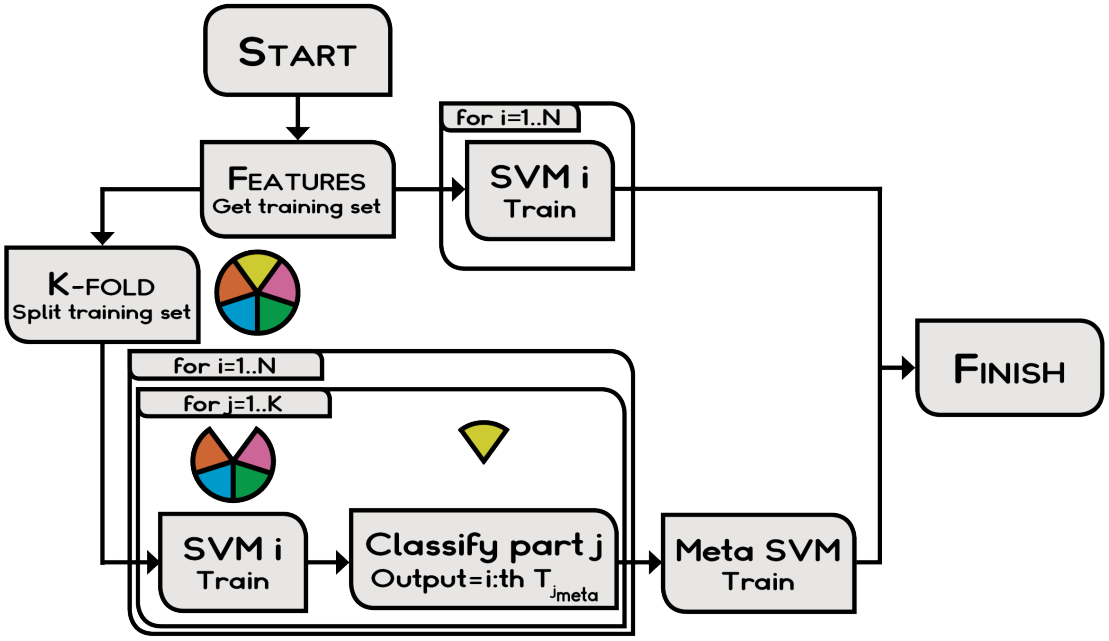
CHALMERS
UNIVERSITY OF TECHNOLOGY



Iterative interactive concept training on visual content

Master's thesis in Computer Science: Algorithms, Languages & Logic
Complex Adaptive Systems

GABRIEL ANDERSSON, MATS UDDGÅRD

Department of Signals and Systems
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2017

Master's thesis 2017

Iterative interactive concept training on visual content

GABRIEL ANDERSSON
MATS UDDGÅRD

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Signals and Systems
Chalmers University of Technology
Gothenburg, Sweden 2017

Iterative interactive concept training on visual content

Supervisor: Josef Eklann, Safer Society Group
Examiner: Fredrik Kahl, Department of Signals and Systems

Master's thesis EX058-2017
Department of Signals and Systems
Chalmers University of Technology
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Cover: A flowchart visualizing the process of training an SVM ensemble that is two layers deep. The process is described more thoroughly in Section 4.4.1.

Typeset in LaTeX
Printed by [Name of printing company]
Gothenburg, Sweden 2017

Iterative interactive concept training on visual content
GABRIEL ANDERSSON, MATS UDDGÅRD
Department of Signals and Systems
Chalmers University of Technology

Abstract

This thesis presents a novel method to quickly sift through the visual content (image material) of a database in order to retrieve as much relevant material as possible. The proposed model uses a combination of classification systems, image retrieval and relevance feedback. Five different feature descriptors, known to be useful within image retrieval, are extracted to later be inserted into a classification system. The material is presented to, and corrected by, a user and can therefore be used as training data in future iterations. The training data is inserted to a supervised learning classifier in order to search through the database. The most relevant material is passed through the feedback loop allowing the model to learn concepts in a fast manner.
The five feature descriptors that are commonly used within the field are the following: *histograms of oriented gradients*, *global color histograms*, *Haar wavelet transformations*, edge detections using a *Sobel filter* and the final activations of a *VGG-16* neural network.
In the classification system a classifier called *Deep SVM* (*Deep Support vector machine*) is used. In the proposed model it consists of 6 SVMs in order to create an ensemble, where one is used for each kind of feature descriptor and the last SVM is used to combine the result of the first order classifiers. Material in the search space is passed through the system and the most relevant material is presented to an expert user.
Evaluations and measurements were performed on the model in two settings. Firstly as a parameter benchmark in order to find the most appropriate setting for the intended use of retrieving all the relevant visual material in a crime investigation case. Secondly as an image retrieval comparison with other studies by using a small training set as query material. The parameter benchmark shows that the model is capable of retrieving the majority of relevant material within a small number of iterations. The study comparision shows that even though the model is designed to have sets of images as query data, the size of the sets does not have to be greater than 10 in order to outperform the related approaches.
The contributions of the thesis consist of the following: Using a Deep SVM in combination with relevance feedback to perform an image retrieval results with great performance and a complete retrieval within a low number of relevance feedback iterations. Content-based image retrieval has previously been performed with one image as query material while this thesis presents a method of using a set of images for the task in order to achieve a higher abstraction level.
Keywords: *Machine learning, Ensemble learning, Image analysis, Content-based image retrieval, Relevance feedback, Semantic gap, Feature extraction, Neural network, Support vector machine, Deep SVM*

Acknowledgments

Firstly, we want to reach out our uttermost thanks to our host company *Safer Society Group* for contributing with an office location where we could work and write a report for the entirety of the thesis.
Secondly a thanks to our supervisor, *Josef Eklann*. Thank you for providing with knowledge and bringing support.
Finally, a thanks directed to our examinator, *Fredrik Kahl*. This would not had been possible without your help.

Gabriel Andersson
Mats Uddgård

Gothenburg, June 19, 2017

Contents

List of Figures

List of Tables

1
Introduction

Digital video and image files are normally important evidence in criminal investigations, and the amounts of images and videos that constitute the evidence are larger now than ever [1]. The computer forensics community has ever increasing problems with this continued growth of information and in investigations, the amount of data to be examined is often problematic [2, 3, 4]. In order to effectively help the investigators in their tasks of organizing and prioritizing evidence, methods to scrutinize the data in an efficacious manner is vital. In investigations pertaining relevant digital information quickly and effectively material is of importance, as evidence can be of large quantities. Methods focused on grouping material by some collective attributes are often found to be efficient. There are several of these attributes that can be correlated to the information obtainable in visual content, such as in images and video. Which images that are relevant might differ from case to case, it would be a good praxis if the investigators could define their own respective grouping setting for each separate case. By letting the investigator define some form of concept by directly specify image examples as relevant and non-relevant an algorithm can be trained to recognize a concept queried by an investigator.

1.1 Problem definition

Within digital forensics, the amount of investigation material has grown exponentially while the workforce still grows at a linear pace. In order to handle larger amounts of material, algorithms need to be designed to handle large amounts of data in favor of the user. The purpose of the different cases that the forensic investigators handle varies in-between investigations. Because search engines and methods of retrieving material have a static behavior, the handlers adapt their behavior to the currently used search engines. The workflow and usage areas of an algorithm should be specified by a handler and the behavior of the algorithm should adapt to the need of the user and should do so in a generic manner.

As described above the problems that this thesis approaches are the following:

- The material of the investigations are not handled fast enough and the time needed to handle it needs to be reduced.
- There are no image retrievals that fit the current need and to have an algorithm that adapts its behavior to the user is crucial.
- Using search methods that are good at certain things does not cut it. An image retrieval method that works for the general case is to prefer.

The standard use case of such a system that solves these issues can be described as:

1. The algorithm presents a set of images based on previous knowledge of the search preferences to the user.

2. The user specifies which of these images that are relevant to the current case and which are not relevant.
3. The algorithm adapts the search criteria based on new knowledge.
4. Repeat from item 1.

The entire procedure can be continued and each iteration refines the search criteria and after a while, the database of material should be exhausted of relevant material.

1.2 Goals

The aim of this project is thus to:

- create an algorithm that helps to identify relevant images in a database which should be subjected to a user defined concept, a dynamic general concept search engine.
- exhaust the database of relevant images faster than an independent user or random search.
- put more emphasis on trying to minimize the number of false negatives than the false positives, in the search as to lower the risk of neglecting images that would be of importance to the operator. False negatives being images that are classified as non-relevant while being relevant, and false positives being non-relevant images but predicted as relevant.

1.3 Delimitations

In the scope of this thesis choices were made to be able to propose a functioning model that solves the problem specified in Section 1.1. The aim is to create a dynamic general concept search engine with following delimitations:

- Some parameters of the model need to be chosen empirically since all settings are missing support from previous papers. However there are some choices that are made more elaborately, e.g. parameter benchmarks are performed in order to find the optimal setting.
- The classification method chosen will be binary since this will be enough in the scope of this project. The classifier will not be tested towards other methods of classifying.
- Only five different feature descriptors will be used, as the proof of concept of a general learner is central and not how the addition or omission of certain parts changes this function. By using five different feature descriptors makes it possible to get the distinction and variation sought for in a general sense.
- The data used from relevance feedback will be limited to only consist of which images that are truly relevant and which that are not. Even if it is possible to use more data this will be the case.

- The selection process of images in the search of relevant ones is not covered by the scope of the thesis and is therefore performed as a random search.

1.4 Contributions

A classification method is used as a generic image retrieval system to help quickly identify and learn user defined concepts which are previously unknown to the system. The implementation of a Deep SVM, described in Section 4.4.1, with relevance feedback, described in Section 2.2 to perform image retrieval with a low false negative rate within a low number of relevance feedback iterations. An attempt to use both weak and strong learners in the same ensemble to enhance performance in terms of a more generic classification. In this report a new content-based image retrieval (CBIR) method is tested that uses more than one image as the query to achieve a higher abstraction level and boost performance.

1.5 Organization of thesis

Chapter 2 **Theory** presents the basic and central concepts in the scope of the thesis such as CBIR and relevance feedback. An introduction to the datasets used and information of why different formats and color ranges are important in this thesis.

Chapter 3 **Image analysis theory** introduces the relevant parts of image analysis. What an image feature is and the different feature descriptors that are implemented.

Chapter 4 **Machine learning theory** describes supervised learning, what classification is and which methods that are used.

Chapter 5 **Method** explains how the proposed model is constructed and how evaluations were performed in order to test the model. The evaluations split between parameter benchmarks and study comparisons with other CBIR models.

Chapter 6 **Results** presents the obtained results of evaluations described in Chapter 5.

Chapter 7 **Conclusion** discusses the results presented in Chapter 6. From these discussions possible extensions on the model are presented as well as how some functionality of the model can be extracted for external usage.

2
Theory

In this chapter the theory of the key concepts behind this thesis are presented. Here content-based image retrieval is explained as well as the difficulty of closing the semantic gap. The idea of relevance feedback is introduced as it is a key module in the thesis. Lastly the different datasets used in evaluations are presented accompanied with a brief introduction to how images are stored digitally. The chapters *Image analysis theory* and *Machine learning theory* extends theory in their respective branches.

2.1 Content-based image retrieval

Content-based image retrieval (CBIR) is a term referring to techniques used in computer vision, where the goal is to find images with similarities in large databases. Given a query that is presented to the system, the output would be a set of images extracted from the total set which have the highest resemblance to the query. Content-based refers to the information stored in the image itself and not, as is the case in tag-based image retrieval (TBIR), the metadata of the image. CBIR is therefore a viable approach if either the metadata is non-existent or the classification is of some other variety than what the metadata can give. A situation when CBIR might be good to use is when the visual content of the semantic nature or there are reoccurring objects in several different images. In modern CBIR systems there are four reoccurring major parts [5]:

- Feature extraction, where the raw features are recovered.
- Feature reduction, the recovered features are used to reduce feature dimensionality and storage space usage.
- Ranking, systemize the images so that the system can categorize the images in the dataset depending on resemblance to the query.
- Finally relevance feedback, the final feedback given by an expert user if needed correcting the algorithms predictions.

A large problem in image retrieval is that the query image might hold information easily perceived by a user but hard to concretize in pixel data and features, this problem is called the semantic gap. It can be said to be the difference that arises when two different linguistic representations try to describe the same object. This is a relevant issue whenever the perspective of a human is tried to be represented by a computer. This, commonly known as the semantic gap, is bridged by functions that interprets the pixeldata of the image for the computer so it can get a concept of the object and recognize what it perceives [6].

2.2 Relevance feedback

A way to avoid the problems that arise when dealing with the semantic gap is to use relevance feedback. Relevance feedback can be said to be the direct interaction between a user and a machine in learning. The user reviews and corrects the predictions that the machine has made. The machine can in return use this information to re-evaluate the predictions that were made. The process is in general that a user is presented with a number of images by a machine learning algorithm. Images that the algorithm has tried to label with the help, or occlusion, of some pretraining. These images are re-evaluated by the user and corrected by her if the corresponding label happens to be falsely assigned and acknowledged otherwise. With these adjustments to the data the to the data the retrieval process is refined in an attempt to make future classifications better. These two parts are then iteratively carried out as the algorithm keeps searching through the dataset for the required images [7]. There are different kinds of relevance feedback methods, the three most common are explicit, implicit and blind (pseudo) feedback [8]. Using explicit feedback means that a user, knowingly of that her actions will affect how future material will be presented, indicates the relevance of the material presented to it. The first of the other two feedback variations is implicit which either means that the users behavior is observed or that the user is unknowing that the feedback are used as relevance feedback for the system. The other is pseudo relevance feedback which is a form of automated feedback that uses the first query as relevant results. In view of the situation of investigations having a need for the user to review all images in any case, explicit feedback is the best viable option. The explicit feedback is used to create a continuous data confirmation and thus creating more reliable data in each iteration.

2.3 Image formats

There are several file formats available and different ways to store images, such as uncompressed and compressed raster formats as well as vector formats. When an image is stored with a raster format it is represented by a grid of pixels with a depth depending on the information of the image. The most common way to store image information is to use a 24-bit RGB pixel, where RGB is an abbreviation for red green and blue. Each 24-bit pixel has three equally sized channels of 8 bits, which makes each color channel range between 0 and 255. This results in approximately 16.8 million different combinations for colors, where a human can perceive about 10 million [9]. A computer screen usually operates at the described color setting. The size of the file simply depends on how many of these pixels that are stored, i.e. the size directly depends the dimensions (width and height) of the image. As the sizes may vary, some images can become expensive to process. Because of this it can be prudent to downsize the image to a smaller number of pixels and thus avoiding unnecessarily large amounts of data. Downsizing can be especially useful since high pixel information is not always equivalent to good performance [10]. There are

several different color spaces, of which some are used in image analysis. The most commonly known being RGB which is an additive color range where each channel is a color incorporated with light intensity. It is based on generated light and the addition of these to create the color spectra. The use of additive color combinations of red, green and blue are useful in technology as television sets and computer screens, the different channels of RGB are visualized in Figure 2.1. As humans tend to react more on the hue and saturation of an image than color they are considered rather inept in image analysis [11, 12].

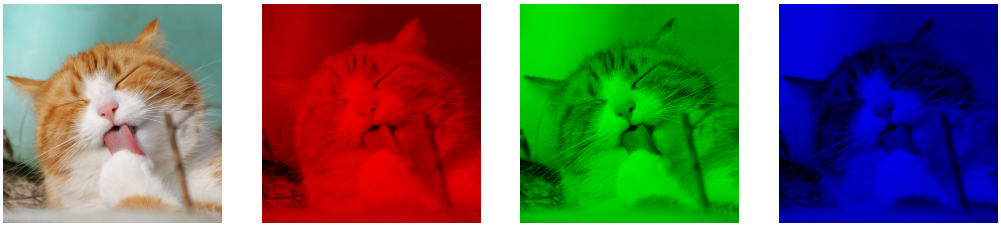


Figure 2.1: The different channels of the color space RGB. The left image is the original image and the other three show each independent channel. From the left: Red, green and blue channels. The brighter the pixel the higher color value since RGB is an additive color space. Best viewed in color.

2.3.1 Hue, saturation, value

Hue, saturation and value, abbreviated as HSV, is a color space in line with RGB. The difference is that HSV is a cylindrical representation of RGB, where RGB is mapped as a cube where the channels r, g and b can be interpreted as the often named x, y and z axes. The HSV can be mapped cylindrically where hue is the degree position of the cylinder, saturation is the radius and value is the height. HSV is considered to be a color space that is a closer representation how a human perceives the world and thus also often used in the field of CBIR and image analysis as a whole. The different color channels can be seen in Figure 2.2.

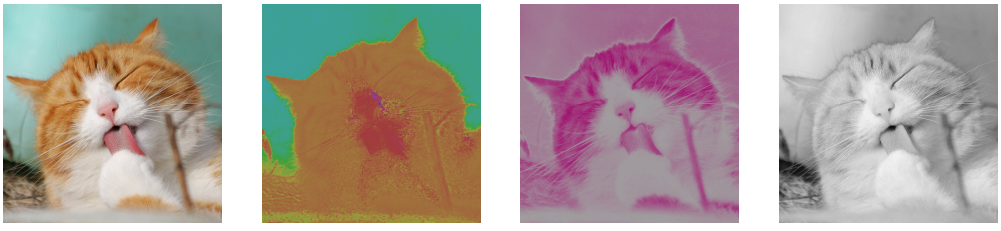


Figure 2.2: The different channels of the color space HSV. From the left: All color channels are active, only the hue channel, only the saturation channel and only the value channel. Best viewed in color.

2.3.2 YCbCr

This representation is composed to work towards human perception where the luminance component (Y) can be seen as analogous to the brightness or light component and the two chroma (Cb and Cr) filling out the color spectra [13]. The three channels can be seen in Figure 2.3. The color information is not always as vital for human perception as the brightness is. The human retina has three types of photoreceptor cells where two are commonly referred to: The rods, that are very sensitive to light and can be triggered by a single proton, and the cones, that are less sensitive to light but reacts differently to individual wavelengths of light. A human has ≈ 120 million rods and ≈ 6 million cones. The number of photoreceptors is somewhat of an indicator of which channel in YCbCr that affects human perception the most. The luminance is often used in edge detection since it conveys textures, illuminates the shapes of objects and portrays depth in images [14, 15].

Figure 2.3: The different channels of the color space YCbCr. From the left: full image, the luma channel (Y) and the two chroma channels (Cb) and (Cr) in that order. In the luma channel edges are abstracted from the color compositions. Best viewed in color.

2.4 Datasets

Datasets are used to evaluate and compare the performance of a proposed model with other studies. This is often done for CBIR systems as well as classification systems. Examples of evaluations that uses datasets are plain recognition, image retrieval and image classification. To be able to evaluate and compare with several recent papers [16, 17, 18, 19], one of the datasets used in the thesis is the dataset Corel-1000. This set comes with its limitations as it is relatively small in comparison with the huge datasets used to train deep neural networks. Neural networks such as GoogLeNet [20], AlexNet [21] and VGG-16 [22] are designed to compete in the ImageNet Large Scale Visual Recognition Competition (ILSVRC) [23], a yearly object detection contest where 1000 object categories are present. Since the goal of the thesis is to learn general concepts and not to detect objects another dataset was used: The dataset Places205, designed by MIT, described in Section 2.4.2.

2.4.1 Corel-1000

The Corel set is an image dataset often cited and used in validation of different CBIR systems [24]. The dataset is a low resolution set composed of 80 classes (concepts) with 10.800 images in total. Due to the size of the dataset a subset, called the Corel-1000 dataset, is used to compare the proposed method with related CBIR approaches [16]. The Corel-1000 dataset is a subset of the Corel dataset which contains 1000 images, composed by 10 classes with 100 images in each class. The images in this dataset are of the sizes 64×96 and 96×64 pixels depending on their orientation. The 10 classes are referred to as Africans, Beaches, Buildings, Buses, Dinosaurs, Elephants, Flowers, Horses, Mountains and Food.

2.4.2 Places205

Places205 is a dataset produced by MIT and collaborators in the search for ever better human-reaching performance with machine-learning.

The MIT places205 is chosen to be part of the evaluation of the algorithm presented in this thesis since it has a large variety of images and classes and is a well-known dataset [25]. It is a repository of millions pictures, labeled with scene semantic categories and attributes. The dataset consists of 205 sceneries with an average of 12000 images in each class. The different sceneries of the dataset places205 can be seen in Table A.1 in the appendices. The table lists all the names of the categories.

3
Image analysis theory

This chapter introduces and explains how images can be represented in more ways than the rasterized formats that simply consists of the pixel data. The chapter introduces and dives into some of the different content representations that can be used as well as what features and feature descriptors are. The theory behind the feature descriptors within the scope of the thesis is explained.

3.1 Visual features

Features in image processing are embedded information in either the picture itself or the meta-data concerning the picture. These features are extracted and used to solve various problems in computer vision, machine learning and pattern recognition. There are myriads of methods to extract features from an image and which one that suits the problem at hand varies. Commonly used features for CBIR are those that describe color, texture and shape. When the features have been extracted the resulting data are called feature vectors. The length of the feature vector varies, depending on the image being processed, the method used for description and the detail in which the features are extracted.

3.2 Feature detectors and descriptors

Feature detectors are usually built towards detecting either global features or local features. Global feature detectors are often color or texture oriented and make descriptions based on all pixel data in the image. These are good at identifying similar images but can have a hard time to distinguish between foreground and background of an image as they work with the whole image and thus usually fail to find local nuances and differences. They are often built to output a feature vector with the focus on a number of properties of the image involving all the pixels. In contrast, the local feature detectors focus on key points in the image and try to describe these and sometimes the area of pixels around these. Local feature detectors are often used to locate and recognize identical objects which may be skewed and transformed in some way in different images. The result is often several vectors representing the points of interest in the image. These attributes come with a cost, local feature detectors are often expensive in terms of computational power and data storage [26].

A feature descriptor is used when the interest points have been identified or detected as previously stated. The descriptor creates a set of vectors based on this information which can be used in the proposed retrieval model. In image processing and image analysis, one uses feature descriptors to facilitate the transfer from an abundance of features that are derived from images to a subset or transformation of these features

to create a more manageable dataset. Depending on the scope of the project the raw features one would get from the feature descriptors can become too large to handle and work with. To minimize this problem, the feature descriptors can be said to carry out dimensionality reduction on the data. Without any form of reduction of the actual information size, there would be a large requirement on memory and computational power.

Feature descriptors can, based on their performance, also be divided into either weak learners or strong learners. A weak learner is a feature descriptor that has an average precision higher than random chance towards a certain category. If the data were constructed of two classes with equally large categories a weak learner should have a precision just above 50%. A strong learner, in contrast, is a feature descriptor with a much higher precision which on its own can discern a majority of the images, this can e.g. be a deep convolutional neural network. Note that there are not a clear distinction here since the classification of a feature descriptor as a weak or strong learner is not only which method that is used but which dataset it should be applied on.

3.2.1 Histogram of oriented gradients

Histogram of oriented gradients (HOG) is a feature descriptor used for object detection in the fields of image processing and computer vision. The idea is that a local object shape and appearance can be described by the distribution of intensity gradients, or edge directions. The HOG computes the first order gradient, as these capture contour and some texture information. This is all done on the locally dominant color channel which usually is the gray channel. Then a pooling method is used where the image is divided into a number of cells in where the gradient orientation is acquired. The orientations are distributed in a fixed number of bins or possible orientations. The different magnitudes determine the result of the gradient histogram. When these has been evaluated the cells are grouped into blocks which creates a film over the "surface" of the cells of the image thus creating a new normalized gradient oriented image. The blocks are set to overlap thus each cell are accounted for several different calculations over different blocks. The normalized blocks are referred as HOG descriptors. The descriptor is essentially the concatenation of these local histograms. Examples of how the gradient orientations are put into bins can be seen in Figure 3.1. The HOG is well suited for human detection according to [27].

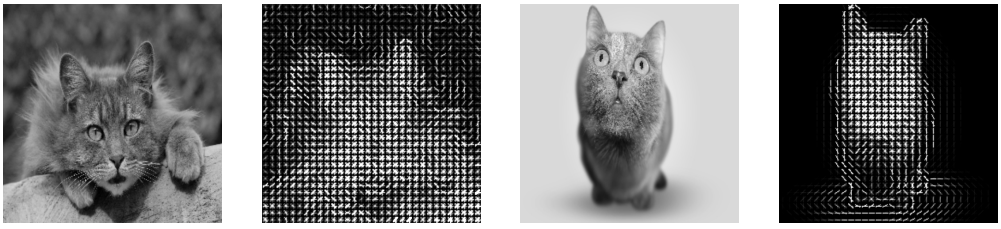


Figure 3.1: Examples of how the directions of gradients are binned in two images. The images are split up into cells with directional histograms. The number of histograms is substantially larger than when used in the thesis.

3.2.2 Global color histogram

A color histogram is a representation of the color distribution in an image. The color histogram method counts the number of pixels with similar attributes and stores them in a number of bins. There are no specific sizes for the bins, though there is a max size based on the current color range used. The size used is dependent on the performance of using fewer bins in contrast to the computational cost of using more bins. In image analysis, it is common to use HSV for this histogram, which would result in three dimensions of bins, one for each color channel. In the Figure 3.2, two examples of putting colors into bins are visualized, one with high and one with low resolution. When calculating a global color histogram (GCH) one does not take smaller patches of the image and calculate the concentration of information locally, instead it is calculated over the image as a whole. This approach can be insensitive where the objects might be different but the color characteristics are not. Two completely different images can still contain the same GCH values due to that the images have similar color settings.

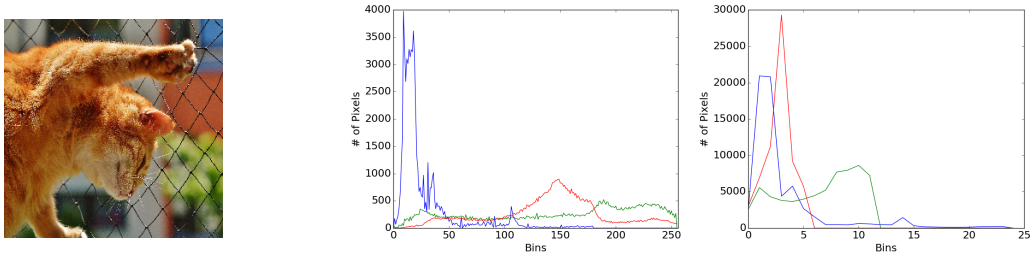


Figure 3.2: The HSV color channels of the image to the left are binned into two histograms (H=blue, S=green and V=red). The left uses one bin per level of the color channels and the right histogram is grouped into a substantially smaller number of bins.

3.2.3 Wavelet transform

In this thesis, a Haar Wavelet transform was implemented, the simplest of wavelets. Haar-like features are a type of digital image features which are used in object recognition [28]. The wavelet was implemented by first taking the differences and means of each pixel to its neighbors. This is done once per pixel in horizontal calculations and once in vertical calculations, the values are then divided into the different rectangles. The signal is decomposed into a subset of signals representing different elements, approximations of the image and intensities in the different directions and orientations. So for each time the wavelet transforms the image a couple of smaller ones are created with a dimensionality of 2. This makes it possible for a wavelet with an image of size $256 * 256$ to be minimized 8 times ($2^8 = 256$). In Figure 3.3 a representation of the wavelet transform for the three first levels is presented.

Figure 3.3: From left to right is shown representations of the different levels of wavelet transform, original image, one level applied, two levels applied and three levels applied respectively.

3.2.4 Convolutional neural network activations

The field of Artificial neural networks emerged with the McCulloch and Pitts neuron in 1943 [29]. The idea was to simulate the human brain, where there are about 10^{11} nerve cells, or neurons, that through a symphony of signals communicate information from and to other neurons. This is achieved by creating nodes for the neurons and edges interconnecting the different neurons to simulate the axons and dendrites. Dendrites work as a form of input to the neurons which have different intensities in their signals, in artificial neural networks modeled using weights. The neuron, or soma, then sums up the inputs into an output which then the axon can signal forward to other neurons and so forth. From the basis of this simple adaptation, the simulated neural network is created. In the simplest form, a neural network is composed of a single layer of neurons. These are presented with an input with weights, which are analogous to the intensity of the signals measured frequency in the animalistic brain. The neurons then process these values into a response or output.

3.2.4.1 Convolutional neural networks

In this thesis a subset of the field of artificial neural networks is used; convolutional neural networks (CNNs). A CNN is a type of a feedforward neural network that tries to simulate the animal visual cortex. A feedforward neural network is an artificial neural network wherein the connections between the units do not form any cycles or loops. A multilayer feedforward network is composed by, an input layer, an output layer and zero or more “hidden” layers. In this simple illustration the first layer, the input layer, receives the first number of inputs to be processed by the neurons. this is then sent to the next layer as inputs and the chain continues until the final layer outputs the final result for the whole neural network.

3.2.4.2 Network components

The architecture of a convolutional neural network is not static, which results in that the design may vary a lot between different networks. The most common building blocks are convolutional layers, pooling layers and fully-connected layers. They are

variations of multilayer perceptrons which are designed to use minimal amounts of preprocessing. In addition, an often used unit is the rectifier linear unit. The CNNs are constructed by using several different components, some of the most commonly used ones are the following.

The convolutional layer is the backbone of a CNN. They are composed of a set of learnable filters, also known as kernels, with their distinct size of receptive fields. They traverse the whole area of the image, convolved across the height and width of the input, all the while computing the dot product. The combination of the filter locations produce a 2D activation map for each filter and all 2D activation maps are stacked for each filter which creates the output matrix of the layer.
The pooling layer is essentially a non-linear down-sampling of the input image. This is done by some algorithm where max pooling is one of the most commonly used ones. It splits the input into several smaller non-overlapping rectangles and then outputs the max value from these regions. The idea is to periodically insert pooling layers in between the convolutional layers, in so doing reducing the number of parameters as the size decreases and thus lessen the amount of computation needed for the network. This also helps to reduce the risk of overfitting.
The rectifier linear unit, known as ReLU, which applies a non-saturating activation function to increase the nonlinear properties of the network. This is done without directly affecting the receptive fields of the convolutional layer. The general idea is to reduce the time it takes to train the network while still retain the generalizing nature of the network. It checks the values of the input layer and if the value is below 0 sets it to 0, and otherwise sets it to 1.
The fully-connected layer is normally the last couple of layers in a CNN. The layer is completely connected to the previous on meaning that the activations of all the neurons in previous layers connect to all neurons in the next layer. These layers constitute the high-level reasoning of the network.
The softmax activation function is normally used to produce the final output of the CNN where a loss function is set to determine how to penalize the network if prediction deviates from the actual labeling. A softmax function is designed to use a probabilistic interpretation of the activation values and then normalize them. which very is useful when the output is applied to a cross-entropy loss.

One of the hardest parts is to configure a CNN based on these layers to create a well versed and functioning network. In this thesis the VGG-16 [22] is implemented. How the VGG-16 is built by using all these different layers in their convolutional neural network is shown in Figure 3.4. Transfer learning is a method where one use pretrained CNN models and then just remove the last output layer, and extract the features directly from the fully-connected layers [30]. To be able to use the network for the purpose of a feature descriptor modifications were made. The final fully-connected layer as well as the softmax layer, the loss layer, are both omitted as can be seen in Figure 3.4.

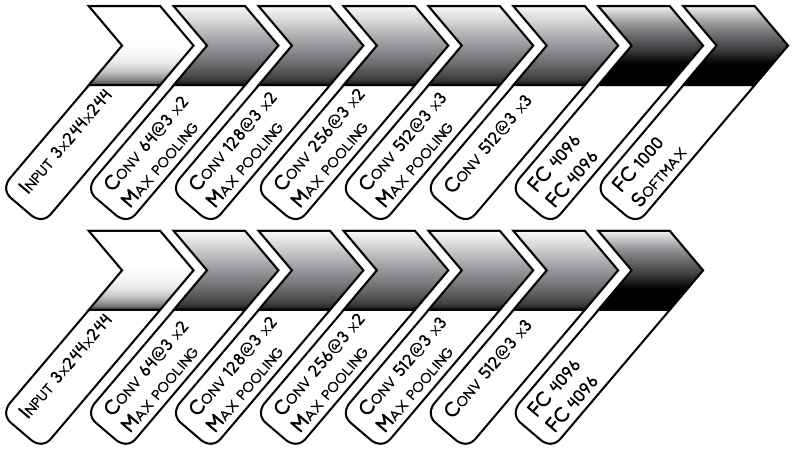


Figure 3.4: Top: A simplified visualization of the VGG-16 CNN. Bottom: The modification done in this thesis.

3.2.5 Edge detection histogram

The set of edge detectors is a group of different methods that aim to identify strong shifts in images, which could signify discontinuities in the image [31]. A common way is to use the channel representing the light of a color space to find the discontinuities, like the luminescence (Y) as mentioned in Section 2.3.2. The name edge detection is based on the relevant points that signify discontinuities which are called edges. There are two distinct methods commonly used in edge detection. These are the search based and the zero-crossing based edge detection. The first method, the search based, looks for the local directional maxima of the gradient magnitude, often using a first-order derivative expression to compute this. Examples of these are the Roberts, the Sobel and the Prewitt operator. All these edge detectors utilize convolutional masks in order to calculate the gradient. The result of Sobel edge detection is shown Figure 3.5. Zero-crossing based, the other method, uses second-order derivative expression to find where there is a jump of values, the zero-crossing of the image. An example of these is the Laplacian operator, which often is used with an approximate convolutional mask [32]. The identification of these edges can be used to find more relevant objects and shapes in the images, which can be used to build predictions on [33].

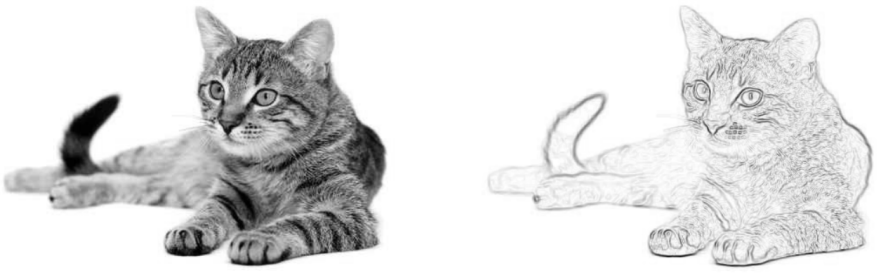


Figure 3.5: Edge detection, to the left a gray image of the original, to the right a image showing the edges found using Sobel edge detection

4
Machine learning theory

This chapter introduces key concepts within the machine learning subfield of out thesis. It will start with a brief overview of machine learning theory and how supervised learning works. The chapter will continue with the classification in general and the basic ideas of Support vector machines. It will then conclude with the key feature of ensemble learning and Deep support vector machines.

Machine learning is a field in computer science focused on methods where the computers learn without being explicitly programmed. Thus it can be said to be the study and construction of algorithms that can learn from data and make predictions based upon it. In order for a person to learn about something new, the person looks back on previously learned knowledge and machine learning algorithms do not differ from this pattern. A famous quote that describes machine learning by Tom M. Mitchell is *"A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E"* [34]. Machine learning is applied to several different subjects in a number of different fields. Even though there are several learning methods and utilities of machine learning this thesis will only handle the concepts of supervised learning and classification.

4.1 Supervised learning

Supervised learning is a training method for a computer program, where labeled data is explicitly used. Labeled data is a group of samples $\mathbf{x} \in \mathbf{X}$ composed of some form of information, distinguishing the samples $\mathbf{x}_i$ where $i = \{1, ..., t\}$ and labels $y_i \in \mathbf{Y}$, or targets, corresponding each to one sample. The labeled data is usually split into three different sets: A training set, a validation set, and a test set. The training set is used to train an algorithm towards a certain concept by generating a function based on the data. If presented some unknown data this function should be able to categorize the data into correct labels. So presented with a set of samples $\mathbf{x_j}$ where $j = \{1, ..., s\}$ it should predict the correct labels y_j. The validation set is then applied to get an idea of how it is performing and to see if further changes are needed to get an acceptable result. If one tries multiple approaches, this should determine which to use if the learning algorithm performs as anticipated. The last part of the labeled data, the test set, is then used to check what a possible expectation of the algorithm could be when presented unlabeled data.

4.2 Classification

Classification is a general problem in pattern recognition where some form of input value should result in an output value which is representing the label of the element.

The research behind classification is extensive and many different fields are working with classification. The choice of which classification method that should be used varies depending on the data. It is of note that no one classifier suits all cases and no one classifier outperforms every other in every other case as per the "no free lunch" theorem from Wolpert and Macready [35]. The most basic form of classification is binary classification where the data is separated into two categories, for instance as relevant and non-relevant. The input to these algorithms is often referred to as feature vectors, $\mathbf{x_i}$. The number of dimensions of these vectors might vary in-between different feature descriptors, which the classifier must be able to handle.

4.3 Support vector machines

A Support vector machine (SVM) is essentially a supervised learning model where data is analyzed for classification and regression analysis. It can be said to be a non-probabilistic binary classifier since new examples are assigned to either of two categories. The way an SVM works is that the training data used is mapped in a way that separates the different categories by an as large margin as possible. The margin are between a boundary and the data points are measured geometrically and therefore distance functions are designed in various ways to attack this kind of problems. A data point of a set is considered as a p-dimensional vector (composed of p numbers) and the goal is to be able to separate the data with a (p-1)-dimensional hyperplane.

SVMs are good at handling feature vectors of both small and large numbers of dimensions, are fast at classifying and are relatively tolerant to noise, this type of classifier is used in this thesis [36, 37, 38], but all datasets are not easily divided by a linear model. Due to this, the Kernel trick was invented. A non-linear classification implicitly mapping their inputs into a different dimensional feature space.

SVMs can be used for classification, regression and outlier detection, but since the thesis has its focus within classification the theory covering the other two use cases will be omitted.

In order to classify a SVM constructs a hyperplane, or a set of hyperplanes in higher dimensional spaces, that can be used to classify data points depending on which side of the hyperplane they reside. As a hyperplane have two sides the classification is binary and thus the label set becomes defined as $\mathbf{Y} \in \{1, -1\}$. The optimal hyperplane $\mathbf{w}^T\mathbf{x} + b = 0$, as seen in Figure 4.1, is found when the given training data is separated with an as large margin $\gamma = \frac{1}{||\mathbf{w}||}$ as possible. This means that it will also be found when minimizing $||\mathbf{w}||$ or by simply minimizing $\mathbf{w}^T\mathbf{w}$.

Given training data points, or vectors, $\mathbf{x_i} \in \mathbb{R}^p, i = 1...n$ and the respective label

$y_i \in \mathbf{Y}$ the primal (4.1)

$$\begin{aligned} &\min_{\mathbf{w}\in\mathbb{R}^p, b\in\mathbb{R}} \mathbf{w}^T\mathbf{w} \\ &\text{subject to} \\ &\forall j\colon (\mathbf{w}^T\mathbf{x}_j + b)y_j \geq 1 \end{aligned} \tag{4.1}$$

can be constructed.

As soon as a stable hyperplane has been found the test set can be classified by simply checking which side, of the hyperplane, the data points end up on by computing the label value (4.2)

$$y_i = sign\left(\mathbf{w}^T\mathbf{x}_i + b\right). \tag{4.2}$$

The larger the distance from the hyperplane to a point the more certain the prediction is that the data point belongs to a certain category. Hence the data points that are within the margin of the hyperplane have the most uncertain predictions. This can, in fact, be used in order to calculate some certainty that a data point belongs to a class or not. If the distance between two data points and the decision boundary compared is of different sizes, the point with the greater distance is more probable to belong to the desired category [39].

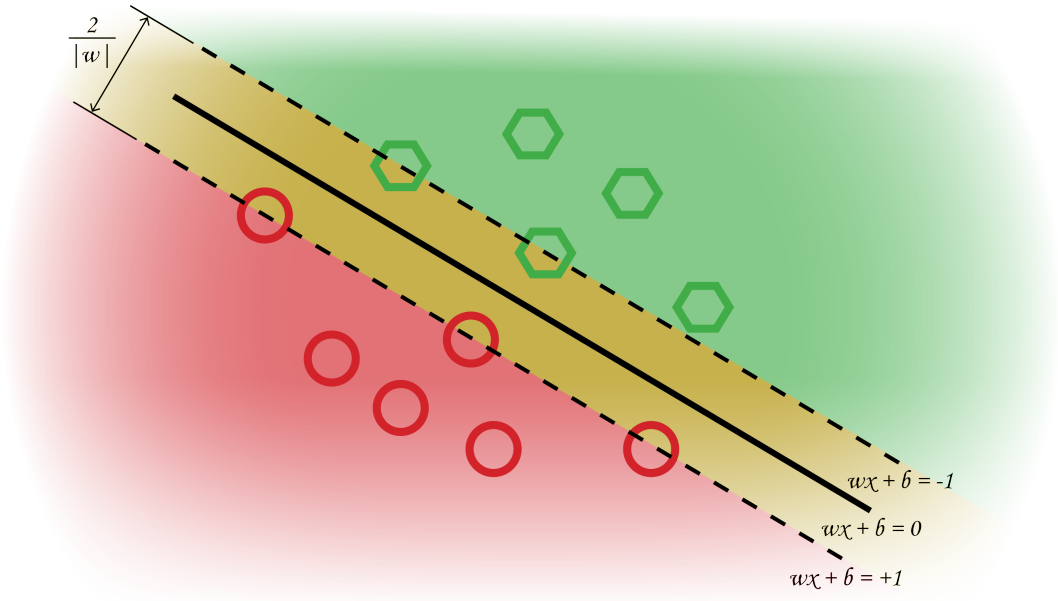


Figure 4.1: A simplified visualization of how data is linearly separable in a two dimensional space. Best viewed in color.

4.3.1 Kernels

Kernels define the Cartesian product between vectors, which can be used to get a real value. In order to create a kernel one defines a function K from the Cartesian product of the feature space to a real value ($K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$). This real value

can subsequently be used to evaluate a distance value. Depending on the chosen kernel the distance between different points vary and thus the choice of a kernel can improve the results for different datasets. Kernels in SVMs are different methods of how the hyperplane is generated for the SVM and thus gives different ways of separating the data. The most common ones are the linear and the radial-basis function (RBF) kernels. If the data is linearly separable a linear kernel is the straightforward approach. An example of a kernel that could solve the problem if this is not the case is the RBF. The RBF kernel is a fast approach that often works, as long as the feature space is not too large. Different kernels are used to make data points linearly separable in their own dimensional spaces, causing the decision boundary in the original feature space to be, and appear, non-linear.

4.4 Ensemble learning

Ensemble learning methods use a setup of different learning methods which are then combined to achieve a better predictive behavior than if using a single one. There are several things to be taken into consideration when implementing. The different feature descriptors need to show some form of diversity in their representations otherwise it will only create multiple calculations for the same information which would risk overfitting [40, 41]. One can use the Condorcet jury theorem as an example of this methodology, *"If each voter has a probability p of being correct and the probability of a majority of voters being correct is P, then p>0.5 implies P>p. In the limit, P approaches 1, for all p>0.5, as the number of voters approaches infinity"* [42, 43]. There is great potential with the use of several classifiers. It is important use different feature descriptors so they yield results based on independent data, else the idea of using more classifiers becomes redundant [44].

4.4.1 Deep support vector machine

A Deep support vector machine (Deep SVM) is model aimed to enhance the performance by using multiple SVMs by positioning them in layers, inspired by deep belief networks [45] and other stacking generalization approaches using SVMs [46]. The idea is to build layers of SVMs where the output of the previous layer becomes the input of the next layer. The use of more layers gives new possibilities in classifications which cannot be achieved with single, however complex, kernel functions. One example of this is the XOR function which can not be solved with a single SVM but can be when using layers of SVMs. The structure can be perceived in Figure 4.2, where the initial box, presenting the different feature vectors to the classifying system. The first layer of SVMs receives the feature vectors as input and the output of the first layer becomes the input of the following layer, called meta-level feature vector. The number of classifiers in the first layer is only restricted by the number of feature vectors presented to the system, which is an arbitrary number. The SVM in the final layer, in the figure called "Meta SVM", presents the final result of the system.

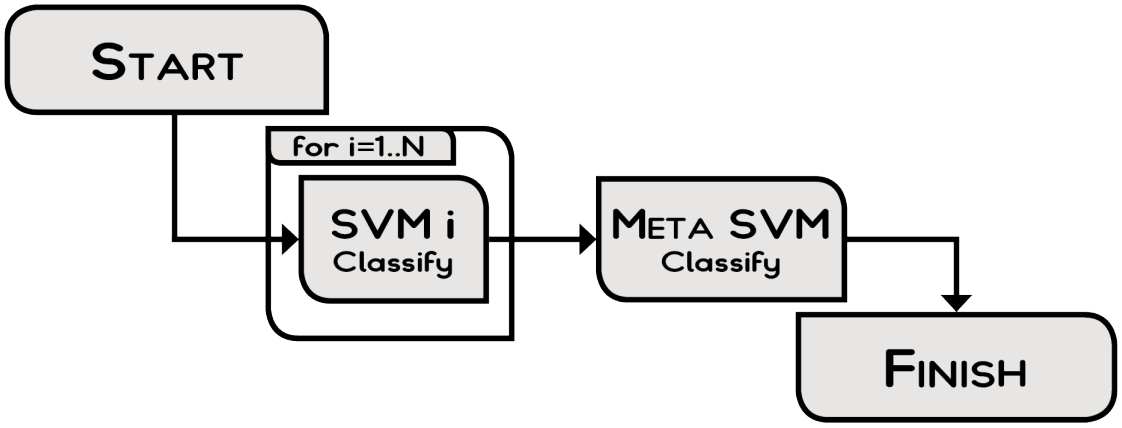


Figure 4.2: A simplified sketch of how a Deep SVM classifies data. The test data is passed through the first order classifiers and the output of those is the input for the Meta SVM. The output of the Meta SVM is the distance from the decision boundary that the entire classification system has created.

The training of a Deep SVM is performed in several steps, as presented in Figure4.3. The first step is to perform a K-fold split on the training data, $T = \{T_1 \cup ... \cup T_K\}$, that is applied to the classification system. All the first layer classifiers (first order classifiers) are then trained with the training subset $T_1^c = T \setminus T_1$, in order to use the remaining subset of the training set T_1 as a test set. The output of the first order classifiers then becomes the training subset for the second order classifier $T_{1_{meta}}$. This process is repeated K times to produce the full training set for the second order classifier T_{meta}. When the K-fold process has been completed the first and second order classifiers can be trained with the full training set T and T_{meta} respectively. The process of training a single SVM is described in Section 4.3.

This setup takes much more time to train than when just using a single SVM. The reward is an estimator that is capable of a more abstract level of classification.

There is a drawback to having two layers and creating estimation by using the K-fold process. In order to make estimations for relevant one data point another one is required to be in a different cake piece of the split. The same rule goes for the non-relevant datapoints. This means that the smallest training set for such a classifier is of size four (two relevant and two non-relevant data points). Even if this is a minimum size, the training process might be inconsistent if the training set is to small. Having a to small training set might cause the decision boundary might flip completely based on what training data is used. To avoid the training data to be inconsistent the number of folds can be increased, but to perform a 5-fold one needs five relevant and five non-relevant images. In other words the classifier works better with larger training sets were data overlaps can occur.

Which kernel functions that the different classifiers have in the Deep SVM does not matter since each unit is independent of the other ones. The selection of kernels in the first order classifiers depends on which feature vectors that they have as input. The classifier at the second layer, however, separates an n number of dimensions if

there are n classifiers in the first layer since they have all produced an estimated distance to a decision boundary. Due to the low level of dimensions and the values of the input vector should be positive if a point is predicted correctly, a linear kernel is often possible to apply.

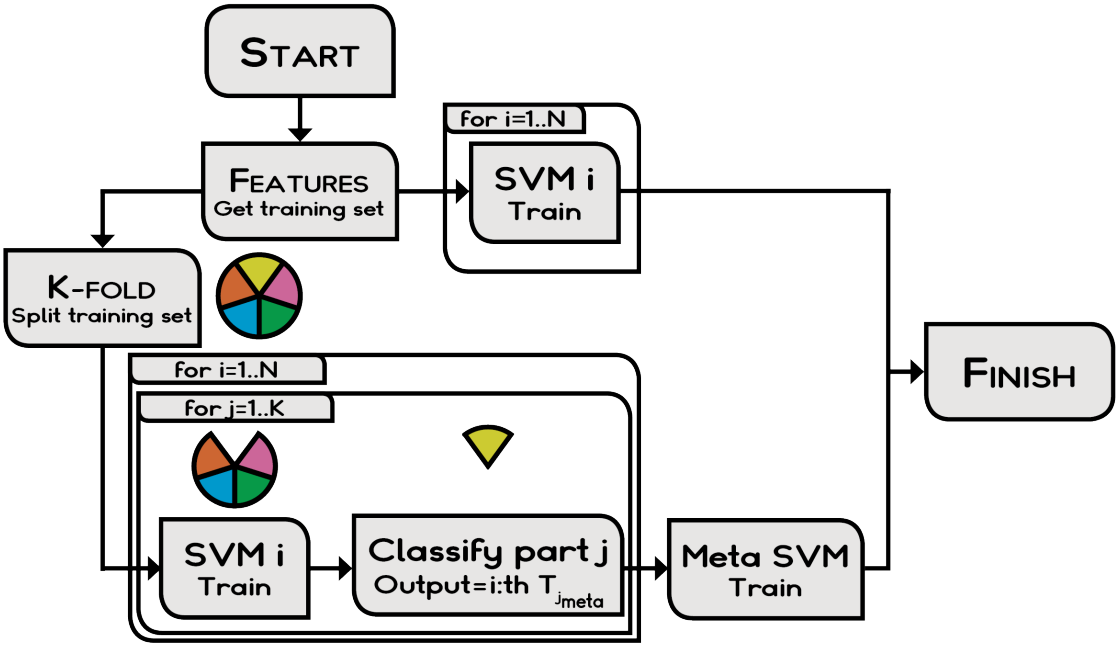


Figure 4.3: A simplified sketch of how a Deep SVM is trained. The Meta SVM needs approximations of how the first order classifiers treat the training data in order to fit its own decision boundary.

5
Method

This is where the methodology of the thesis is presented. Related approaches are presented as well as how the proposed model is structured. In the end of the chapter there is a presentation of how evaluations are performed and what the evaluations are.

5.1 Related approaches

The approach of modeling a system presented in this thesis is, to our knowledge, still untested. There are however several papers that implement different components of the proposed model. Most content-based image retrieval (CBIR) systems use a set of different feature descriptors that are proven to be good at finding equalities or similarities between images. The system is then presented with a single query image in order to find matches in a database. The images in the database are compared to this query image and the similarities are calculated which is usually some distance measure. If a certain threshold is crossed the images are labeled as similar [16, 17, 18], e.g. calculating the Euclidean distance and sorting the data having the most similar data points first. Other CBIR systems have created a feature vector from extracting certain feature descriptors, trained a neural network with a subset of the data and use the classifier to retrieve images [19]. There are other implementations where the use of relevance feedback is used in conjunction with the feature descriptors to even further increase performance, in light of the difficulty to identify feature descriptors that are good at "understanding" concepts [47].

5.2 Proposed model

This thesis presents a model that uses relevance feedback and CBIR in order to categorize a search space of unlabeled data in an iterative and a more efficient manner. The material that has been verified or recategorized by the user through relevance feedback in previous iterations can be used by the model to present better matches in future iterations. The unlabeled search space will in other words shrink as the labeled set for training will grow.

As mentioned in Section 5.1, the proposed model slightly deviates from other setups. Yet, the general structure is the same as most CBIR systems use. The proposed model consists of three modules; one for matching, one for feature extraction and one that handles relevance feedback. When a search iteration is initiated the matching module fetches a training set and information about the current search space from the feature extraction module, makes elaborate predictions about the material and passes the most accurate information to the relevance feedback module. The relevance feedback module processes the information, requests feedback from the

user, passes the ground truths on to the feature extraction model, that updates the search space with new information, and then terminates the iteration which allows a new one to be started. The communication between the modules during a search iteration can be seen in Figure 5.1.

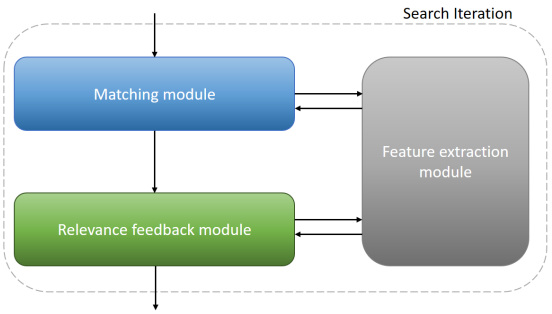


Figure 5.1: The system consists of three modules; matching, relevance feedback and feature extraction. Here the workflow for the model during a single search iteration is presented.

5.2.1 Relevance feedback module

There are numerous ways of using relevance feedback in order to improve CBIR and to categorize material. In Section 2.2 the different ways of relevance feedback are divided into three categories and they are referred to as explicit, implicit and blind feedback. In order to make elaborate guesses the model needs to have validated data in its training set and as mentioned in Chapter 1 all the case material has to be handled by an investigator in order to build a case. The model has therefore been designed to use explicit feedback in the end of each search iteration.

The feedback that the model receives from the user gives the module information about which images that were correctly categorized and which images that were falsely categorized as negatives and positives. The information of which images that have been categorized as false negatives or positives could be used in order to improve classification in some direction, but due to limitations of the scope the information from the relevance feedback reduced to only consist of ground truths. Since the model is designed to deplete the search space of relevant images, the only information that is drawn from relevance feedback is the knowledge of which images that are relevant and non-relevant for the specific case. This knowledge is passed on to the feature extraction module and the search iteration is then terminated as seen in Figure 5.2.

The information that the relevance feedback module receives from the matching module is overly simplified to reduce calculations. The material that is received is sorted to have the most relevant images first and the least relevant images last. The material is however only labeled as relevant and non-relevant. To present all the material at once would be overwhelming for the user and the model would not need to learn iteratively. To present the top-k images is a common method within CBIR, and the number of images that are presented could make it easier for the user to

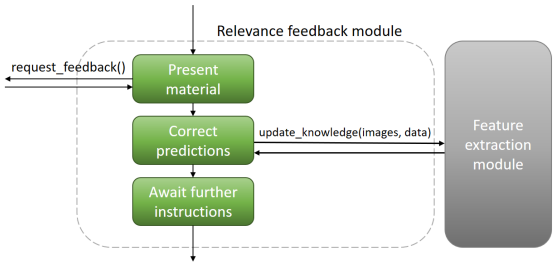


Figure 5.2: The relevance feedback module is the intermediator of the user and the rest of the model. When the feedback is given by the user the search iteration can be terminated.

oversee the material. The number 20 was arbitrarily chosen and was empirically manageable. An extension to this top-20 approach in order to improve classification was to also to present some images from the other extreme; the bottom-k images. With the intent to avoid presenting to much information to the user it sufficed with 5 images. Resulting in that the user could quickly give feedback to 25 images in total every iteration. The setting of which images and how many of each group of images that are presented each iteration was not easy to set. This decision paned out to be made using measurements and is described in Section 5.3.2.

5.2.2 Matching module

As a search iteration is initiated the matching module begins with retrieving training data from the feature extraction module. This training data consists of relevant and non-relevant images that can be used to fit the classifier of the model. When the classifier is set up, the search space can be retrieved from the feature extraction module and then be explored. The search space is processed in one batch at a time to avoid performing predictions for more images than necessary. When the exploration of material has resulted in a sufficient amount of material, the material is sorted from most to least relevant and then passed on to the relevance feedback module with labels of the material being relevant or not. In Figure 5.3 there is a visualization of the workflow of the matching module during a search iteration.

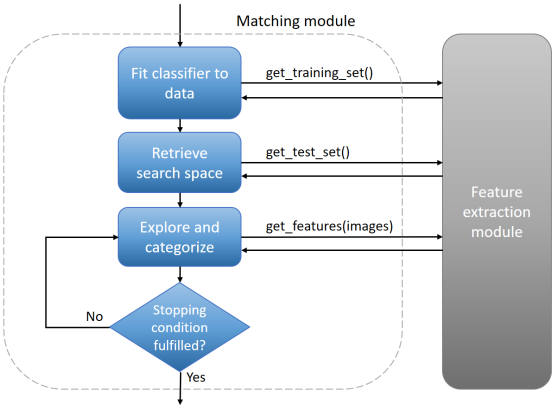


Figure 5.3: The matching module performs the initial work of a search iteration. At this point the feature extraction module provides with a training set and details about the search space.

5.2.2.1 Classifier

As mentioned in Section 1.3, there was no plan to compare how well different classifiers would perform in this thesis. Since having many dimensions can result in more general predictions, a classifier that scales well is preferable. A classifier that is capable of handling a high number of dimensions is the Support vector machine (SVM), see Section 4.3. Comparative studies such as [48], [49] and [50] have deemed SVMs as classifiers that continuously show good results in different implementations.

In the field of CBIR there are myriads of different feature descriptors that are used and the more feature descriptors one can combine, the more general the classifier can become. As mentioned in Section 1.3 the number of different feature descriptors used in this thesis is limited to five. This solely because adding or removing feature descriptors could improve performance in a general sense, but there was not enough time to cover the subject. In order to combine these five feature descriptors a tree of SVMs were created; one for every feature descriptor and one that treats the output of the different SVMs as input. Read more about the classifier structure Deep SVM in Section 4.4.1.

As mentioned in Section 5.1, most CBIR systems can quantify certainty of relevance by using a distance function in order to sort the material from most to least relevant. SVMs are geometrical tools that create a decision boundary in some dimensional space to split the categories so they can be categorized depending on which side they are of the decision boundary. Due to the fact that the relevance feedback module expects the material to be sorted the certainty of the classification needs to be quantified. Instead of using the *sign* function, Equation (4.2), to determine the category of an image, the matching module can use the distance between each data point and the decision boundary, a quantification method that is mentioned in Section 4.3, to see how probable it is that some image belongs to a certain category. This gives the matching module the possibility to sort the data from the most to the least relevant and the categories of the data points can still be predicted by using the sign function later on.

5.2.2.2 Training data

The training data used to fit the classifier is mainly intended to have been categorized recently by a user. As mentioned in Section 5.2 the search space (the unlabeled material) shrinks as the labeled set grows for every iteration. The labeled data is used to create a training set in order to fit the classifier and make new predictions. In the first iteration however, there is no labeled set to work with and therefore no training set for the classifier. When the classifier can not be fit the exploring of the search space is not possible. Instead of locking the workflow of the proposed model the matching module instead selects some material at random (without any predictions) to pass on to the relevance feedback module which creates some labeled data to use in future iterations.

In order to fit an SVM at least one relevant and one irrelevant data point is necessary and in order to fit a Deep SVM it is necessary to have two relevant and two irrelevant data points, due to how it is trained (see Section 4.4.1). When selecting material from the search space at random there is no guarantee that enough material is sampled in order to fit the classifier correctly within a certain number of iterations. To work around this issue the possibility to install an initially predefined training set was introduced. The initial training set is used in combination with the labeled data, that will slowly grow with every search iteration. A justification to add such a training set to the model is that in most cases when a person knows what to look for it has some previous knowledge of how such material would appear. When having a predefined training set with at least 2+2 relevant and irrelevant images, it is always possible to fit the classifier and the search space will be explored. This results in being able to perform actual predictions and having better odds of finding more relevant material to improve the training set even more.

Not having enough training data results in the incapability of fitting the classifier correctly, but having too much training data would force the classifier to take too much time to fit the best possible decision boundary for the data. Since the size of a search space could theoretically be infinitely large, the labeled set would also go towards infinity as the number of iterations increases. To prevent the labeled set to become too large an upper size limit was set to 500 data points. No evaluation or research was put into this number. Instead it was selected by the intuition of having a training set of that size (500 data points) it should be possible to present a small portion of material with some certainty. Having an upper limit of training material adds the possibility of two things: The time spent training the classifier is done in the same amount of time every iteration and by sampling a subset from a large labeled set allows the decision boundary to shift in-between iteration.

5.2.2.3 Exploring search space

After the search space has been received from the feature extraction module the exploration loop can begin. In this loop the classifier in the matching module is used to categorize the material and calculate the distances from the data points to the decision boundary. As mentioned in Section 5.2.2, this is done in batches. The

batches of the search space are selected at random, i.e. a subset is sampled from the search space. When a batch of material has been selected the feature extraction module provides the feature descriptors for the material in the batch. The material is presented to the classifier and receives a calculated distance from the classifier.

In order to avoid sampling batches until the entire search space is depleted every search iteration, three stopping conditions were introduced. The first rule was implemented to eliminate the risk of an infinite sampling loop. If a sampled batch only would contain material that already has been run through the classifier during the same iteration, the search is over. This stopping condition only exists because of how the batches are selected from the search space and will therefore not be evaluated in the same manner as the other two. The second rule is called Early stopping: If no relevant images seem to be found, stop the iteration and make the relevance feedback module present the least relevant images. When 200 data points have been passed through the classifier and none of them are categorized as relevant the material with the largest distance from the decision boundary is tallied up. Finally, the third rule is called Threshold: Meaning that a number of images has to be further away from the decision boundary than a certain value. This value is initially set to 1 until at least one image in a sampled batch has been passed through the classifier during the same search iteration. As soon as this occurs the limit changes to a function value depending on the search space size and how many unique images that has been sampled during the search iteration. When using the stopping condition Threshold the value of the threshold (5.1)

$$\text{Threshold} = \begin{cases} \text{unique images only} & 1 \\ \text{otherwise} & \frac{n - x log(x)}{n} \end{cases}, \tag{5.1}$$

where n is the search space size and x is the number of sampled images, is calculated in the beginning of each exploration iteration.

5.2.3 Feature extraction module

The feature extraction module is really not a part of the workflow during a search iteration. It is, however, a necessary supporting module that centralizes the control of information regarding search space, predefined training sets and the feature descriptors that are extracted. It provides the matching module with information about the search space during search iterations and delivers a training set in order for the classifier to work. The feature extraction module updates the search space with ground truths when the relevance feedback has been received and makes sure that the presented data can be used as part of the training set.

When the matching module is exploring the search space, the feature extraction module ensures that the feature descriptors of the material that is about to be classified are extracted. The process that occurs in the background is visualized in Figure 5.4. If the feature descriptors for a data point are not extracted, the module extracts them and stores them in the databases for future use as well. As mentioned in Section 1.3, only five different feature descriptors are extracted and used. The

weak learners were picked from the classes of feature descriptors mentioned in Section 3.1, where two were oriented towards shape, and one each for texture and color respectively. The convolutional neural network was chosen as the strong learner with the potential to categorize well on its own and the VGG-16 was chosen as it had a good performance in ILSVRC-2014 and is open for use.

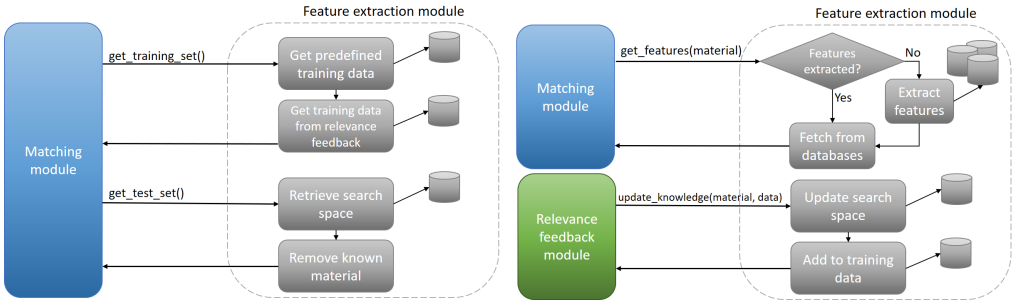


Figure 5.4: The feature extraction module centralizes the control of information transferred between the modules of the proposed model. The feature extraction module provides data to the matching module and updates the search space with information that is passed on by the relevance feedback module.

5.2.3.1 Histogram of oriented gradients

The histogram of oriented gradients (HOG) are implemented using the algorithms developed by `scikit-learn` [51]. The HOG is presented with an image that is converted into a single color channel, gray in this case. In the installment from scikit a number of parameters need to be set. The parameters are the following: The number of pixels per cell and cells per block are selected to determine how large the pooling squares become. The number of orientation bins is selected as well to set how many possible gradients that should be in the output for each block as described in Section 3.2.1. In this implementation 8 orientations were used, with 32-by-32 pixels per cell and 4-by-4 cells per block. When the convolution is complete the final step is to collect the data from all the blocks and create a feature vector.

5.2.3.2 Global color histogram

The Global color histogram is based on the one proposed in [47]. The color range of the image is changed to Hue, Saturation and Value (HSV). The intensities of the channels are then stacked depending on which values they have and divided into a number of predetermined bins. In this project they are set as 24, 12, 6 respectively, instead of the 8, 4 and 2 which are used in Wang et al. implementation [47]. Every bin is summarized with every other bin in the other two channels which gives 24*12*6=1728 different combinations which are divided by the total number of interest points as,

$$H(i) = \frac{n_i}{N} \text{ where } (i = 1, 2, ..., 1728), \tag{5.2}$$

where n_i number of interest points, N is the total number of interest points and i is the possible combinations of the bins. $H(i)$ is used to create one of the feature descriptors used by the classifier described in Section 5.2.2.1.

5.2.3.3 Wavelet transform

The Haar wavelet transform has good performance at a cheap computational cost compared to many other wavelet transforms explained in Section 3.2.3. In the proposed model, 3 levels of transformation is used. The first decomposition is done by applying two filters on the rows and columns of the original image. This results in four new images where one is the average of the image and the other three represent details. This is done twice more as it is a 3 level decomposition. At the end the values from all these resulting matrices are placed in an array to form the feature vector, the implementation is based on the one proposed in Wang et al. [47].

5.2.3.4 Convolutional neural network activations

In the proposed model a pretrained neural network is used. The network is the 16-layer network used by the VGG team in the ILSVRC-2014 competition [22], also called VGG-16. The network is presented with a 256×256 image as input which is processed by the entire network. The feature descriptors are extracted from the activations of the last fully-connected with 4096 neurons, as per Section 3.2.4. The vectors are concatenated into a feature vector presentable to the SVM described in Section 4.3.

5.2.3.5 Edge detection histogram

The Sobel edge detector is implemented with four directions. The directions are horizontal, vertical, 45° and 90°. In addition, a non-directional edge filter is added. The pixels are put into different numbers of bins that represent the quantitative states of the pixel counts of the images in their respective orientations. The locations without edges are also formed into a separate bin. The edges found in the image are binned as detailed in Section 3.2.5. The output is a feature vector of values, describing the shapes shifts present in the image.

5.3 Evaluation of model

Performing evaluations on a larger scale often demand that labeled datasets are used, whilst the proposed model is designed to have a user standing by each iteration. This section covers how these evaluations were performed in order to achieve the results presented in Chapter 6.

The feature extraction module, Section 5.2.3, is described as a system that will extract the necessary feature descriptors for material evaluations in real time. How-

ever, to reduce the number of factors that alter time taken while processing material, the feature descriptors of the search space are extracted before all evaluations begin.

The evaluation is divided into two parts: How benchmarks are performed in order to achieve the optimal settings for the proposed model and how the model compares with other CBIR studies. Because the model requires that a user performs the arduous task of peer-reviewing the predictions of the model every iteration, a simulation of the user was created in order to retrieve data from evaluations.

5.3.1 Relevance feedback simulation

Relevance feedback is, as the proposed model suggests in Section 5.2.1, used to help the model mount the semantic gap. A user peer-reviews the presented images each iteration and makes corrections where necessary to make sure that the entire dataset is labeled correctly. To do this on large data sets is both time consuming and would take quite a toll on the user. There is also no guarantee that a user will label the same image as the same category in two separate settings while searching for the same material. Since the evaluations are run on datasets that already are labeled this risk of mistakes can be reduced by instantiating a user simulation. The relevance feedback simulation takes the role of a user that communicates with the relevance feedback module. This way, time is saved and no user needs to be present in order for the evaluations to be performed.

The simulated user will not be mislabeling images because of negligence or exhaustion as a human would. A normal user would fail to label material correctly in the same extent as a simulated one would. However, the error rate for a "trained" human is extremely low, comparable to the best neural networks that compete in the ImageNet Large Scale Visual Recognition Challenge (ILSVRC), e.g. GoogleNet [52]. In this case a "trained" person refers to a person that is aware of the situation and well versed in classifying images. Yet, this still means 4% misclassification compared to 0% that is derived when a user is simulated using the labels of a dataset. The data sets presented in Section 2.4 are very diverse and in some cases a simulated user would categorize the material differently than an ordinary one would. This is however not a problem rooted in how the simulated user works but how the datasets are designed and categorized to begin with.

5.3.2 Parameter benchmarks

The presentation of the proposed model in Section 5.2 covers the most part of how the optimal implementation is designed. What it does not cover is if some of the design decisions are good or not. For example in Section 5.2.1 the selection of images to present to the user is mentioned but not elaborated. There are two stopping conditions introduced in Section 5.2.2.3 which need to be evaluated to see how they effect the performance. The five feature descriptors covered in Section 5.2.3 also need to be evaluated. Not only in combination but how they perform on their own, as well as the effect of the training sets mentioned in Section 5.2.2.2. In other words,

how much the behavior of the model can change depending on training set sizes. To summarize, the parameter benchmarks are the following four evaluations:

- The effect of presenting different sets of images to the user for relevance feedback (Section 5.3.2.2).
- The effect of pruning the search space during the search iterations (Section 5.3.2.3).
- How well the ensemble of learners performs compared to its parts (Section 5.3.2.4).
- The effect of using different kinds of training sets (Section 5.3.2.5).

Before specifying which metrics that are intended to be used in these evaluations, the metrics need to be defined. The two more commonly used measures are (5.3) and (5.4).

$$\text{recall} = \frac{\text{True positives}}{\text{True positives + False negatives}} \tag{5.3}$$

$$\text{precision} = \frac{\text{True positives}}{\text{True positives + False positives}}. \tag{5.4}$$

Often presented in pairs due to the fact that recall can reveal if the matching module presents too many false negatives while precision can indicate if too many false positives are presented.

A metric that measures the models effectiveness is the (5.5)

$$\begin{aligned} \text{F1-measure} &= 2\frac{precision \times recall}{precision + recall} \\ &= \frac{\text{True positives}}{\text{True positives} + \frac{\text{False positives} + \text{False negatives}}{2}}. \end{aligned} \tag{5.5}$$

The F1-measure is the harmonic mean of recall and precision [53], meaning that recall and precision are equally weighted and that smaller values are punished more than when using a normal average function.

The final metric to present is the accuracy (5.6)

$$\text{accuracy} = \frac{\text{True positives} + \text{True negatives}}{\text{True positives} + \text{True negatives} + \text{False positives} + \text{False negatives}}, \tag{5.6}$$

which simply put is the total rate of correct predictions.

Apart from these metrics the evaluations include how long the calculations of an iteration takes, how many images that are passed through the classifier in the matching module and how many of the presented images that are relevant.

In order to make sure that the search space is explored and categorized correctly and at the same time see how well the model performs at classification, two different methods are used to calculate performance. In addition to the search space an evaluation set, in complete disjunction to the search space, is used. The datasets for the benchmarks and how they are constructed is presented in Section 5.3.2.1.

The benchmark evaluations are measured in two different ways.

1. How well the model classifies an evaluation set each iteration. For the evaluation set the following metrics are used:
    - Recall
    - Precision
    - F1-Measure
    - Accuracy
2. How well the model classifies the images that are presented to the user each iteration. For the search space the following metrics are used:
    - Accumulated recall
    - Accumulated precision
    - Accumulated F1-Measure
    - Accumulated accuracy
    - Retrieved relevant images
    - Handled images
    - Time taken calculating

When measuring the performance of classifying the search space the metrics are the accumulated value of the performance so far during the search. This mean that after iteration 20 the performance is measured on the 500 images that have received an prediction by the proposed.

In order to retrieve a general trend, each evaluation setting is run five separate times and the metrics during these runs are presented with the maximum, the minimum and the geometric mean of each iteration. This allows the graphs that are presented in this section to show how much the metrics varied depending on different factors, such as image selection or how the decision boundary was fitted.

In all benchmarks, except for the one in Section 5.3.2.4 where the ensemble is evaluated against its parts, the classifier is implemented as proposed in Section 5.2.2.1 and will have use all five feature descriptors. In addition, all benchmarks, with the exception of the one described in Section 5.3.2.5 where training data is evaluated, the matching model will have a predefined training set of five relevant images and 50 non-relevant.

The evaluation of the model is intended to be fair and the performance to be measured in an as general way as possible, but still able to perform within a relatively small time frame. The datasets for the evaluation are therefore constructed specifically for this purpose.

5.3.2.1 Datasets for benchmark

Since scenery datasets such as the dataset Places205, presented in Section 2.4.2, have a broad base of image material and have a large variety of material within the categories, Places205 is perfect for a parameter benchmark. However, due to the size

of the dataset, the decision to only use a subset was made. Instead of using all 205 different scenery categories a subset of 23 classes was cherry-picked; all categories with a name starting with the letter B. This gives a $\approx 4.3\%$ chance that a randomly selected image is relevant. Since only 25 images are presented per iteration using the entire subset as a search space would cause the number of iterations to be ≈ 11000, the search space was reduced a bit further. The search space during the benchmark evaluations consists of 200 images of each category where one of the categories is marked as relevant and the others are not. The choice of just using a part of the dataset makes it hard to compare with other implementations of image recognition implemented on this dataset. As well as the fact that the algorithm is designed to be more lightweight and less time-consuming than the more usual approach of using large deep neural networks.

In Section 5.3.2 an evaluation set is mentioned. The Evaluation set is constructed by using 50 images, that are not represented in the search space, of each category. This results in having an evaluation set of 1150 images with the same probability of randomly selecting a relevant image as in the search space.

In addition to the 250 images of every category used in the search space and the evaluation set, 250 images of every category were sampled in order to have material for different predefined training sets. The different sizes of the predefined training sets vary and how they are evaluated is presented in Section 5.3.2.5. The different training sets are simply constructed to have a broad base of irrelevant images, sampling some images from all categories that are not relevant, and the smaller sets of relevant images are subsets of the larger sets of relevant images.

To make sure that the thesis covers a wider base of data, the benchmarks have been performed with three different categories marked as relevant in three different evaluations. These three categories are Bar, Baseball field and Bedroom. The three categories does not have very much in common but some of the other categories in the dataset might have some similarities. Having three different categories results in having three different search spaces, three different evaluation sets and three different setups of training sets. The data drawn from these categories will be presented in parallel.

5.3.2.2 Classifier learning method

Since the proposed model is designed to acquire deeper understanding of a concept for every passing iteration, it is important that what the model learns from is chosen in an appropriate manner. In each iteration 25 images are presented and this evaluation is designed to decide how these 25 images should be selected. The best setting in this benchmark is used in the evaluations that are performed later. The goal of the evaluation is to find a method that ensures that the model learns the concept as fast as possible but still perform well enough in order to reduce work for the user.

Four distinct settings were chosen that are representative for how the data can be selected and still processes the dataset in an efficient manner. The different settings

of the evaluation are

1. **Top20+Bottom5**: To present the 20 images that the classifier finds the most relevant and the 5 images that the classifier finds the least relevant. As mentioned in Section 5.2.1, this is the how the model is designed to present material.
2. **Top25**: To present the 25 images that the classifier finds the most relevant.
3. **Top20+Middle5**: To present the 20 images that the classifier finds the most relevant and the 5 images that are the closest to the decision boundary of the classifier.
4. **Top5+Bottom20**: To present the 5 images that the classifier finds the most relevant and the 20 images that the classifier finds the least relevant.

In order to make the different settings deviate as much as possible, the model processed the entire search space every iteration. Thus making the selection of images each iteration more predictable and the measurements of each setting are less diverged in between the five different runs.

5.3.2.3 Limiting search space

Evaluating the entire search space every iteration is not just time consuming but also unnecessary since only 25 images are presented at a time. The stopping conditions presented in Section 5.2.2.3 are therefore evaluated to measure their effect on performance. As previously mentioned the first stopping condition – if a sample from the search space only contains material that has been passed through the classifier the same search iteration the exploration is over – is always in use to prevent the possibility of an infinite loop while exploring.

The four different settings evaluated in this benchmark are

1. **All images:** To stop after the entire search space is evaluated.
2. **Threshold:** To stop after enough images are classified to be above a decision threshold specified in Equation 5.1.
3. **Early stopping:** To stop if 200 images have been sampled but none are on the positive side of the decision boundary. In this case the relevance feedback module presents the 25 images that are considered the least relevant.
4. **Both rules:** The additive result of using setting 2 and 3.

The intention of the evaluation is to measure the trade-of between correctness in classification of the search space and time spent calculating instances to different data points.

5.3.2.4 Feature descriptors

To determine how well the classifier performs as an ensemble compared to only using its parts this evaluation has been constructed by using the first order classifiers either by themselves or in unison. When only using one classifier in the first order,

the second order classifier is given a linearly separable value in one dimension and thereby just passes on the information. The purpose of this evaluation is to see which feature discriptors that can discriminate which categories from the rest of the dataset, but it is also intended to evaluate if different combinations of feature descriptors will improve the performance of the model. Therefore the evaluation handles 7 different settings:

1. **HOG:** Only using the first order classifier for the HOG descriptors (see Section 5.2.3.1).
2. **GCH:** Only using the first order classifier for the GCH descriptors (see Section 5.2.3.2).
3. **WT:** Only using the first order classifier for the Haar wavelet transform descriptors (see Section 5.2.3.3).
4. **CNN:** Only using the first order classifier for the neural network activation vectors as descriptors (see Section 5.2.3.4).
5. **Edge:** Only using the first order classifier for the edge detection histogram descriptors (see Section 5.2.3.5).
6. **All:** Combining all the first order classifiers as intended and as in the proposed model described in Section 5.2.
7. **All-CNN:** Combining all the first order classifiers with the exception of the neural network activation vectors. This setting was added half-way through the evaluation since the setting **CNN** almost performed as well as the setting **All**.

5.3.2.5 Training data

Is it necessary to provide initial training data to the model in order to improve classification correctness in order to learn a specific concept? If so, how much difference would it make? This evaluation is designed to answer these two questions. A comparison will be made between only having a predefined training set, only using the data that is provided from the user during iterations and a combination of the two. Different sizes of the pretrained data sets were also evaluated to see how it effected the performance.

The different settings during the training data evaluation were the following:

1. To train the classifier once in the beginning with given data and use the same classifier until the search space is empty.
2. To train the classifier every iteration with data given by relevance feedback together with a predefined training set.
3. To train the classifier every iteration with data given by relevance feedback only.

The predefined training sets were assembled by different number of relevant and non-relevant images. The different sets were created after the different criterias:

A. The training set consists of 5 relevant and 5 non-relevant images.

B. The training set consists of 5 relevant and 50 non-relevant images.

C. The training set consists of 22 relevant and 484 non-relevant images (thus giving it the same relevance ratio as the search space).

D. The training set consists of 250 relevant and 250 non-relevant images (thus making it contain more relevant images than the search space does).

Since there is no good and concise way to refer to these different settings, they will in this section be referred to by their index in these two lists. The setting that only uses a predefined data set of 5 relevant images and 50 non-relevant images is referred to as setting 1B and the setting that solely uses data given by relevance feedback is referred to as setting 3.

As mentioned in Section 5.2.2.2, the model uses at most 500 images taken from the relevance feedback as training data. Those 500 images are intended to be as close to evenly divided between relevant and non-relevant images as possible. Due to the fact that the search space consists of 200 relevant images and 4400 non-relevant images, the training data in setting 3 consists of at most 200 relevant images and 300 non-relevant. Whereas setting 2D will have at most 450 relevant images and 550 non-relevant images in its training set since the predefined data are added on top of the data received from relevance feedback.

5.3.3 Study comparisons

Even though the proposed model is defined to become better at retrieving images iteratively and continually increasing the query set, there is still value in seeing how well a barely trained version of the proposed model compares with retrieval methods from other papers. Most CBIR system are designed to have a single image as a query, [16, 17, 18] among others, which is not possible with the proposed model. A model that uses an SVM (see Section 4.3) as classifier would require at least two (one relevant and one non-relevant) images in the query set in order to fit a decision boundary. The proposed model uses a Deep SVM (see Section 4.4.1), that uses a K-fold split in order to fit the decision boundary for the second order classifier, and does therefore require at least four (two relevant and two non-relevant) images in the query set. In order to have a fair comparison to the other studies the query set must be kept small. There are however studies that uses a training to tackle the challanges within CBIR [19]. A too small query set would be unfair towards the proposed model and the query set size will therefore be balanced to compare with the studies at hand.

5.3.3.1 The Corel-1000 evaluation

The Corel-1000 dataset is presented in Section 2.4.1 and is often used to compare different image retrieval methods. This evaluation is intended to measure how well the system handles the diversity of the different classes in a dataset. Given a query image, present 20 images that are similar and the precision on that set of images

is evaluated. The test is intended to be run with every image in the data set as query image and an average for every class is taken. There are in other words 100 data points for each class, but as previously described in Section 5.3.3, the proposed model requires at least two relevant images and two irrelevant images in the query set, which results in 4950 different combinations of relevant images in the query sets. Evaluating all combinations would be time consuming as well as pointless since the performance of the model can be observed in a much smaller number of evaluations. In order to test more combinations, the query set is sampled 500 times per class instead of only 100 times. This results in an approximation based on around 10% of all combinations of relevant images when having the smallest possible query set. In order to calculate how much the different query sets differs during an evaluation the variance of the result is calculated as well. When a query set is sampled the system processes the rest of the search space and presents the $n = 20$ most similar images. The retrieval precision for n images (5.7)

$$P(Q_{k_i}, n) = \frac{1}{n} \sum_{e \in \xi(Q_{k_i})} \delta(\Phi(e), \Phi(r)) \Bigg|_{r \in Q_{k_i}, \delta(\Phi(r),k)=1, |\xi(Q_{k_i})|=n}, \tag{5.7}$$

where Q_{k_i} is the ith query set for category k, $\xi(y)$ is the retrieved set for query set y. $\Phi(x)$ is the category of image x, $\forall images\ a,b : \delta(\Phi(a),\Phi(b)) = \begin{cases} 1 & \Phi(a) = \Phi(b) \\ 0 & Otherwise \end{cases}$. In short $P(Q_{k_i}, n)$ is the number of retrieved images that are relevant divided by the total number of retrieved images. The average retrieval precision for category k (5.8)

$$ARP_k = \frac{1}{m} \sum_{i=1}^{m} P(Q_{k_i}, n), \tag{5.8}$$

where n is the number of retrieved images, k is the desired category and m is the number of evaluations per category. As well as the total average retrieval precision (5.9)

$$ARP = \frac{1}{t \times m} \sum_{k=1}^{t} \sum_{i=1}^{m} P(Q_{k_i}, n), \tag{5.9}$$

where n is the number of retrieved images, m is the number of evaluations per category and t is the total number of categories in evaluation. If briefly described the ARP is simply what the average ratio is of the n retrieved images that are predicted as relevant and actually are of the intended category after m evaluations for every class.

In order to retrieve the variance for each evaluation the result is split into 20 equally sized subsets of which ARP is measured on. The variance is then derived from those and finally an average is calculated over the 20 variances for each subset.

Evaluation is run with $t = 10$ categories, $m = 500$ different query sets per category and number of retrieved images $n = 20$ resulting in that the model calculates distances to all the images in the dataset for at least 5000 times.

As mentioned in Section 5.3.3 the number of query images that will be used is determined by how the well other papers perform at the task. The most common

approach of CBIR is to use one query image and calculate a distance to other images in some dimension space, which is done in [16, 17, 18]. There are other approaches to the problem as well. In the case of [19], where the authors train a machine learning classifier in order to find images of relevance. One might argue that the proposed model is a combination of these two, where a small query set is fitted onto a classifier and checks it towards the test set. Even though there are similarities to the other models a fair comparison can not be made. In [16, 17, 18] the test set consists of the Corel-1000 dataset minus the query image, in [19] the test set only consist of a tenth of the Corel-1000 dataset, and in the proposed model the test set consists of the Corel-1000 dataset minus the query set.

6
Results

The proposed model as described in Section 5.2 is implemented and is presented in this chapter. The results of the parameter benchmarks and the CBIR evaluations that are introduced in Section 5.3 are presented here.

6.1 Parameter benchmarks

As mentioned in section 5.3.2 there are some parameters to evaluate in order to find the optimal setting for the proposed model. The evaluations that are presented in this section are:

- **Classifier learning method**: Described in Section 5.3.2.2 and presented in Section 6.1.1.
- **Limiting search space**: Described in Section 5.3.2.3 and presented in Section 6.1.2.
- **Feature descriptors**: Described in Section 5.3.2.4 and presented in Section 6.1.3.
- **Training data**: Described in Section 5.3.2.5 and presented in Section 6.1.4.

The different metrics used in this evaluations are described in Section 5.3.2 but some of the graphs are omitted in this thesis. The graphs that are omitted are not included due to inconclusive results and insignificant differences in-between the settings of the evaluations. In most cases, the information that is excluded by removing some figures from the thesis can be derived by interpreting the results in-between the metrics of different evaluations.

Recall that in Section 5.3.2.1 the datasets used for the evaluation is a subset of 23 categories drawn from the dataset Places205 (see Section 2.4.2) and all benchmark evaluations are performed five times with three different categories as the target.

6.1.1 Classifier learning method

The entirety of this evaluation is introduced in Section 5.3.2.2 and the results of the different settings are presented here. The four different settings of the evaluation are to present the **Top20+Bottom5**, **Top25**, **Top20+Middle5** and **Top5+Bottom20**.

As mentioned in Section 5.3.2 the performance of the model is measured in two ways. The performance on an evaluation set as well the performance over the entire search space.

6.1.1.1 Evaluation set

When classifying the evaluation set each iteration the measurements of the settings ended up to be very similar. The performance of the first three settings were almost identical. An initial performance peak in classifying the set that later on dropped off. While the performance of the fourth setting deviated from the performance of the others the measurements were about the same throughout all the iterations.

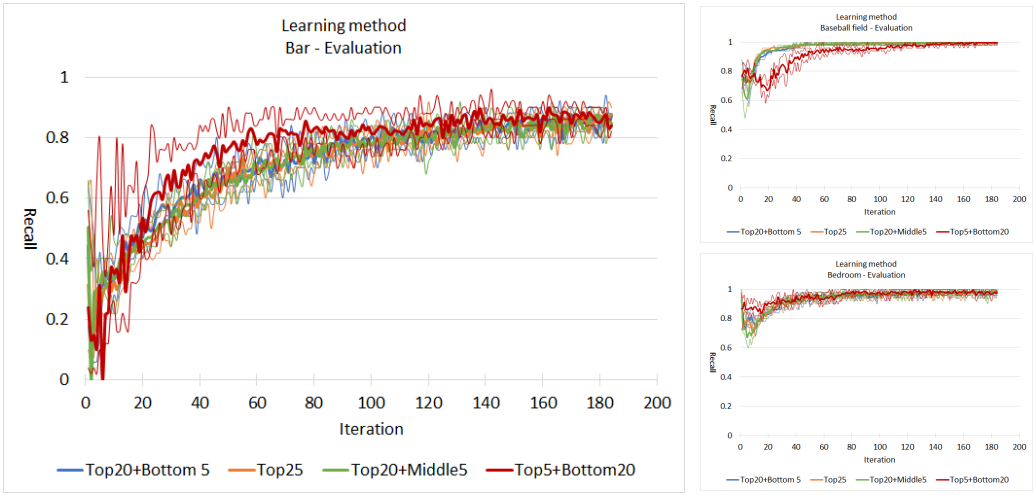


Figure 6.1: The recall rate on the evaluation set.

Independently of how images were selected the recall (see Figure 6.1) of the settings only differed marginally. With an exception of the fourth setting that in some manner deviated from the other ones. The precision of the fourth setting deviated from the other settings, which can be seen in Figure 6.2. Due to the low precision of the fourth setting, the F1-measure is low as well, which can be seen in Figure 6.3. The cause of this is that the fourth setting had an higher number of false positives than the other settings had.

An important observation to make is that towards the end of the evaluations all four settings have the same training data and therefore classifies the evaluation set accordingly. Having the complete training set results in an F1-measure around 0.32 when retrieving the category Bar and around 0.65 when retrieving the other two from the remaining 22 categories in the evaluation set.

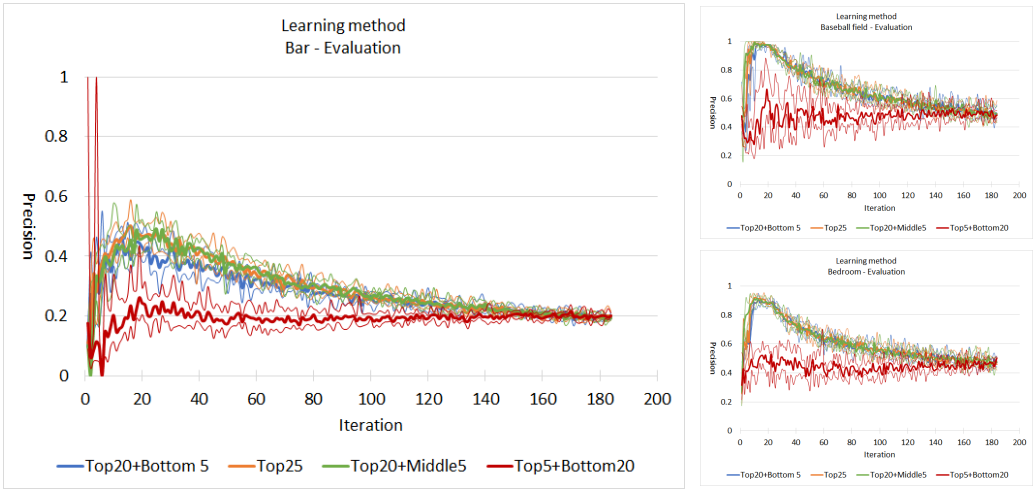


Figure 6.2: The precison that the different settings had on the evaluation set for the three different category searches. Note that the setting Top5+Bottom20 deviates from the other settings.

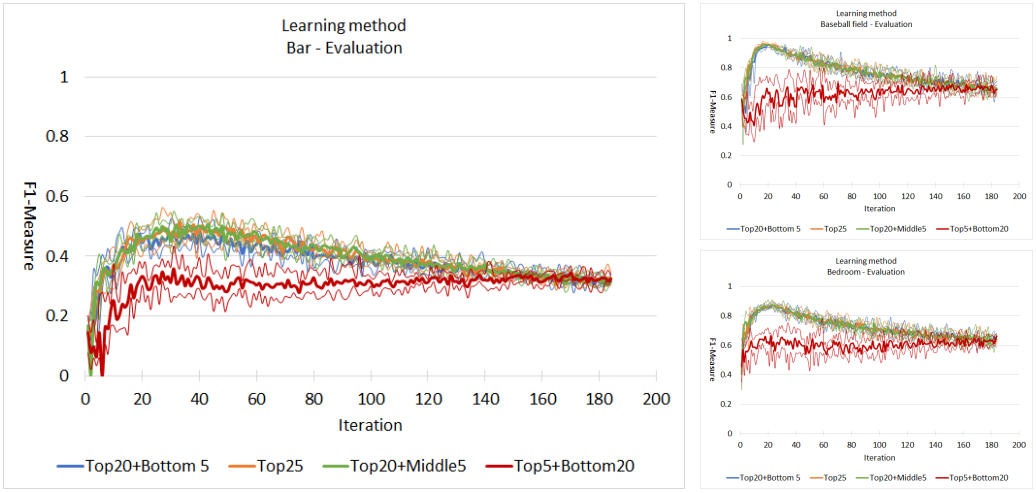


Figure 6.3: The harmonic mean of recall and precision, F1-measure, read on the evaluation set over iterations. The performance of the first three settings peak early and then drops towards the end of the evaluation.

The accuracy when using the different settings did not vary that much either. As seen in Figure 6.4, the accuracy of the settings is around 85% when classifying the category Bar and around 95% on the other two category evaluations. Just as with the F1-measure, the accuracy measurements of the first three settings were higher in the first couple of iterations and then dropped off towards the end of the evaluation.

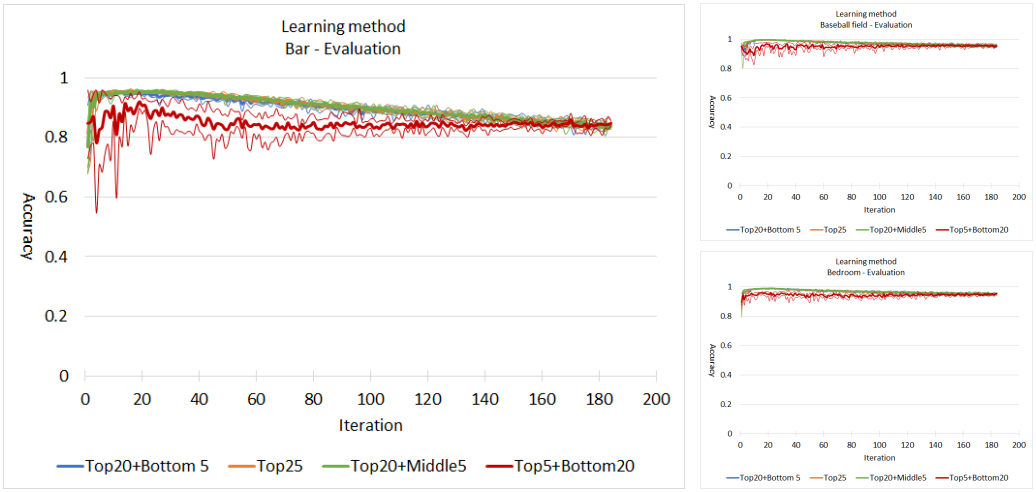


Figure 6.4: The accuracy of the different settings on the three evaluation sets. Note how the accuracy of fourth setting varied more in-between evaluations than the accuracy of the other three settings did.

6.1.1.2 Search space

The different settings deviated a lot more from each other when comparing how the search space was classified compared to when the evaluation set was. The first three settings stopped predicting images as relevant early on in comparison with the fourth setting, resulting in the precision presented in Figure 6.5. At the end of the evaluations the fourth setting had a precision of 15-25% in all three categories while the other three settings could maintain a precision of about 80% on the categories Bedroom and Baseball field. When classifying the third category, Bar, all the settings had a rather low precision. In this category, the first setting was the one showing the lowest precision. While the fourth setting had a relatively low precision, it did produce the best recall over the different search spaces as seen in Figure 6.6. When searching for the third category it took until the final iterations before the last relevant images were retrieved. This is rather late if one would compare with the settings, where the last relevant image was retrieved slightly after the first half of the evaluation.

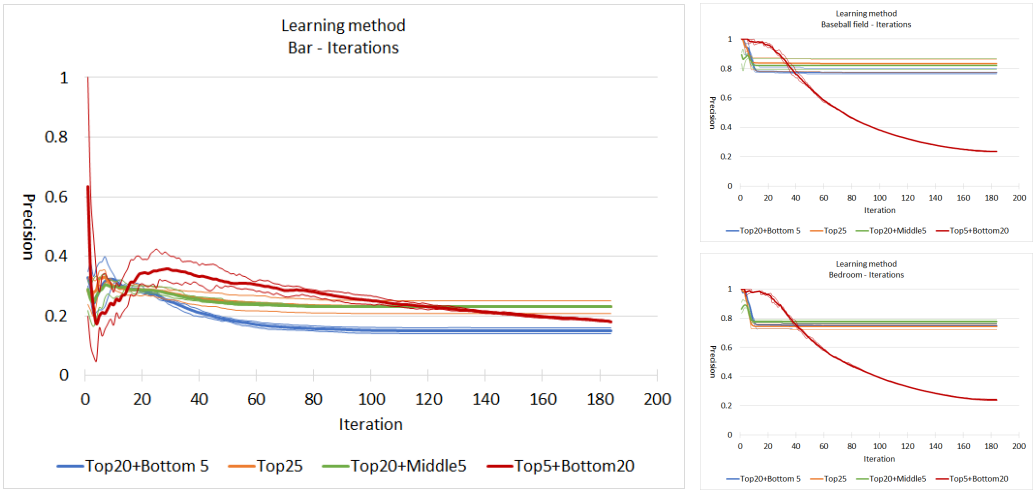


Figure 6.5: The precision that the different settings had classifying the search spaces for the three evaluation categories. Note how the setting Top5+Bottom20 continued to predict images as positives when they indeed were negatives throughout the entire search.

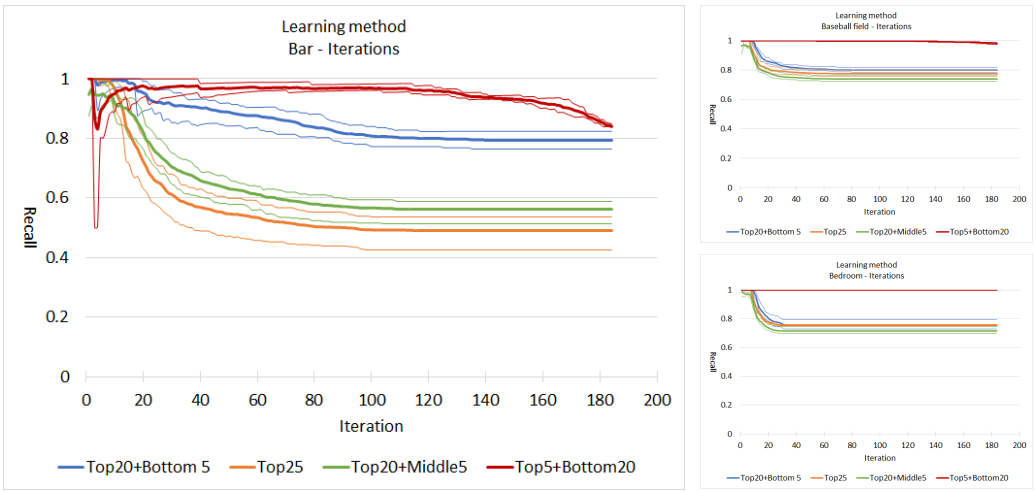


Figure 6.6: The recall rate on the three evaluation categories. The recall is slightly higher for those settings that continually present some of the bottom images in each iteration.

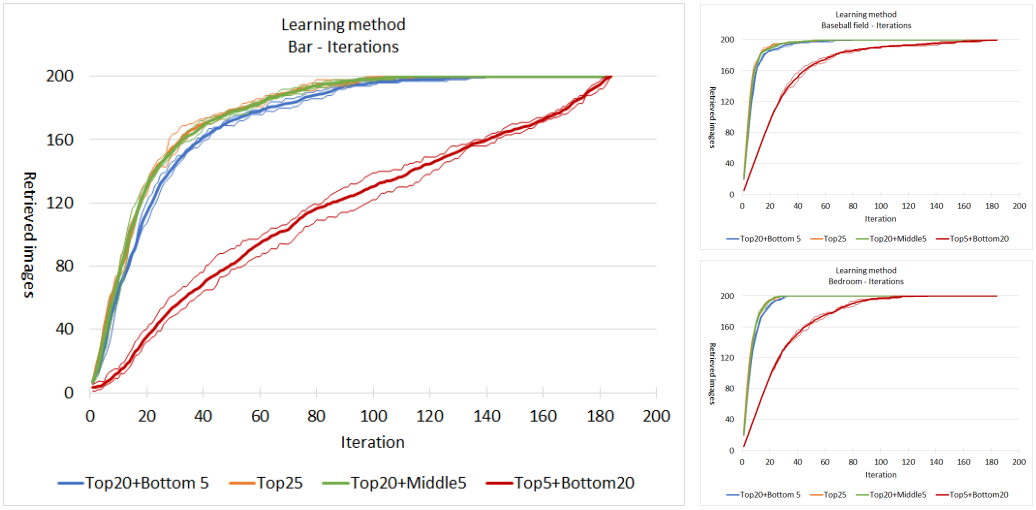


Figure 6.7: The number of retrieved images over iteration that the different settings had while classifying the search spaces for the three evaluation categories. Note how the category Bar is more difficult than the other ones to retrieve relevant images from.

In terms of performance as an image retrieval system the setting of presenting a majority of negatives is not to prefer. As seen in Figure 6.7, the number of retrieved images by the fourth setting is a lot lower than the number for the other settings over the entire run. Looking at the performance on the category Bar, there are some iterations where the fourth setting has retrieved fewer relevant images than when selecting images at random. This is only momentarily and the rate soon returns to being slightly above that. Reading into the F1-measure in Figure 6.8 and the accuracy in Figure 6.9, the data is out of favor of the fourth setting, but when inspecting the values of the category Bar, the setting Top20+Bottom5 has a slightly worse F1-measure and accuracy than the other settings.

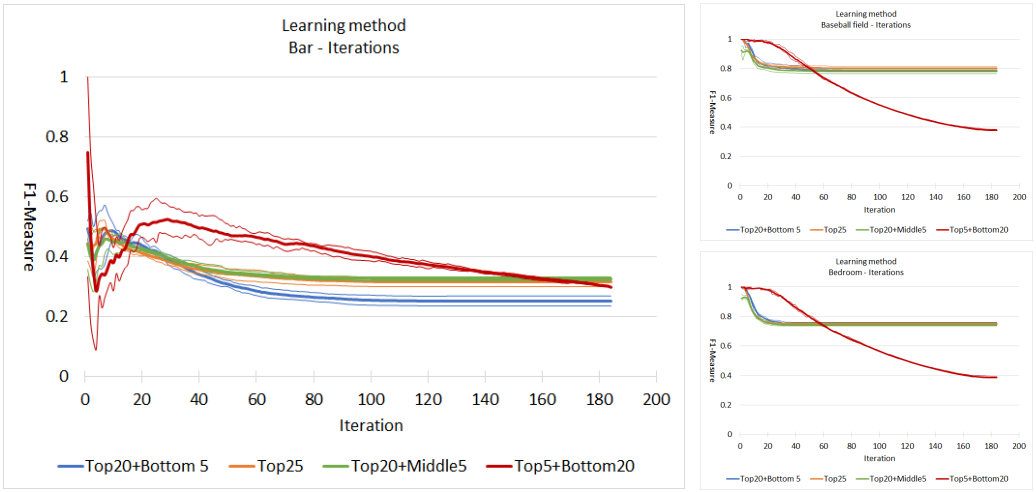


Figure 6.8: The F1-measure that the different settings had classifying the search spaces for the three evaluation categories. An harmonic mean of the precision and the recall.

The setting that will be used to select the set of images each iteration needs to be picked in order to continue the parameter evaluation. When working with image retrieval the model needs to have a high retrieval rate of relevant images early on which causes the Top5+Bottom20 setting to not meet the preferences. To select between the remaining three settings one can recall what was mentioned in Chapter 1: Investigation material needs to be retrieved in a quick manner and labeled correctly. In other words the model needs to produce as few false negatives as possible. In all three categories; the Top20+Bottom5 setting had a higher recall rate on the search space throughout the evaluations (see Figure 6.6). This means that the first setting is the most appropriate one to use in the following benchmarks.

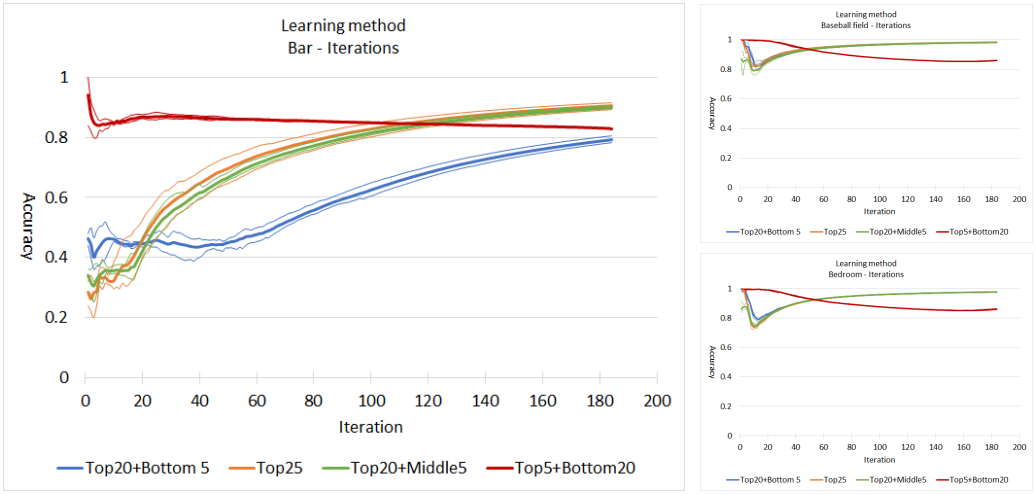


Figure 6.9: The accuracy that the different settings had classifying the search spaces for the three evaluation categories.

6.1.2 Limiting search space

This parameter benchmark is described in Section 5.3.2.3 and evaluates the four settings **All images**, **Threshold**, **Early stopping** and **Both rules**.

The metrics used in this evaluation is presented in Section 5.3.2. Recall that the effectiveness of the model during search iterations is also measured in terms of time taken and how many images that are processed each iteration in order to find the best set of images to present to the user.

The Performance of the model is measured on an evaluation set as well as over the entire search space.

6.1.2.1 Evaluation set

When measuring the performance on the evaluation set, none of the stopping conditions affected the learning rate in any direction. The evaluation set was classified in a similar manner by all the settings throughout the whole benchmark. Resulting in that all settings have about the same values on all metrics, which is for instance visible when inspecting the F1-measure in Figure 6.10. Since all settings followed the same pattern the presentation of recall rate, precision and accuracy is omitted in this section. However, the general trend of the measurements can be observed in what is referred to as Top20+Bottom5 in Section 6.1.1.1.

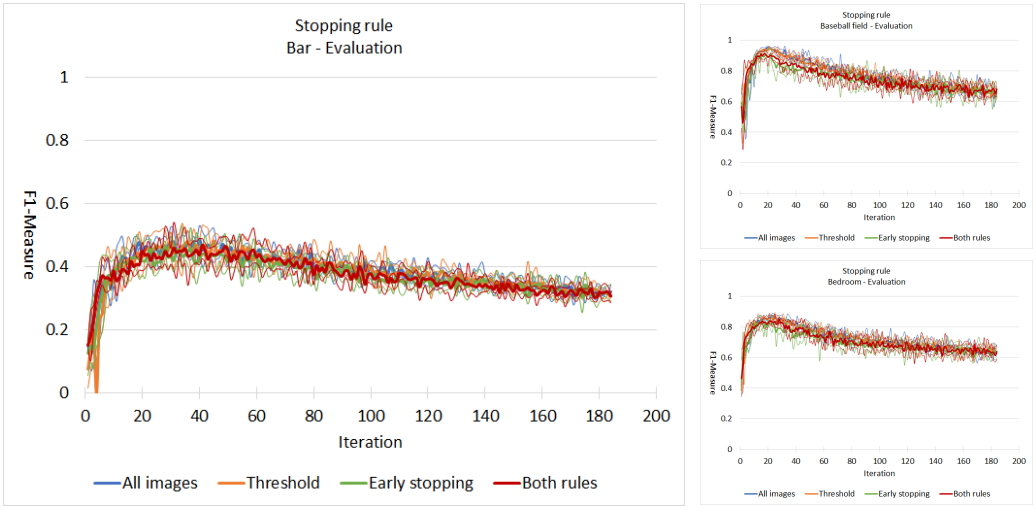


Figure 6.10: The harmonic mean of recall and precision, F1-measure, read on the evaluation set over iterations. All of the settings had performed similarly throughout the evaluation.

Since the concept was learned, independent of which of these stopping conditions that were used, the classification performance on the search space becomes more important as well as how much time that is spent calculating distances to data points each search iteration.

6.1.2.2 Search space

During the search iterations, the condition Early stopping introduced a notable trend compared to the condition Threshold in terms of correctness. As seen in Figure 6.11, the precision, when prediction the search space, using the stopping condition Early stopping gradually decreased, causing the precision of using both rules to decrease as well. Since the selection of images near the decision boundary on the negative side is delayed when using the early stopping condition, the material near the decision boundary is predicted correctly due to more knowledge. Thus causing the recall to stay higher than the other settings, as seen in Figure 6.12.

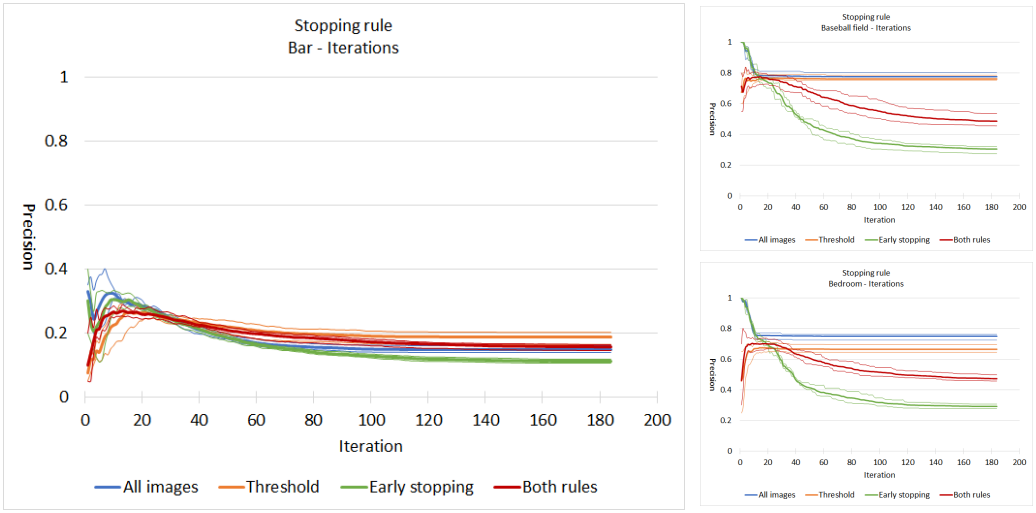


Figure 6.11: The precision of the different settings when classifying the search spaces of the three evaluation categories. The early stopping condition appear to have a negative impact on precision.

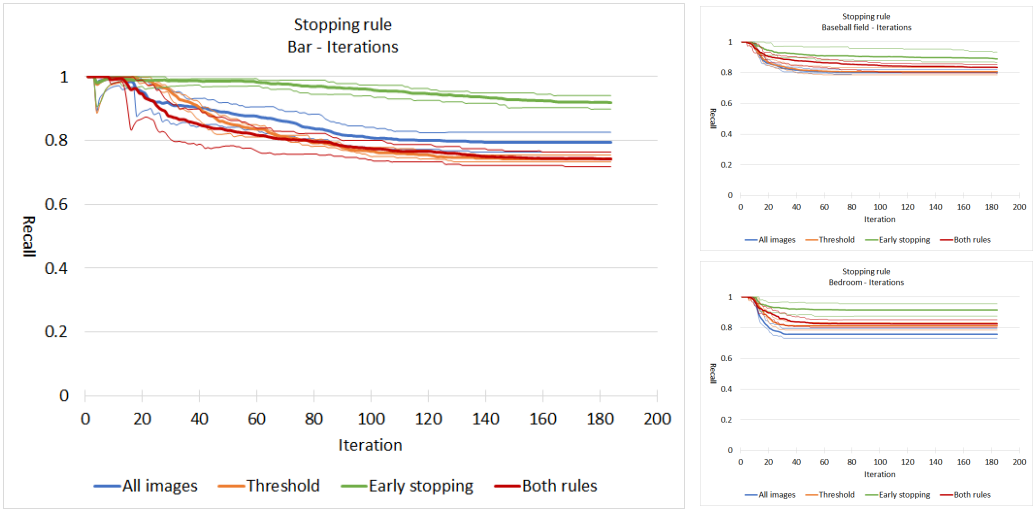


Figure 6.12: The recall rate of the different settings classifying the search spaces of the three evaluation categories. The condition Early stopping may have a positive effect on recall.

If no stopping condition is used, the total number of handled images – when the total search space is initially 4600 images – is (6.1)

$$\text{Handled images}_{max} = \sum_{i=1}^{4600/25} 25i = \sum_{i=1}^{184} 25i = 425500. \tag{6.1}$$

The total number of handled images for the different stopping conditions settings in relation to this number can be seen in Table 6.1. This indicates that the reduction of total handled images when combining the stopping conditions is $\approx 80 - 85\%$ on this dataset. Yet, the F1-measure levels were only reduced by 20-25% according to the result in Figure 6.13.

	Number of processed images; values are derived ratios of Equation (6.1)			
Desired category	All images	Threshold	Early stopping	Both
Bedroom	1	0.4168	0.2918	**0.1522**
Bar	1	0.2537	0.5717	**0.1862**
Baseball field	1	0.3843	0.2770	**0.1491**

Table 6.1: The average of total images that are handled during a search for each setting. The ratios of the value in Equation (6.1) are presented. The values are the average of five evaluations for each setting.

When inspecting the number of handled images by the different settings per iteration in Figure 6.14 it becomes clear how the reduction of handled images could vary so much between the classes when using each condition solely. In the evaluation of retrieving the category Bar the model finds more images that are predicted as relevant in a more dense manner. This is why the early stopping condition does not evaluate as true for most of the early iterations and why the threshold setting stops exploring earlier every iteration than in the other two evaluation categories. The two stopping conditions are so disjunct in terms of when their conditions are met, that the combination of the two actually reduce the number of handled images in the search for all three categories.

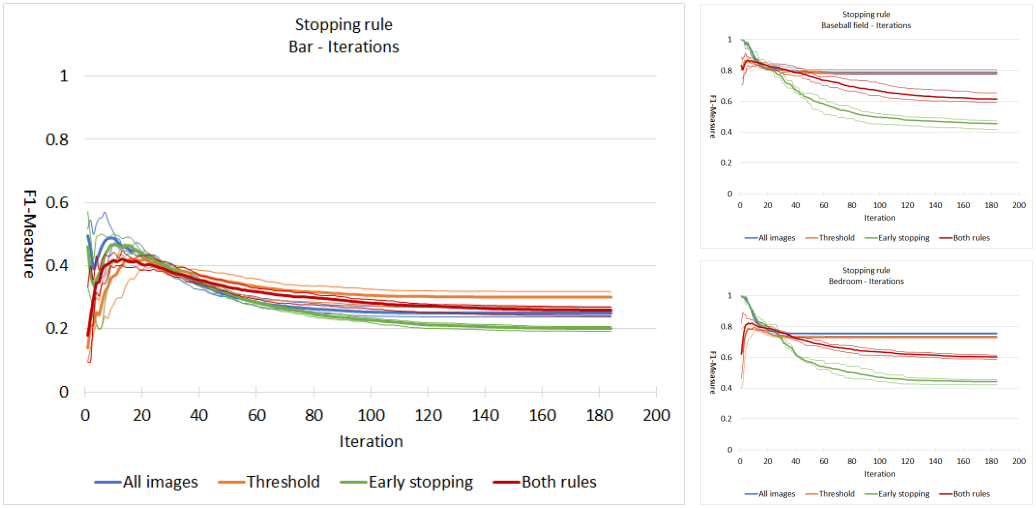


Figure 6.13: The F1-measure that the different settings had when classifying the search spaces of the three evaluation categories.

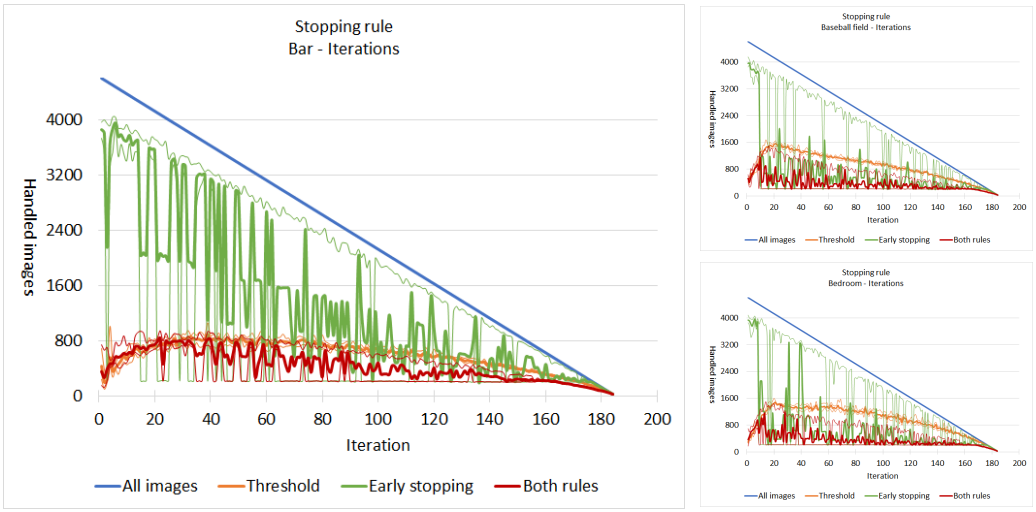


Figure 6.14: The number of handled images every iteration for the different settings. Having both stopping conditions results in processing fewer images in all three categories.

In terms of time taken there is a notable improvement when using a stopping condition instead of handling all the images in the search space. The time that an iteration takes varies a lot for the setting that only uses the condition Early stopping, see Figure 6.15, while the time taken for setting that only uses the threshold condition is just about the same throughout the entire search. Depending on which category that the algorithm is looking for, the total time spent searching with the two conditions seems to vary compared to each other and combining the two conditions results in less time spent in all evaluations according to the measurements in Figure 6.16. On average, the total time reduction when using both stopping conditions compared with classifying all images every iteration was $\approx 60\%$.

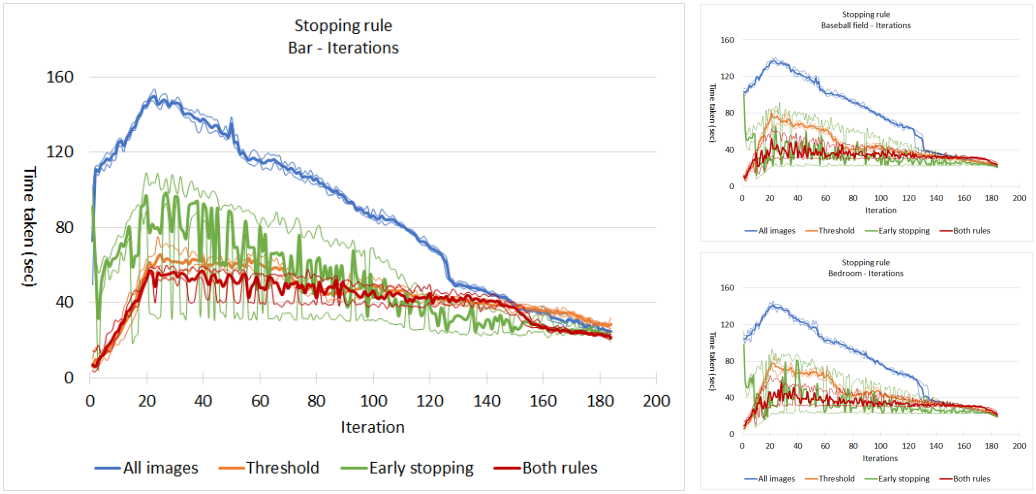


Figure 6.15: The time taken every iteration for the different settings. Having both stopping conditions results in the lowest peak in terms of time taken in all categories.

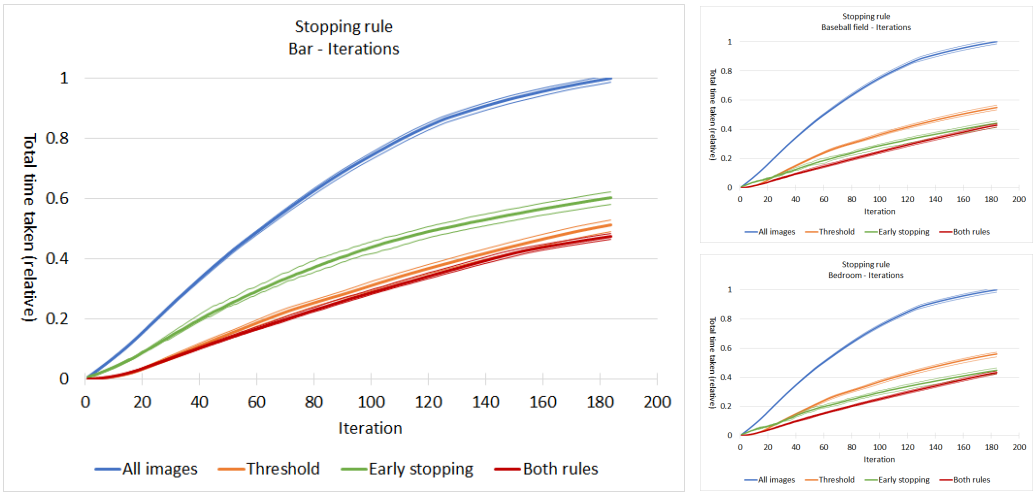


Figure 6.16: The total time taken for the different settings. There is a slight correlation with the ratios of processed images presented in Table 6.1 and the time taken ratios in these graphs.

In the same line as the F1-measure results, the number of retrieved images was a bit lower for the condition Early stopping (see Figure 6.17). When searching for the category Baseball field, the early stopping condition seem to delay the full retrieval of the category by about 80 iterations but when retrieving material from the category Bar the Early stopping setting follows the trend of the other ones and retrieves the relevant images at the same pace. This could be caused by the same reason that was observed in Figure 6.14; the setting Early stopping often samples material until no more unique images are found. It is however clear that the setting Both follows the setting Early stopping in terms of how many relevant images that are retrieved at a certain iteration.

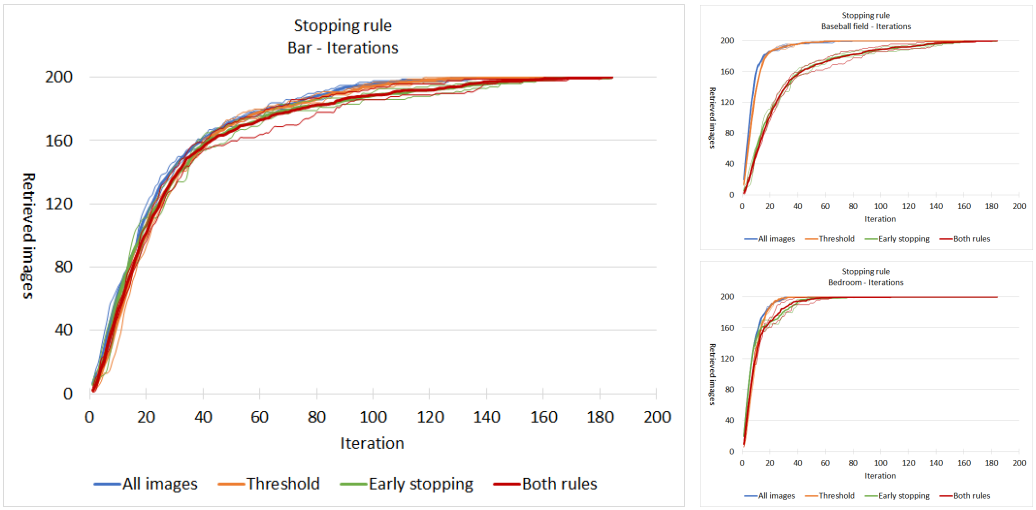


Figure 6.17: The number of retrieved relevant images over iterations for the different settings when classifying the three evaluation categories. The threshold setting retrieves images in the same rate as processing the entire search space each iteration. On the category Baseball field, the early stopping setting requires about 140 iterations to retrieve all 200 images.

6.1.3 Feature descriptors

This evaluation is described in Section 5.3.2.4 and a presentation of the evaluated settings can be found there. The settings described are **HOG**, **GCH**, **WT**, **CNN**, **Edge**, **All** and **All-CNN**.

The evaluation is performed with both of the stopping conditions that were evaluated in Section 6.1.2 and the material that is presented in each iteration is, as the evaluation in Section 6.1.1.2 suggests, a combination of the top-20 images and the bottom-5.

Just like the previous parameter benchmarks, the performance is measured over the entire search space as well as how a separate evaluation set is categorized.

6.1.3.1 Evaluation set

Throughout all the metrics that were measured during the evaluation, the settings that used CNN feature descriptors considerably outperformed the other settings. A trend that is distinguishable in both the F1-measure (Figure 6.18) and the accuracy (Figure 6.19) of the two settings that incorporate the CNN feature descriptor. There was a slight difference between using only the CNN feature vector as a descriptor and combining all the feature descriptors.

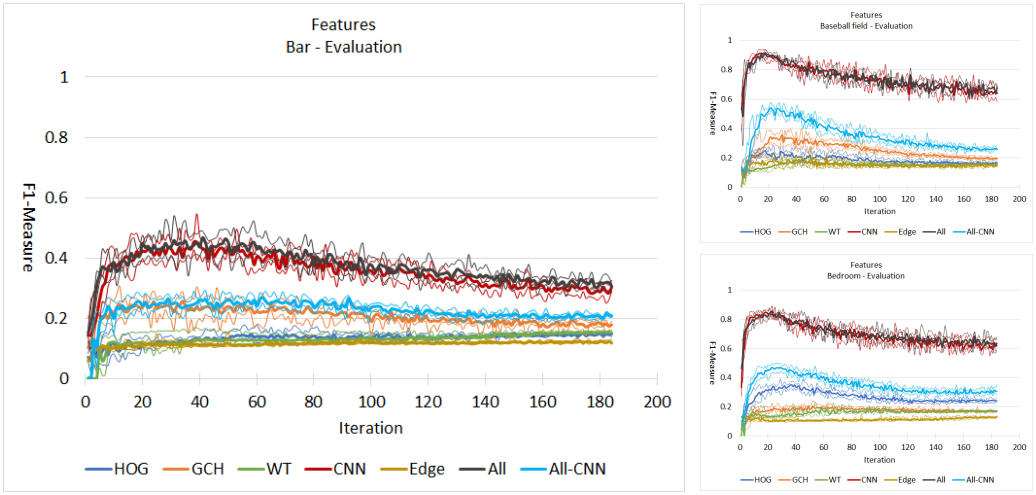


Figure 6.18: The F1-measure read on the evaluation set over iterations. Combining the information in different feature descriptors seems to give equally good or better results compared to the best single descriptor.

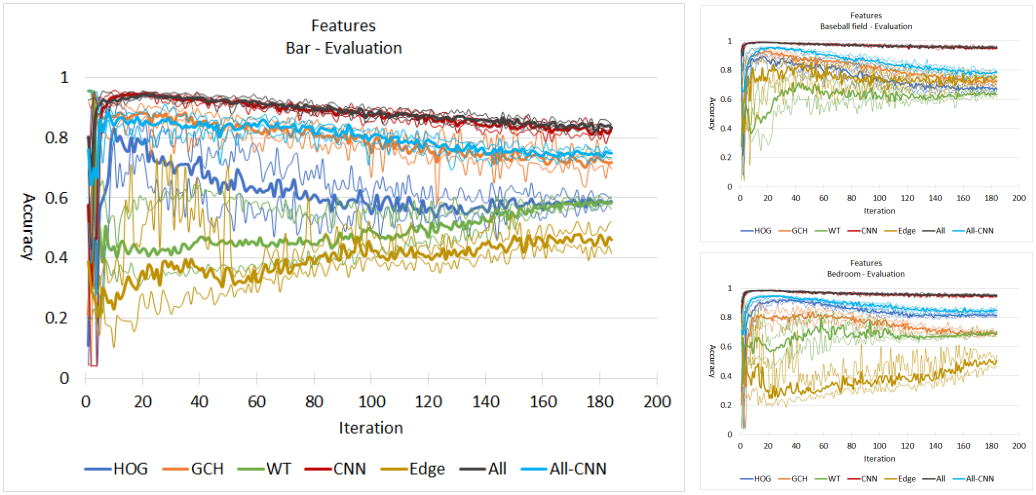


Figure 6.19: The accuracy of the different settings on the three evaluation sets. The CNN feature descriptor performs better as an information vector than the other descriptors.

The value of combining different feature descriptors become more clear when omitting the results of the fourth and sixth setting and only inspecting the remaining four feature descriptors and the combination of those. The F1-measure of the remaining five settings can be seen in Figure 6.20. Which single feature descriptor that performs the best depends on which category that the target is. For the category Bedroom a better result is given when using the HOG descriptor while for the other two categories the target concept is more distinguishable when using the GCH descriptor. More importantly; the combination of the four descriptors performs better or just as good as the best single descriptor.

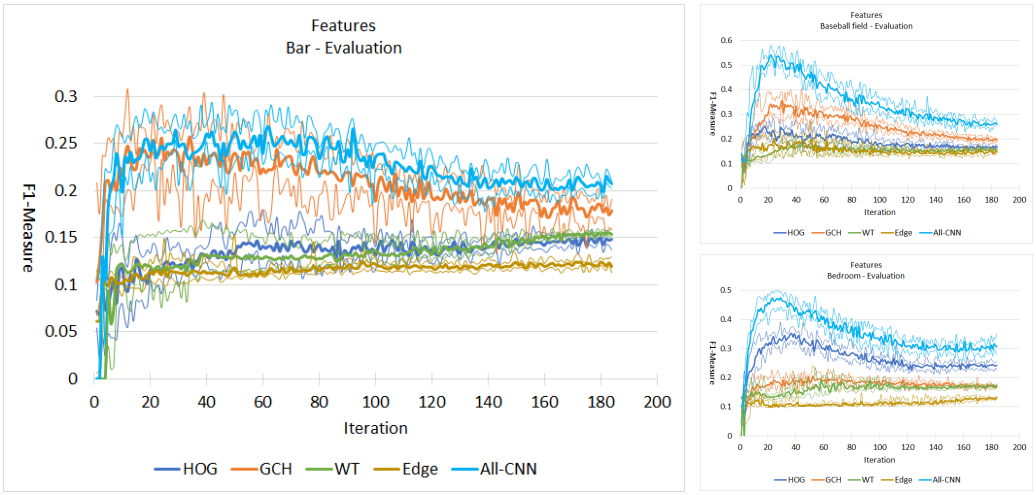


Figure 6.20: A closer look on the F1-measure of all the settings that are not handling feature vectors derived from a neural network. The combination of feature descriptors results in equally good or better results compared to the best single descriptor.

6.1.3.2 Search space

As mentioned in Section 6.1.3.1 the different target categories of the data sets are more distinguishable when using the CNN activation vector during classification. This is also evident in the F1-measure (Figure 6.21) and accuracy (Figure 6.22) when classifying the search space. More interestingly the positive effect of combining the first order classifiers become more clear in these measurements. When inspecting the category Bar in the two figures the setting All outperforms the setting that only uses the CNN activation vector as a feature descriptor.

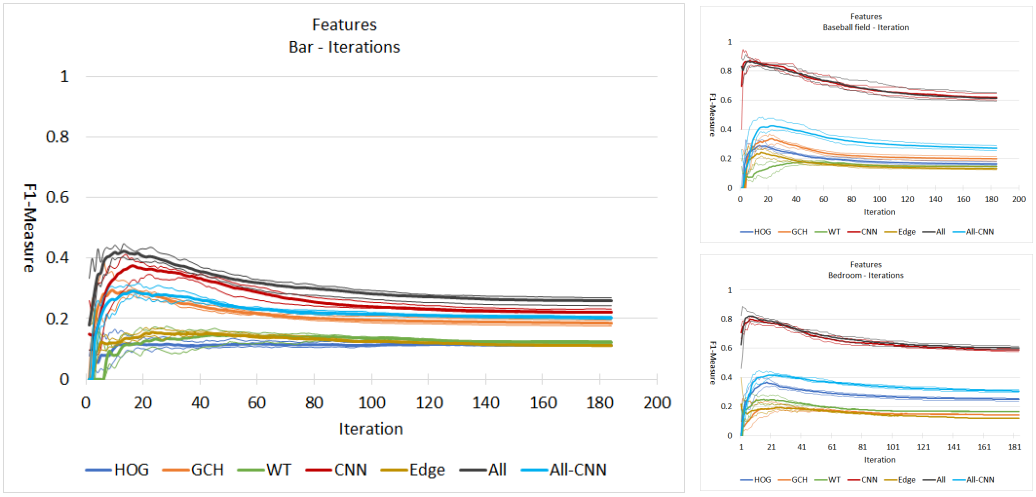


Figure 6.21: The F1-measure that the different settings had when classifying the search spaces of the three evaluation categories. The F1-measure on the category Bar is higher when combining all the feature descriptors compared with solely using the CNN feature descriptor.

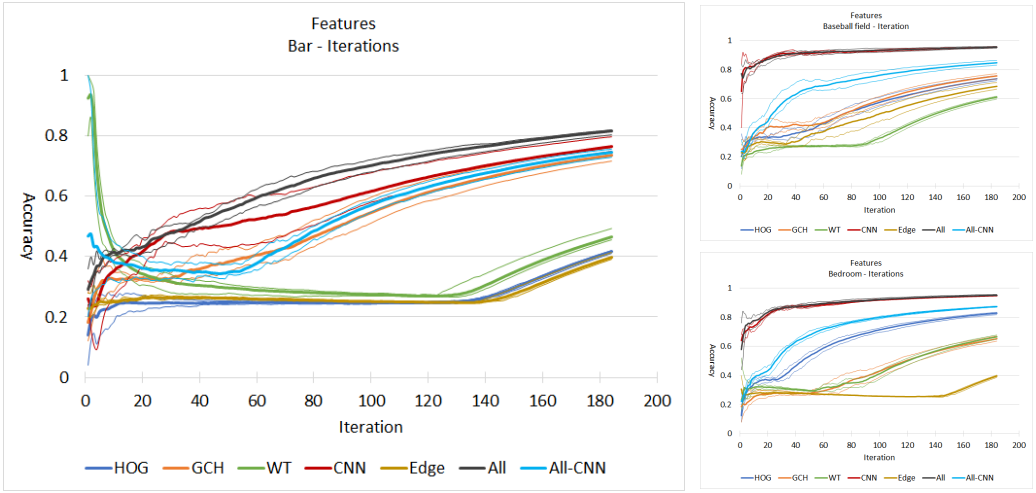


Figure 6.22: The accuracy of the different settings when classifying the search spaces of the three evaluation categories. As in Figure 6.21, there seems to be have a positive effect of combining feature descriptors.

All the different settings of feature descriptors do seem to be able to present some distinguishability between the different categories. With the exception of the initial rounds of searching for the category Baseball field with the setting WT, where the results are in line with selecting images at random. The number of retrieved images at a certain iteration is presented in Figure 6.23. Here the superiority of using the CNN activation vector becomes even more clear: When retrieving the bedroom class, the settings that use the CNN feature descriptor have retrieved the entire target category at iteration 80 while the other settings can not retrieve the entire class until the search space is depleted.

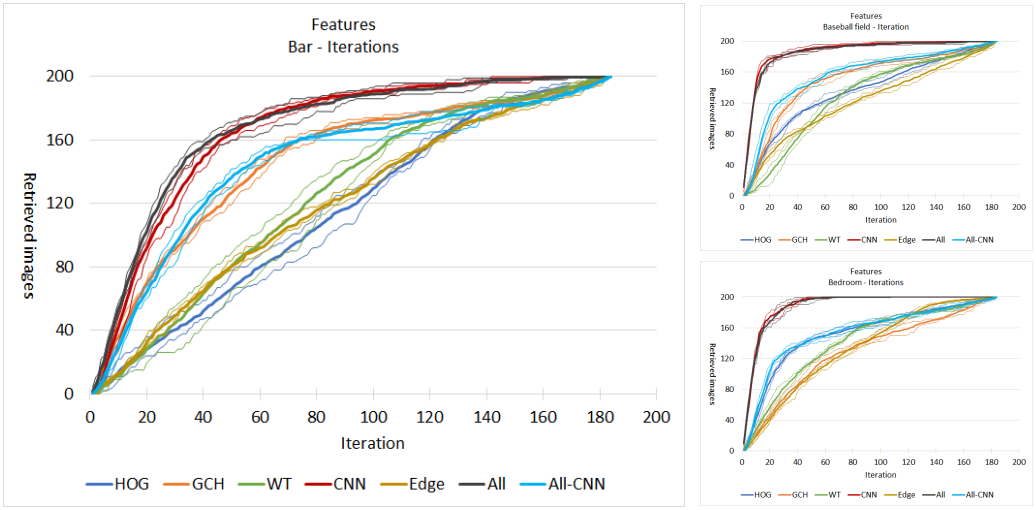


Figure 6.23: The number of retrieved images that the different settings had when classifying the search spaces. By introducing the CNN feature descriptors to the classifier the concepts of the categories seem to be easier to retrieve.

The results of the setting that uses all the feature descriptors are considerably high in terms of pace of retrieving images as well as with the F1-measure and the accuracy, but combining feature descriptors does come with the drawback of fitting more classifiers each iteration. Fitting more classifiers causes the total time taken to grow accordingly. The total time spent computing predictions depended on how many and which classifiers that were used and by using all six classifiers the time taken grew accordingly. In Figure 6.24 it is notable that combining all five descriptors compared to only combining four descriptors causes the computation time to grow to almost the double. Given the performance of the CNN setting one might consider only using this setting in the final evaluation, but as stated in Section 5.3.2 the classifier is used as intended throughout the parameter benchmarks with the exception of this evaluation.

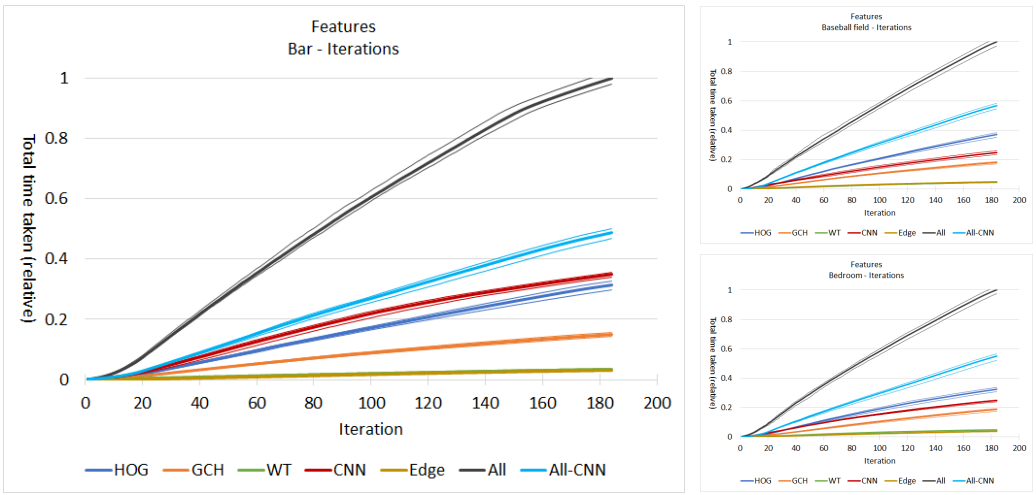


Figure 6.24: The total time taken for the different settings. The more feature descriptors that are used the more time it takes to train the classifying system.

6.1.4 Training data

The results of the evaluation described in Section 5.3.2.5 are presented here. The settings of the evaluation are listed in Table 6.2.

	Evaluation settings; Classifier and training data setup	
Setting	Training	Predefined training set *(relevant+irrelevant)*
1A	Only the first iteration	5+5
1B	Only the first iteration	5+50
1C	Only the first iteration	22+484
1D	Only the first iteration	250+250
2A	Every iteration	5+5
2B	Every iteration	5+50
2C	Every iteration	22+484
2D	Every iteration	250+250
3	Every iteration	0+0

Table 6.2: The different settings evaluated in this benchmark. Deeper explanation found in Section 5.3.2.5.

The metrics used for this evaluation are presented in Section 5.3.2 and are as mentioned performed on an evaluation set and the search space. The presentation of the metrics will only consist of the average values of the five runs for each setting due to the number of different settings in this evaluation. When the graphs included minimum and maximum values for each setting the data became much harder to understand and close to impossible to draw any conclusions from.

The model will during the evaluation use all of the stopping conditions described in Section 5.2.2.3, present the top-20 images and bottom-5 images in the end of every iteration as proposed in Section 6.1.2.2 and the classifier uses the full set of feature descriptors as described in Section 5.3.2.

6.1.4.1 Evaluation set

Naturally the performance of the settings that only use a predefined training set when classifying the evaluation set is same throughout all iterations. As expected the performance increases when having more data to begin with. In terms of F1-measure (seen in Figure 6.25) the performance of smaller training data sets are superseded by the performances of bigger training data sets.

The settings that use training set data retrieved by relevance feedback did however outperform the settings that did not. In the end of the search both the F1-measure and the accuracy where about the same for setting 3 as for the setting 1D, but at around iteration 25 in the settings that use relevance feedback (settings 2A-2D and 3) have performance peaks that are considerably higher than at the end of the search, which is noticeable in Figure 6.25 and Figure 6.26.

The biggest difference between the settings 2A-2D and the setting 3 was that during the initial three to six iterations. The setting with predefined data could actually perform a search while setting 3 could not. Not having any training data caused the performance of setting 3 to be considerably lower than the performance of the other

settings, but after the first couple of iterations the performance continuously rose up to be in level with the settings 2A, 2B, 2C & 2D. Even though setting 2D had a predefined data set of 250+250 relevant and non-relevant images their performance on the evaluation set turned out to be level. Yet it took the setting 2D a few more iterations to be on par with the settings 2A-2C and 3.

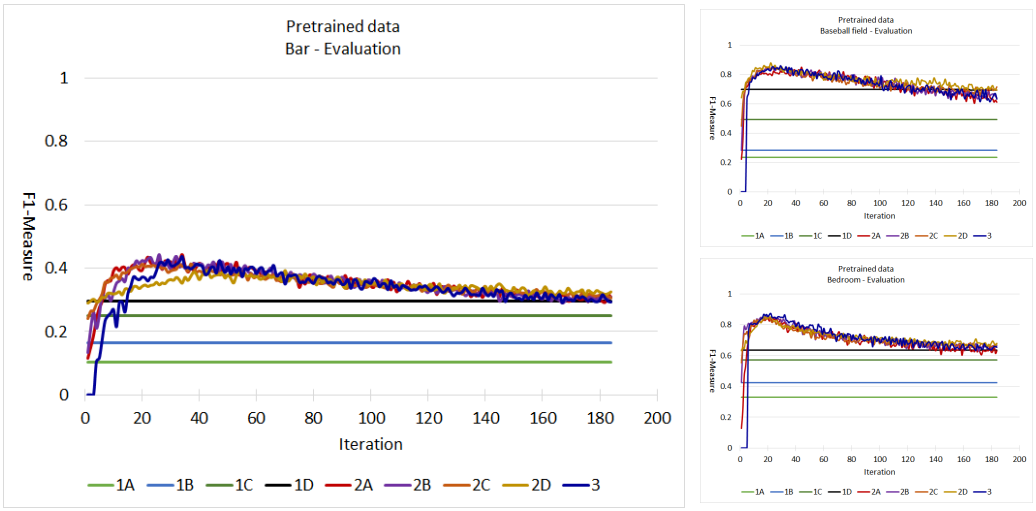


Figure 6.25: The F1-measure read on the evaluation set over iterations.

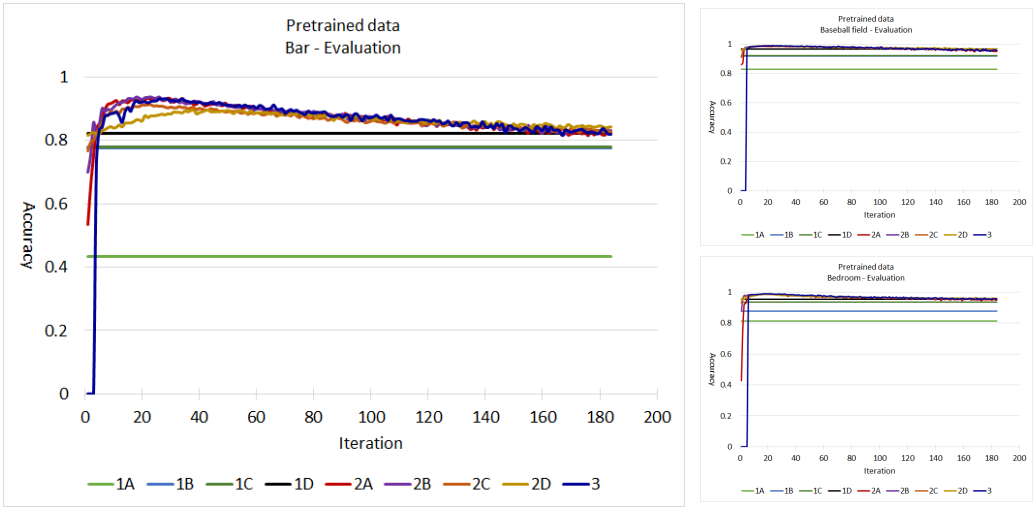


Figure 6.26: The accuracy on the evaluation sets. Using only training set data from relevance (setting 3) achieves higher results than a predefined training set, that has more relevant images than the search space (setting 1D) between iteration 10 and 80.

6.1.4.2 Search space

The results of measuring how the different settings classified the search space were not as indicating as the results that were observed when classifying the evaluation set. The metrics precision, recall, F1-measure and accuracy gave relatively inconclusive results. The values of the F1-measure of the evaluation (seen in Figure 6.27) do however imply some trends. In the first couple of iterations of the search the settings that use training set data extracted from relevance feedback begin to improve their results while the other settings begins with a high performance that shortly after decreases and then stabilizes.

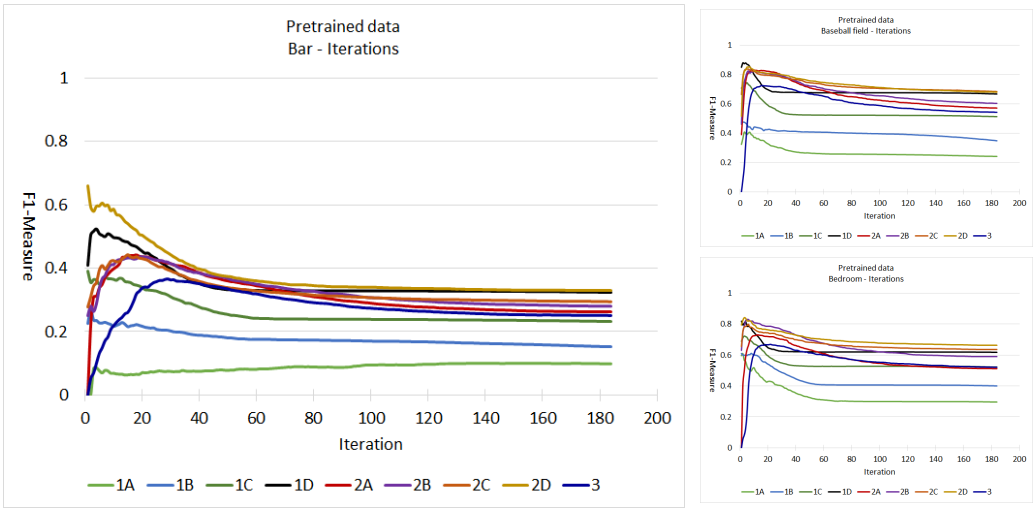


Figure 6.27: The F1-measure that the different settings had when classifying the search space. Having more data implies that the results become better.

In terms of how well the different settings perform these measurements do not imply anything else than the more training data one has the better the result gets. The number of retrieved images after a certain iteration, as visualized in Figure 6.28, reflects the different sizes of traning data they had. By comparing the results of setting 1D with setting 2D when searching for the category Bar in this figure, one can see how the data from relevance feedback improved the result of retrieving relevant material.

In the intended scenario, the search space is unlabeled and is therefore categorized while the data is presented. In this setting the time taken to predict the category of the images should be relative to how long it takes to correct poorly predicted categories, which is not covered by the scope of the thesis. The time taken for each of the settings to categorize the search space relative to each other is presented in Figure 6.29. The impact of refitting the classifiers each iteration truly becomes clear in this figure as well as how the size of the training set causes training the classifier to take longer.

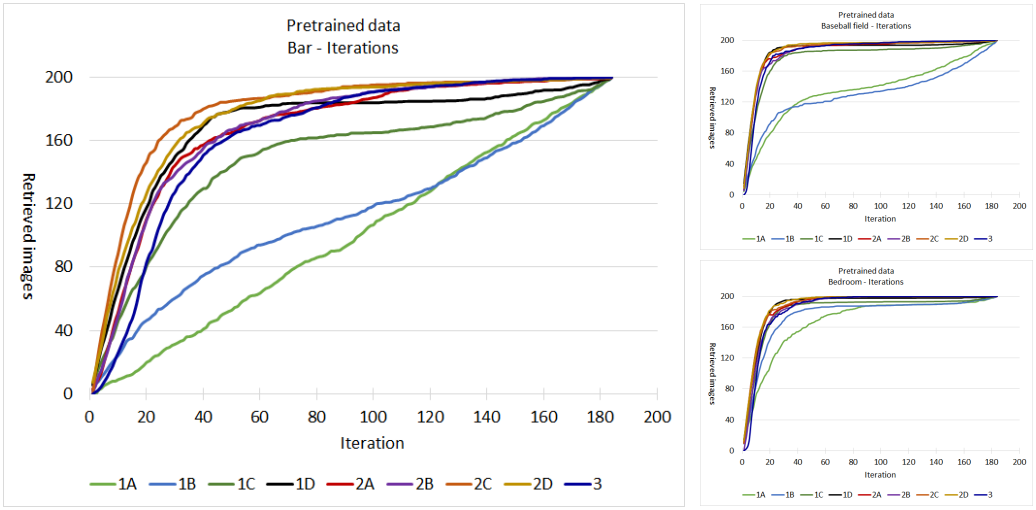


Figure 6.28: The number of retrieved imaged over the search iterations. Solely having 5 relevant images as training data (as in setting 1A and setting 1B) results in the incapability of retrieving the full content of the search space until it is depleted.

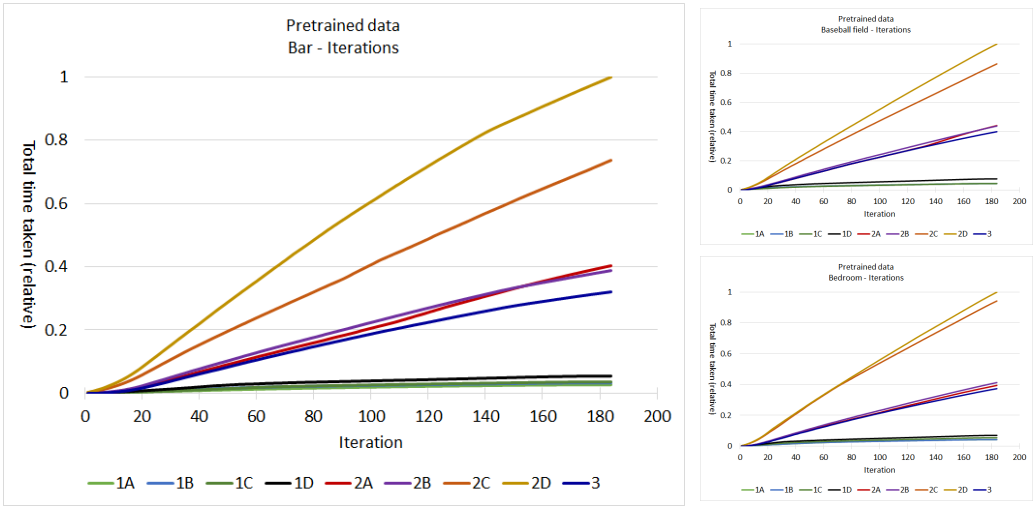


Figure 6.29: The total time taken for the different settings of this evaluation. There is a huge time difference between the training the classifier once and doing it every iteration.

6.2 Study comparisons

This section covers the results of the study comparison evaluations described in Section 5.3.3.

6.2.1 The Corel-1000 evaluation

The results presented in this section are retrieved in accordance with the evaluations described in Section 5.3.3.1. Recall that the evaluation is performed with $t = 10$ categories, $m = 500$ different query sets per category and number of retrieved images $n = 20$.

The proposed model is adjusted to randomly sample the query data from the search space and makes sure that the query set and the test set are disjunct before an evaluation is performed. As mentioned in Section 5.3.3 the number of images that the proposed model has in the training set was not decided on beforehand but was instead selected by inspecting the results of the similar studies. To do so the same evaluation was run with six different query set sizes: $2+2$, $3+3$, $4+4$, $5+5$, $7+7$ and $10+10$ relevant and irrelevant images.

The results of the evaluation on different sizes of query sets for the thesis model is presented in Table 6.3 as well as in Figure 6.30. When having the smallest number of query images and looking for the categories Africans and Beaches resulted in an average retrieval precision that is below 10%. Whereas retrieving the same categoriest when having the largest number of query images, the average retrieval precision is above 80%.

Category	Average precision (%); $n = 20$					
	$r + i$ (IICTVC query set)					
	$2+2$	$3+3$	$4+4$	$5+5$	$7+7$	$10+10$
Africans	05.66 ± 0.08	21.57 ± 0.31	48.03 ± 0.35	69.55 ± 0.49	84.59 ± 0.41	94.58 ± 0.06
Beaches	07.18 ± 0.12	14.63 ± 0.21	43.16 ± 0.41	48.62 ± 0.21	73.96 ± 0.26	81.57 ± 0.06
Buildings	11.53 ± 0.15	37.63 ± 0.70	68.10 ± 0.38	84.40 ± 0.25	95.77 ± 0.08	99.11 ± 0.01
Buses	61.64 ± 0.51	78.88 ± 0.65	97.65 ± 0.08	99.63 ± 0.01	100.0 ± 0.00	100.0 ± 0.00
Dinosaurs	99.06 ± 0.02	99.54 ± 0.02	100.0 ± 0.00	100.0 ± 0.00	100.0 ± 0.00	100.0 ± 0.00
Elephants	34.78 ± 0.49	63.56 ± 0.44	89.32 ± 0.17	93.45 ± 0.08	98.27 ± 0.01	99.24 ± 0.00
Flowers	49.79 ± 0.57	61.30 ± 0.64	96.54 ± 0.06	91.45 ± 0.12	99.66 ± 0.00	99.75 ± 0.00
Horses	63.02 ± 0.22	76.85 ± 0.57	92.50 ± 0.06	94.49 ± 0.05	97.73 ± 0.01	98.33 ± 0.01
Mountains	13.25 ± 0.15	41.41 ± 0.72	73.58 ± 0.69	87.83 ± 0.33	97.38 ± 0.02	99.02 ± 0.00
Food	10.01 ± 0.35	35.53 ± 0.42	68.73 ± 0.67	84.91 ± 0.17	93.97 ± 0.08	98.18 ± 0.01
Average	35.59 ± 0.01	53.09 ± 0.04	77.76 ± 0.02	85.43 ± 0.02	94.13 ± 0.01	96.98 ± 0.00

n - number of images retrieved.
r - number of relevant images in query set.
i - number of irrelevant images in query set.

Table 6.3: Precision when retrieving 20 images from the Corel-1000 set with different query sets. Note that the variance decreases as the query set increases from six images to more. IICTVC is short for the name of the thesis.

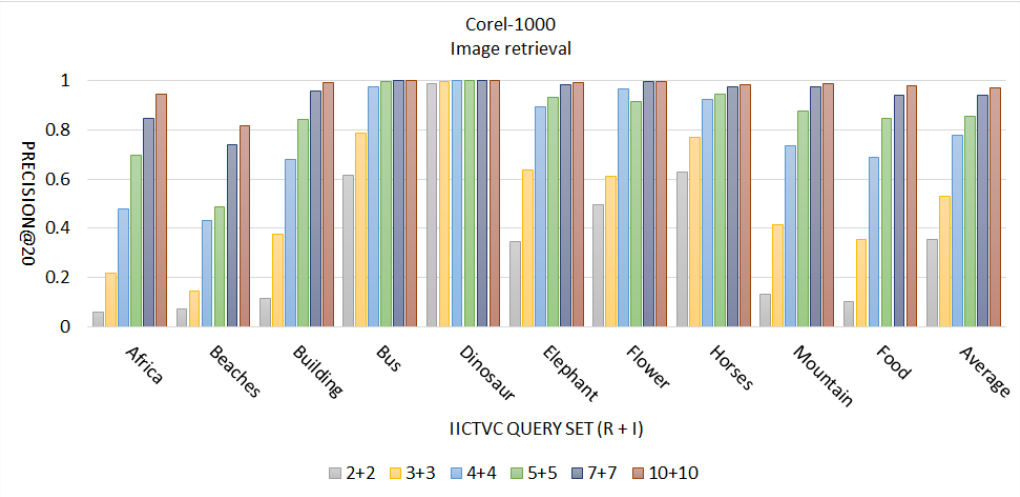


Figure 6.30: Precision when retrieving 20 images from the Corel-1000 set with different query sets presented in a graph. The query size affects the performance.

In both the Table 6.3 and the Figure 6.30 it is notable that the more query images the better the retrieval of the intended category becomes. This is somewhat in line with correlation that a separate image retrieval paper has found. The more query images that were used the better results were received until a certain point on their dataset; using six images as a seed for a performed search resulted in a detection rate of 60%, with ten images the rate turned out to be around 85% [54]. In the paper detection rate is described as the ratio of relevant images that are perceived by the model as relevant, a description that is very similar to how average retrieval precision is described in this evaluation and the values are somewhere near the result given by the test of different query sizes. Using 3 relevant and 3 irrelevant images as query set resulted in a precision of $\approx 53\%$ and using 5 relevant and 5 irrelevant images resulted in a precision of $\approx 85\%$. When selecting a query set of two relevant and two irrelevant images in a randomly generated manner, the model only finds the necessary categorical characteristics in some cases. This results in that the model retrieves zero relevant images in some iterations and sometimes the number of relevant images rises to being more than 15 or 20 images.

Given the results of the inspected papers, mentioned in Section 5.3.3.1, the query sizes that are used to compare with were set to $3+3$ and $5+5$ relevant and irrelevant images. The comparison of the precision@20 evaluations can be seen in Table 6.4 and Figure 6.31. When the query size of the proposed model is set to $3+3$ the model performs slightly better than the model described in [16], yet slightly worse than the models in [17, 18, 19]. When the query set of the proposed model is set to $5+5$ the performance of the proposed model supersedes the models of the other papers.

Category	Average precision (%); $n = 20$					
	Wang, James Z. et.al. [16]	Subrahmanyam, M et.al. [17]	Nagaraja, S et.al. [18]	ElAlami, M.A. [19]	IICTVC (3 + 3)	IICTVC (5 + 5)
Africans	47.50	69.75	56.00	**72.60**	21.57	69.55
Beaches	32.50	54.25	**60.00**	59.30	14.63	48.62
Buildings	33.00	63.95	58.00	58.70	37.63	**84.40**
Buses	36.30	89.65	94.00	89.10	78.88	**99.63**
Dinosaurs	98.10	98.70	98.00	99.30	99.54	**100.0**
Elephants	40.00	48.80	66.00	70.20	63.56	**93.45**
Flowers	40.20	92.30	88.00	**92.80**	61.30	91.45
Horses	71.90	89.45	78.00	85.60	76.85	**94.49**
Mountains	34.20	47.30	58.00	56.20	41.41	**87.83**
Food	34.00	70.90	54.00	77.20	35.53	**84.91**
Average	46.77	72.50	71.00	76.10	53.09	**85.43**

n - number of images retrieved.

Table 6.4: Precision when retrieving 20 images from the Corel-1000 the different studies and the proposed model with two different query set sizes.

Seeing that the proposed model performs this well compared to other CBIR methods is a good result. Yet it is still important to remember that having more query images reduces the risk of only having outliers as query data. The effect of this was noticed more often when the query set size was set to $2+2$ and $3+3$, but considerably less when the query sets were larger than that. The other studies handles all images in the same category as an equal influence which gives the proposed model an upper hand in this evaluation. The results are however still interesting since the proposed model is designed to have much larger training/query sets in order to function as intended.

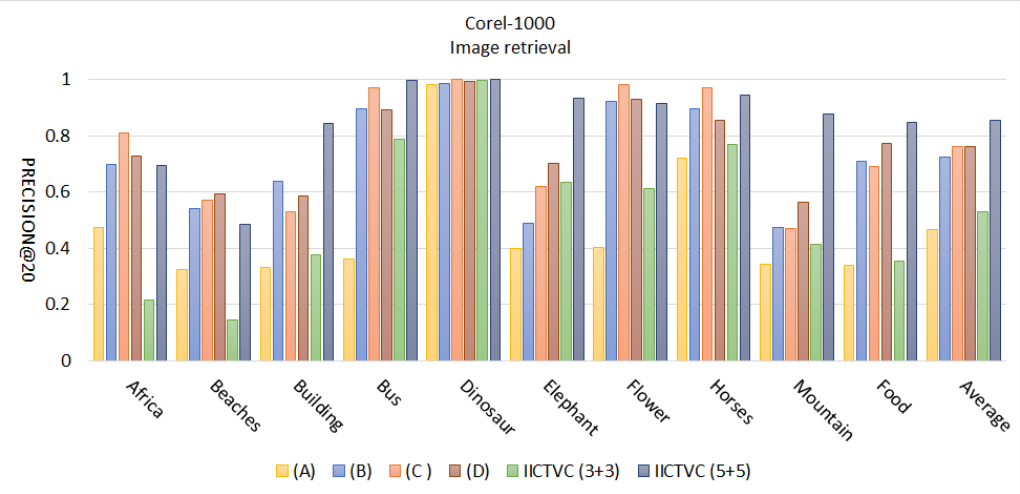


Figure 6.31: Precision when retrieving 20 images from the Corel-1000 the different studies and the proposed model with two different query set sizes had. **(A)**: Wang, James Z. et.al. [16], **(B)**: Subrahmanyam, M et.al. [17], **(C)**: Nagaraja, S et.al. [18] and **(D)**: ElAlami, M.A. [19].

7
Conclusion

The results of the evaluations that are presented in Chapter 6 were based on how the proposed model, described in Section 5.2, initially was designed. With this in mind the achieved results and measurements are discussed here. In this chapter the possible improvements of the proposed model are discussed as well as how knowledge extracted from the results can be used in other system designs.

7.1 Discussion of results

The performance of the model was best in the first half of a search during all parameter benchmark evaluations. This is noticeable when inspecting the learning method benchmark, Section 6.1.1.1. If the training set that is acquired at the turnpoint – when the model stops to predict images as relevant – is persisted, would the performance of the model on the entire search then become better? The difference would probably not be noticeable when inspecting the measurements on the search space since the model has already started to predict the rest of the search space as irrelevant images. This would however mean that training set makes the model perform well on the independent evaluation set and the rest of the search space can be predicted as non-relevant without taking up additional computational power. In every evaluation there has however always existed some outliers that were found after the turnpoint described earlier but in most cases became false negatives. Our iterative model could in other words not prevent this from happening. In the same section, Section 6.1.1, one setting had a lower count of false negatives; the setting called Bottom20+Top5. When evaluating the performance on the search space the recall for this method was higher than for the other ones. This would reduce the risk of any false negatives, but using this setting would result in not reaching the turnpoint mentioned earlier due to positive predictions until the end of the evaluation. Meaning that the matching module has to keep on adapting to the given data. On top of that the rate of retrieving relevant images that this setting will always be lower than for the other settings, which makes it less of an appropriate setting for image retrieval.

To search the entire search space, as the model in evaluation in Section 6.1.1 did, was never intended but since the focus of the thesis was to create a lightweight iterative model that can retrieve images, optimization became a second priority. Then again, the intention was to model something that reduced time spent per image so the two stopping conditions were applied. Since the combined pruning of the search space barely affected how the pace of how the matching module learned, more work could be done to reduce the time spent. To see what would happen to the performance if the classifier in the matching module would, as mentioned earlier, stop adapting its decision boundary to data after a certain point would be interesting. In Figure 6.10 this turnpoint is around iteration 20 and 40 depending on the category. The time

taken of the different settings in Figure 6.29 implies that the total computational time could be reduced by 75-80% by not adapting at some point.

By only using the activation vector of a CNN as a descriptor resulted in a time reduction of about 50% compared to when using the complete set of different feature descriptors as input to the deep SVM, yet the performance of the two settings were about the same. Due to the difference in performance of the two settings it would be interesting to replace the other feature descriptors with activation vectors from other neural networks that have been trained for other purposes. Combining the activation vectors might have the same effect as combining the features used in this thesis. The measurements presented in Section 6.1.3 do indicate that there are some improvements when using an ensemble of classifiers compared to only using its parts. Especially in Figure 6.20 where the performance was noticeably increased by using the feature descriptors in unison. A trend that would be interesting to evaluate

In the final parameter benchmark, the training set evaluation in Section 6.1.4, it became clear that if the refitting of the classifier was omitted the time taken was drastically reduced (see Figure 6.29). The time taken was reduced but the performance of the model never improved. When only using training data that is retrieved from relevance feedback, the model would perform equally well at the evaluation set as a model that also use a predefined training set with 250 relevant and 250 irrelevant images (see Figure 6.25). The data derived from the search space might be a subset actually represent the essence of that category, i.e. the model removes the outliers of a category from that subset. This could be used to not only fine-tune a training set intended for some specific image retrieval, but also to bring forth the material that represent semantic gap between human and machine learning in different datasets. The outliers that are not included in such a search defines what either the model could not see or what the person has put in a category that does not fit the description by mistake.

The productivity of the proposed model was also shown in the evaluation that was performed on the Corel-1000 set in Section 6.2. Even though the model is designed to improve the training set in an iterative manner, it behaved surprisingly well with small query sets. This procedure was never tested on the same dataset that the benchmarks performed on but doing so might give some interesting results. It was interesting to see that the proposed model could produce a similar precision as the neural network model proposed in [19] when having a query set of 5 relevant and 5 irrelevant images while the model described in that paper uses a training set of 900 images. However, the evaluation was not fair to all parts. The model, that uses an ANN to categorize material, uses a test set of 100 images while the test set used for the proposed model was substantially larger. Having a small query set allowed the deep SVM to fit immediately and classifying the a dataset is done rather quick compared to calculating individual distances. Just as having more than one image as a query allows the model to ignore outliers and can still find images that are defined as matches. Outliers that the papers [16], [17] and [18] have to include in their results. This evaluation will however give future studies the possibility to compare with a multi image query-based CBIR approach on the Corel-1000 set.

7.2 Future work

Some improvements have been mentioned previously in this thesis, but in this section all the viable future improvements are listed and discussed. The different ideas of improvements are split up into two parts; how the model can be improved and the possible usage areas outside the scope of the thesis.

7.2.1 Model improvements

The different model improvements included in this section are time investments that can be performed in a near future.

7.2.1.1 Scaling to bigger datasets

A future feature for the system would be to create a search algorithm that improves the way images are selected for each iteration based on previous predictions. As of now the procedure to select images is to randomly sample a predefined set size from the search space. Problems arise as the datasets become larger. Some form of sorting can minimize the time spent searching for the relevant images as well as improve the rate in which they are located. The algorithm can for example, either sort the material after how relevant it was predicted to be in previous iterations or in how the folder structure of the database is composed. This would thus reduce the size of the viable search space and allow larger parts of the unknown set to be evaluated faster as well as the material that is located in the same folders is ensured to be examined at some point.

7.2.1.2 Improvements to the relevance feedback loop

In the proposed model, the number of images predicted as relevant and irrelevant each iteration is set to 20 and 5 respectively. This is in no way based on previous studies or research and was set based on empirical testing by the authors. To optimize the performance of the algorithm in conjunction with the user, considerations should be made in how many images should be presented each iteration. The number of images presented might be decreased or increased, depending on result of studies, to enhance the performance and improve the user experience.

In the proposed model the relevance feedback module, described in Section 5.2.1, only updates the search space with which images that are relevant and which are not. This when knowing exactly which images that were mistakenly perceived as relevant and non-relevant by the matching module. If this information were to be used, it would be possible to "move" the decision boundary. One example is that it is possible to weight each data point in order to make them more impactful in how the classifier makes decisions. If an image is falsely categorized as a non-relevant one, it could be inserted into the dataset as two data points instead of one. If the

two data points are close to a decision boundary this would increase the cost of ending a training session at this point.

As seen in the results the performance of the model will decrease as the number of relevant images in the search space becomes exhausted. As this is a controlled environment this can not be proven for untested datasets. If presented to unknown data, that the algorithm does not find any relevant material after a couple of iterations it does not mean that relevant images are exhausted. There is however much to gain with cutting the retraining short as mentioned in Section 7.1. To stop the retraining and assume that all remaining images are non-relevant when the algorithm only finds non-relevant images seems to be the approach that is more time-efficient and maintains the performance of the model.

7.2.1.3 Selection of feature descriptors

In this thesis, not too much effort was put into selecting the best suited feature descriptors since the case is of the general nature and instead feature descriptors were chosen based on the idea that in order to create a more general classifier, more feature descriptors would be added. There are however feature descriptors that perform better or worse in these situations.

In the results of the feature descriptor benchmark, Section 6.1.3, the setting that uses the CNN activation vector is seen outperforming the settings that use on of the other feature descriptors by a large margin. Even if the margin is far more narrow when presented towards the category Bar. Instead of mixing weak and strong learners. One could for example combine the activations of the final fully connected layers of several neural networks that have been trained towards different datasets and use them as an ensemble. By doing so a wider base of strong learners is used.

In the proposed model, all classifiers are always present and their predictions are always taken into account. If there would be cases where some of these classifiers do not contribute or even might reduce performance by misclassifying, it might be beneficial to shut them down. During the parameter benchmark, in Section 6.1.3, some of the feature descriptors were found lacking in performance on certain sets. The improvement being to reduce time calculating that is of no use for the matching module by terminating certain parts of the classifier.

7.2.2 Miscellaneous usage areas

Besides from improving the proposed model by adding different features there are some possible usage areas outside of the domain of the thesis that were considered.

7.2.2.1 Dataset improvement

As mentioned in Section 7.1 the images that represent the essential bits of a concept are selected early on during a search space exploration and the outliers that are selected later on seem ruin classification results. This information could be used

to design a smaller training set specialized for a certain cause. In the scenario of a system that often uses a large dataset in order to solve some machine learning problem or an image retrieval problem, the feature previously described could be applied. To stop a search at the point where the classifier performs the best, as described in Section 7.2.1.2, could result in receiving a smaller dataset that can train a classifier faster for the intended purpose and the classifier maintains its performance.

Bibliography

[1] O. Isafiade and A. Bagula, *Data Mining Trends and Applications in Criminal Science and Investigations*, ser. A volume in the Advances in Data Mining and Database Management (ADMDM) Book Series. IGI Global, 2016. [Online]. Available: https://books.google.se/books?id=_QcDkAEACAAJ

[2] D. Rubino. The future of science as public service. [Online]. Available: https://www.dhs.gov/news/2011/03/14/future-science-public-service

[3] M. B. Kara Nance, Brian Hay, "Digital forensics: Defining a research agenda," in *Proceedings of the 42nd Hawaii International Conference on System Sciences*. IEEE, 2009.

[4] S. Garfinkel, "Digital forensics research: The next 10 years," in *The Digital Forensics Conference*, no. 7. Elsevier, 2010, pp. 64–73.

[5] M. K. Kundu, M. Chowdhury, and S. R. Bulò, "A graph-based relevance feedback mechanism in content-based image retrieval," *Knowledge-Based Systems*, vol. 73, pp. 254–264, 2015.

[6] A. W. Smeulders, M. Worring, S. Santini, A. Gupta, and J. Ramesh, "Content-based image retrieval at the end of the early years," *IEEE Transactions on Pattern analysis and machine intelligence*, vol. 22, no. 12, pp. 1349–1380, 2000.

[7] S. R. Athira Mohanan, "A survey on different relevance feedback techniques in content based image retrieval," *International Research Journal of Engineering and Technology*, vol. 04, no. 02, pp. 582–585, 2017.

[8] C. D. Manning, P. Raghavan, H. Schütze *et al.*, *Introduction to information retrieval*, vol. 1, no. 1.

[9] D. B. Judd, G. Wyszecki *et al.*, "Color in business, science, and industry," p. 388, 1975.

[10] A. Torralba, R. Fergus, and W. T. Freeman, "80 million tiny images: A large data set for nonparametric object and scene recognition," *IEEE transactions on pattern analysis and machine intelligence*, vol. 30, no. 11, pp. 1958–1970, 2008.

[11] A. Alzu'bi, A. Amira, and N. Ramzan, "Semantic content-based image retrieval: A comprehensive study," *Journal of Visual Communication and Image Representation*, vol. 32, pp. 20–54, 2015.

[12] H.-D. Cheng, X. Jiang, Y. Sun, and J. Wang, "Color image segmentation: advances and prospects," *Pattern recognition*, vol. 34, no. 12, pp. 2259–2281, 2001.

[13] S. Midha, R. Vijay, and S. Kumari, "Analysis of rgb and ycbcr color spaces using wavelet transform," in *Advance Computing Conference (IACC), 2014 IEEE International*. IEEE, 2014, pp. 1004–1007.

[14] Q. Su, Y. Huang, and J. Peng, "Coldimage: Contrast and luminance distribution for content-based image retrieval," in *Image Analysis and Signal Processing (IASP), 2011 International Conference on*. IEEE, 2011, pp. 143–146.

[15] N. Prajapati and A. K. Nandanwar, "Edge histogram descriptor, geometric moment and sobel edge detector combined features based object recognition and retrieval system."

[16] J. Z. Wang, J. Li, and G. Wiederhold, "Simplicity: Semantics-sensitive integrated matching for picture libraries," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 23, no. 9, pp. 947–963, 2001.

[17] M. Subrahmanyam, Q. J. Wu, R. Maheshwari, and R. Balasubramanian, "Modified color motif co-occurrence matrix for image indexing and retrieval," *Computers & Electrical Engineering*, vol. 39, no. 3, pp. 762–774, 2013.

[18] S. Nagaraja and C. Prabhakar, "Low-level features for image retrieval based on extraction of directional binary patterns and its oriented gradients histogram," *arXiv preprint arXiv:1503.03606*, 2015.

[19] M. E. ElAlami, "A new matching strategy for content based image retrieval system," *Applied Soft Computing*, vol. 14, pp. 407–418, 2014.

[20] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 1–9.

[21] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[22] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[23] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein *et al.*, "Imagenet large scale visual recognition challenge," *International Journal of Computer Vision*, vol. 115, no. 3, pp. 211–252, 2015.

[24] D. Tao. The corel database for content based image retrieval, dct-research. [Online]. Available: https://sites.google.com/site/dctresearch/Home/content-based-image-retrieval

[25] B. Zhou, A. Khosla, A. Lapedriza, A. Torralba, and A. Oliva, "Places: An image database for deep scene understanding," *arXiv preprint arXiv:1610.02055*, 2016.

[26] M. Hassaballah, A. A. Abdelmgeid, and H. A. Alshazly, "Image features detection, description and matching," in *Image Feature Detectors and Descriptors*. Springer, 2016, pp. 11–45.

[27] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, vol. 1. IEEE, 2005, pp. 886–893.

[28] S.-K. Pavani, D. Delgado, and A. F. Frangi, "Haar-like features with optimally weighted rectangles for rapid object detection," *Pattern Recognition*, vol. 43, no. 1, pp. 160–172, 2010.

[29] W. S. McCulloch and W. Pitts, "A logical calculus of the ideas immanent in nervous activity," *The bulletin of mathematical biophysics*, vol. 5, no. 4, pp. 115–133, 1943.

[30] M. Koskela and J. Laaksonen, "Convolutional network features for scene recognition," in *Proceedings of the 22nd ACM international conference on Multimedia*. ACM, 2014, pp. 1169–1172.

[31] J. Canny, "A computational approach to edge detection," *IEEE Transactions on pattern analysis and machine intelligence*, no. 6, pp. 679–698, 1986.

[32] R. Jain, R. Kasturi, and B. G. Schunck, *Machine vision*. McGraw-Hill New York, 1995, vol. 5.

[33] R. Maini and H. Aggarwal, "Study and comparison of various image edge detection techniques," *International journal of image processing (IJIP)*, vol. 3, no. 1, pp. 1–11, 2009.

[34] T. M. Mitchell, *Machine Learning*, 1st ed. New York, NY, USA: McGraw-Hill, Inc., 1997.

[35] D. H. Wolpert and W. G. Macready, "No free lunch theorems for optimization," *Transactions on evolutionary computation*, vol. 1, no. 1, pp. 67–82, 1997.

[36] S. B. Kotsiantis, I. Zaharakis, and P. Pintelas, "Supervised machine learning: A review of classification techniques," 2007.

[37] M. A. Kadam, K. S. Kadge, S. S. Mane, S. P. Naikwadi, and M. V. Kulloli, "Study and review of various image classification methods for diabetes mellitus detection," 2016.

[38] J.-P. Vert, "Kernel methods in genomics and computational biology," 2005.

[39] S. Tong and E. Chang, "Support vector machine active learning for image retrieval," in *Proceedings of the ninth ACM international conference on Multimedia*. ACM, 2001, pp. 107–118.

[40] P. Cunningham and J. Carney, "Diversity versus quality in classification ensembles based on feature selection," in *European Conference on Machine Learning*. Springer, 2000, pp. 109–116.

[41] A. Krogh, J. Vedelsby *et al.*, "Neural network ensembles, cross validation, and active learning," *Advances in neural information processing systems*, vol. 7, pp. 231–238, 1995.

[42] M. Cord and P. Cunningham, *Machine learning techniques for multimedia: case studies on organization and retrieval*. Springer Science & Business Media, 2008.

[43] B. Grofman, G. Owen, and S. L. Feld, "Thirteen theorems in search of the truth," *Theory and Decision*, vol. 15, no. 3, pp. 261–278, 1983.

[44] H.-C. Kim, S. Pang, H.-M. Je, D. Kim, and S. Y. Bang, "Constructing support vector machine ensemble," *Pattern recognition*, vol. 36, no. 12, pp. 2757–2767, 2003.

[45] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *science*, vol. 313, no. 5786, pp. 504–507, 2006.

[46] J. Chen, C. Wang, and R. Wang, "Using stacked generalization to combine svms in magnitude and shape feature spaces for classification of hyperspectral data," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 47, no. 7, pp. 2193–2205, 2009.

[47] X.-Y. Wang, H.-Y. Yang, Y.-W. Li, W.-Y. Li, and J.-W. Chen, "A new svm-based active feedback scheme for image retrieval," *Engineering Applications of Artificial Intelligence*, vol. 37, pp. 43–53, 2015.

[48] C. S. A. S. Mizanur Khondoker, Richard Dobson and D. Stahl, "A comparison of machine learning methods for classification using simulation with multiple real data examples from mental health studies," *International Research Journal of Engineering and Technology*, vol. 03, no. 01, pp. 814–820, 2016.

[49] S. S. M. S. P. N. M. V. C. K. Manaswi A. Kadam, Kiran S. Kadge, "Study and review of various image classification methods for diabetes mellitus detection," *Statistical Methods in Medical Research*, vol. 25, pp. 1804–1823, 2016.

[50] S. B. Kotsiantis, "Supervised machine learning: A review of classification techniques," *Informatica*, vol. 31, pp. 249–268, 2007.

[51] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[52] O. R. et al, "Imagenet large scale visual recognition challenge," *arXiv:1409.0575v3 [cs.CV]*, 2015.

[53] D. M. Powers, "Evaluation: from precision, recall and f-measure to roc, informedness, markedness and correlation," 2011.

[54] L.-J. Li and L. Fei-Fei, "Optimol: automatic online picture collection via incremental model learning," *International journal of computer vision*, vol. 88, no. 2, pp. 147–168, 2010.

A

Complete list of categories in the dataset MIT places205

Letter	Scenery name
A	Abbey, Airport terminal, Alley, Amphitheater, Amusement park, Aquarium, Aqueduct, Arch, Art gallery, Art studio, Assembly line, Attic, Auditorium, Apartment building
B	**Badlands, Ballroom, Bamboo forest, Banquet hall, Bar, Baseball field, Basement, Basilica, Bayou, Beauty salon, Bedroom, Boardwalk, Boat deck, Bookstore, Botanical garden, Bowling alley, Boxing ring, Bridge, Building facade, Bus interior, Butchers shop, Butte, Bakery**
C	Cafeteria, Campsite, Candy store, Canyon, Castle, Cemetery, Chalet, Classroom, Closet, Clothing store, Coast, Cockpit, Coffee shop, Conference center, Conference room, Construction site, Corn field, Corridor, Cottage garden, Courthouse, Courtyard, Creek, Crevasse, Crosswalk, Cathedral, Church
D, E	Dam, Dining room, Dock, Dorm room, Driveway, Desert (sand), Desert (vegetation), Dinette, Doorway, Engine room, Excavation
F	Fairway, Fire escape, Fire station, Food court, Forest path, Forest road, Formal garden, Fountain, Field (cultivated), Field (wild)
G, H	Galley, Game room, Garbage dump, Gas station, Gift shop, Golf course, Harbor, Herb garden, Highway, Home office, Hospital, Hospital room, Hot spring, Hotel room, Hotel (outdoor)
I, J, K, L	Ice cream parlor, Iceberg, Igloo, Islet, Ice skating rink, Inn, Jail cell, Kasbah, Kindergarden classroom, Kitchen, Kitchenette, Laundromat, Lighthouse, Living room, Lobby, Locker room
M, N	Mansion, Marsh, Martial arts gym, Mausoleum, Medina, Motel, Mountain, Mountain snowy, Music studio, Market, Monastery, Museum, Nursery
O, P, Q	Ocean, Office, Office building, Orchard, Pagoda, Palace, Pantry, Parking lot, Parlor, Pasture, Patio, Pavilion, Phone booth, Picnic area, Playground, Plaza, Pond, Pulpit
R	Racecourse, Raft, RailRoad track, Rainforest, Reception, Residential neighborhood, Restaurant, Restaurant kitchen, Restaurant patio, Rice paddy, River, Rock arch, Rope bridge, Ruin, Runway
S	Sandbar, Schoolhouse, Sea cliff, Shed, Shoe shop, Shopfront, Shower, Ski resort, Ski slope, Sky, Skyscraper, Slum, Snowfield, Staircase, Supermarket, Swamp, Stadium (baseball), Stadium (football), Stage, Subway station, Swimming pool
T	Television studio, Topiary garden, Tower, Train railway, Tree farm, Trench, Temple (east Asia), Temple/ (south Asia), Track, Train station
U, V, W, X, Y, Z	Underwater (coral reef), Valley, Vegetable garden, Veranda, Viaduct, Volcano, Waiting room, Water tower, Watering hole, Wheat field, Wind farm, Windmill, Yard

Table A.1: A complete list of all the categories in the set places205 by MIT. The sceneries in bolded text were used in the parameter benchmark.