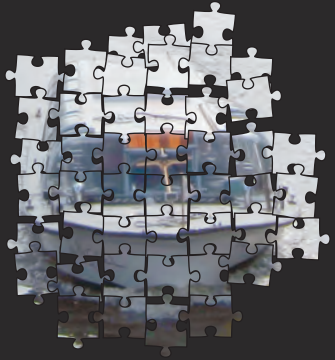
Robust visual scene categorization in context

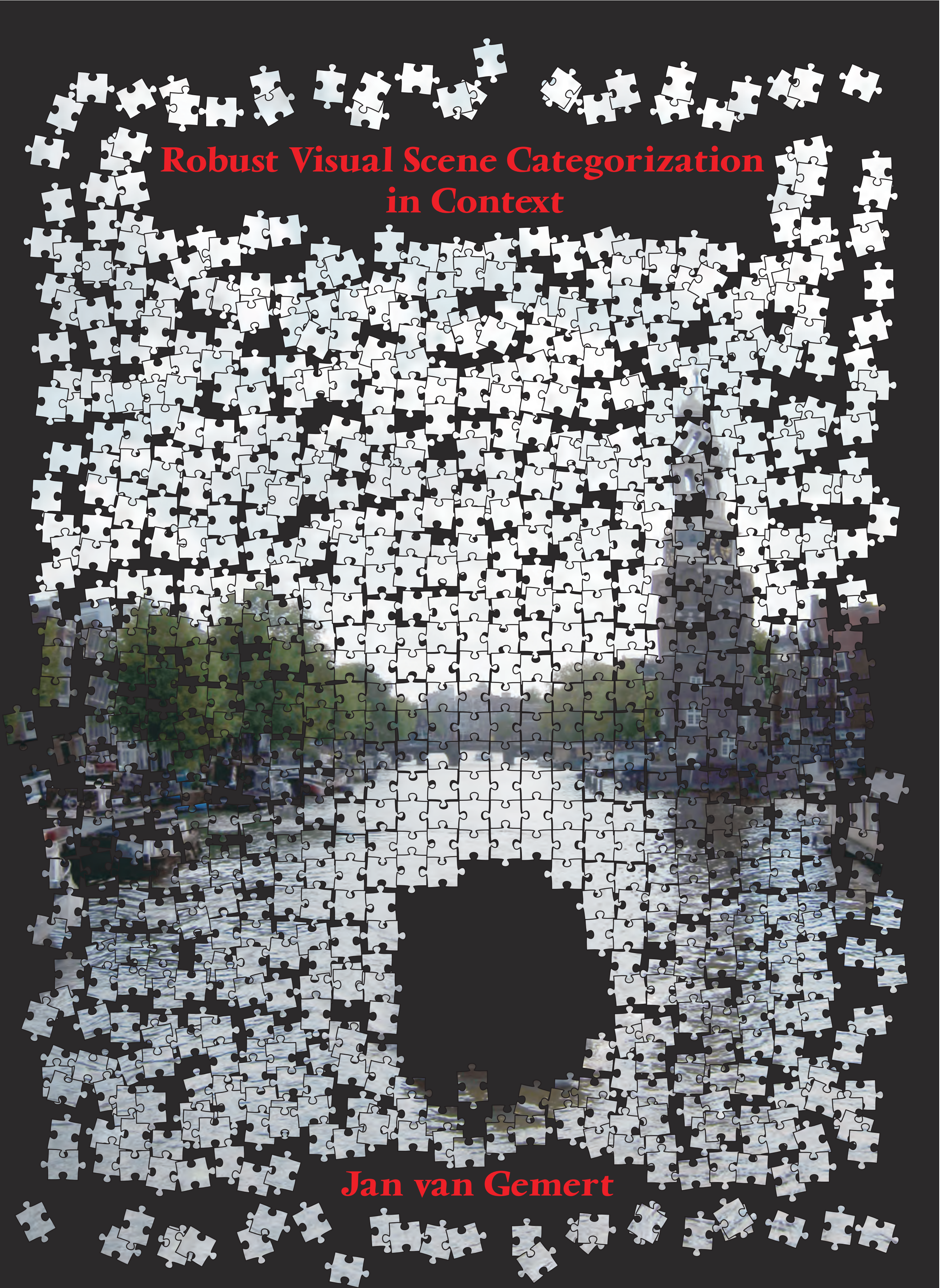
van Gemert, J.C.



Robust Visual Scene Categorization in Context

Jan van Gemert

Robust Visual Scene Categorization in Context



Jan van Gemert

Robust Visual Scene Categorization in Context

Jan C. van Gemert

This book was typeset by the author using LaTeX 2_ε. The cover was created by the author with python and matplotlib. The image represents local image features and the puzzle of what to put in the hole in the water. It is taken in front of the Montelbaanstoren next to the author's house in Amsterdam.

Printing: F&N Boekservice

ISBN 978907867592 1

Robust Visual Scene Categorization in Context

ACADEMISCH PROEFSCHRIFT

ter verkrijging van de graad van doctor
aan de Universiteit van Amsterdam,
op gezag van de Rector Magnificus prof. dr D. C. van den Boom
ten overstaan van een door het College voor Promoties ingestelde commissie,
in het openbaar te verdedigen in de Agnietenkapel
op vrijdag 24 september 2010 te 12.00 uur

door

Johannes Christianus van Gemert

geboren te Veghel

Promotiecommissie:

Promotor:	Prof. dr ir A. W. M. Smeulders
Co-promotor:	Dr J. M. Geusebroek
Overige leden:	Dr ir R. P. W. Duin
	Prof. dr Th. Gevers
	Prof. dr D. G. Lowe
	Prof. dr R. C. Veltkamp
Faculteit:	Natuurwetenschappen, Wiskunde en Informatica

Advanced School for Computing and Imaging

The work described in this thesis has been carried out within the graduate school ASCI, at the Intelligent Systems Lab Amsterdam of the University of Amsterdam.

Universiteit van Amsterdam

Contents

Chapter 1

Introduction

Prrobly, u have no trouble understnd this sntence. Its syntactical and grammatical errors are easily seen through. Similarly, humans see through noise, shadows, color variations, and depth/motion ambiguities to perceive the world. Just how noisy the perceived world is can be appreciated when trying to explain in detail 'how to see'. When it comes to explaining such processes in detail, there is no end to the incomprehension of a computer: a computer has to be told exactly what to do. This algorithmic nature is well-suited for situations where all possible steps are clear. In such formal settings, the computation speed of modern computers excels the power of the human brain, as exemplified by the victory over the human chess world champion in May 1997. When it comes to less formal situations, however, the human brain outclasses the computer. Consider for example figure 1, illustrating some results of a winning method for a computational object recognition task in 2008 [29]. Some of the unrecognized examples are so easy for a human that we lose sight of the complexities involved in visual recognition. Whilst the human brain is defeated at calculations, the trivial actions which we take for granted may be the hardest to perform for a machine.

In our increasingly digital world, automatic object and scene recognition is not merely an academic pursuit. YouTube has millions of videos online, and film and TV broadcasters have started to digitize their collections [118]. Hence, the ability to intelligently browse and query these large visual collections is of significant practical use. Automatic visual recognition techniques such as scene and object classification can benefit such tasks by providing the semantic handles to grasp these large pixel collections.

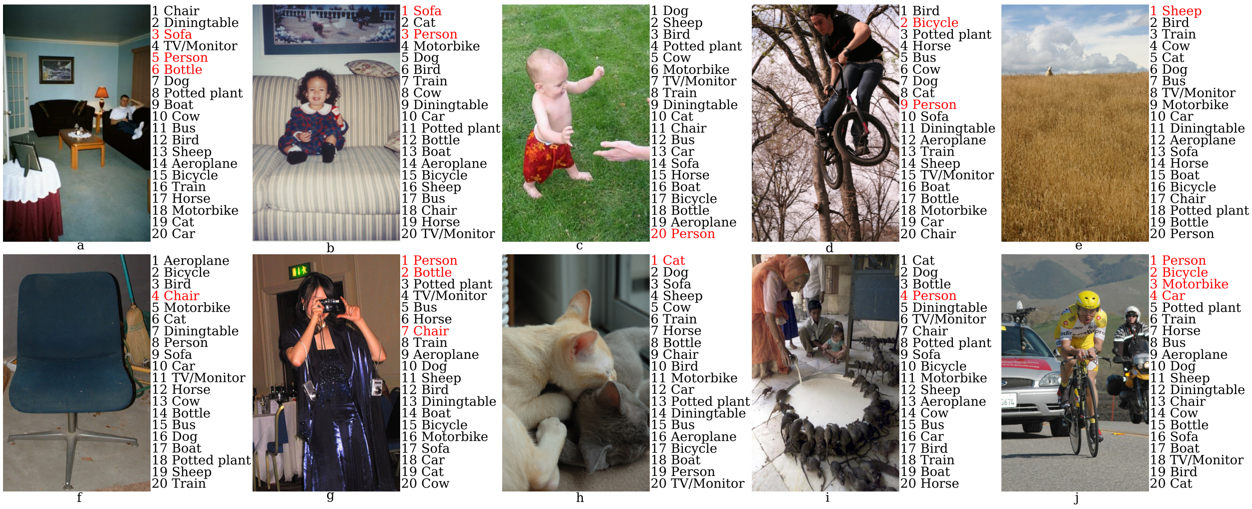


Figure 1.1: Ten images with results of a state of the art object classification method. Next to each image the 20 available objects are ranked by probability, with the correct object highlighted.

This thesis investigates the use of context to increase the robustness of automatic visual scene classification. Several types of context are investigated: the global contextual configuration of

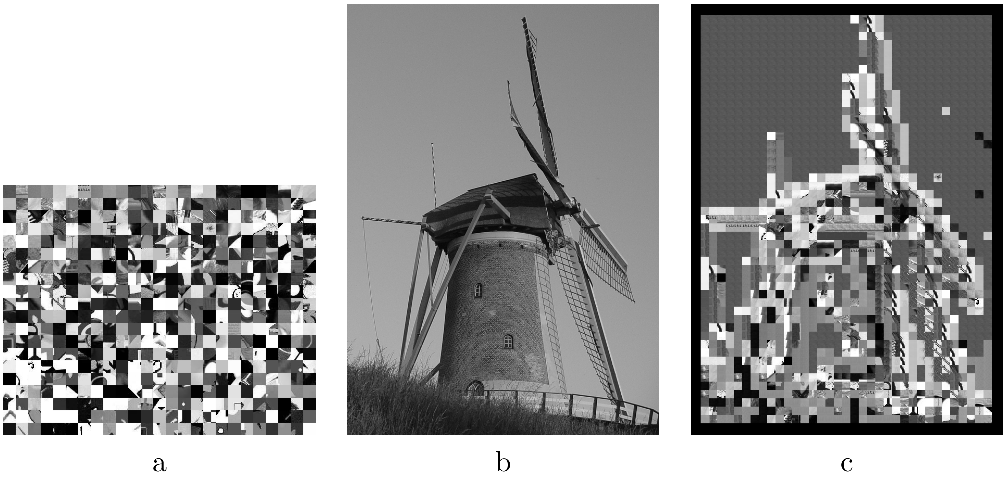


Figure 1.2: Example of the codebook model (a) Visual word vocabulary (b) Original image, (c) Codebook representation of the image.

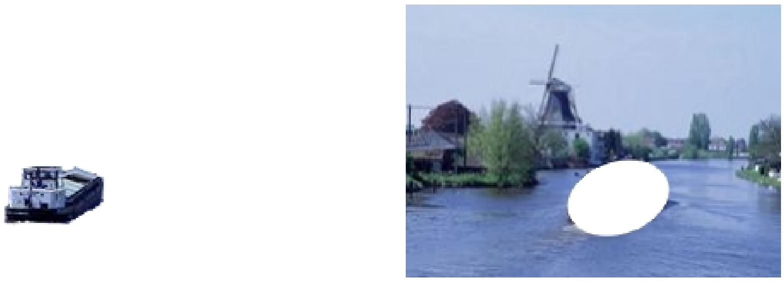
objects in an image, the local context of pixel representations and the narrative context of an image frame in a video sequence. Increased robustness is measured by improved classification performance on large image and video collections.

1.1 Robust Automatic Scene Recognition

A robust and well-known visual scene classification method is the bag-of-visual-words, or codebook, model. The codebook model describes an image as an unordered bag of discrete prototypical patches (visual words) selected from a predefined vocabulary [114, 146]. Each feature in an image is assigned its most similar visual word from the vocabulary under the assumption that a visual word is a prototypical representative of this image feature. Subsequently, the number of visual words in an image are counted, and a histogram of these visual word counts is input to a classifier. Figure 1.1 illustrates the codeword model. The model has, among others, the following robust traits. First, the unordered visual words in an image are translation invariant. Hence, the absolute positions of objects in an image are considered unimportant. Further, the model represents an image feature by a prototype, under the supposition that similar image features will be assigned to the same prototype. Thus, the model can deal with slight appearance variations. The third robust trait is the dimensionality reduction from numerous image features to a sparse set of visual word counts. Such a compact representation allows efficient storing and indexing of large image and video collections. These robust properties of the codebook model have made it the *de facto* standard for state of the art scene classification [29, 27], and therefore, is the model of choice in this thesis.

The visual words in the codebook model require some form of robust feature representation. Using raw pixel values for such a representation will incorporate unwanted variations unrelated to the content of the scene, as for example, camera rotations, camera distance, global intensity changes, shadows and shading effects. Such issues may be dealt with by a feature representation that is invariant to the unwanted variation. For example color invariants [44, 46], scale invariants [69] and affine transformation invariants [71, 80]. A particularly successful feature descriptor for the codebook model is SIFT [71] and its variations [6, 22]. The SIFT descriptor includes intensity invariants, and has recently been extended to include color invariants [18, 132]. Other invariant feature representations focus on shape [7], self-similarity [112], statistical distinctiveness [39] and color regularities [55]. When these invariant feature representations are used in the codebook model they lead to state-of-the-art results [29, 27, 132] and in the following chapters, we will use SIFT, as well as other invariant feature descriptors.

Apart from the image feature representation, there is the question of which image patches



a b

Figure 1.3: (a) Example of an object that is ambiguous without context, (b) The object in (a) as a hole in the water.

to use. The answer for scene recognition is different from the one for object recognition. For object recognition—where the same object is recorded under varying imaging conditions—it is beneficial to focus on distinctive patches such as corners [51], blobs [71], color points [133], stable patches [75] or statistically different patches [63]. These salient points or patches can be detected under varying imaging conditions [81]. Rather than focusing on interest points, the best approach for scene recognition is to use many to all available patches. For scene recognition—where the imaging conditions as well as the objects themselves may vary—a rich source of information is found in image context [4, 94]. The use of context is as old as Adam's fig leaf, and as illustrated in figure 1.3, an object's surroundings may be more revealing then the object itself. For scene recognition the use of context by densely sampling image patches is beneficial over interest point detection [93, 129] and thus adopted in this thesis.

The performance of the codebook model depends on the ability of the machine to learn to separate the scene categories. A popular and well-performing classifier is the Support Vector Machine (SVM) [17]. The SVM aims to maximize the classification margin between positive and negatively labeled training images. This maximum margin can subsequently be used to classify a new, unseen image. Besides the choice of the classifier, however, the final performance evaluation depends on the choice of performance metric. This choice depends on the application at hand. For a classification application [32, 49] the average classification rate is suitable. Alternatively, for a retrieval application [29, 116] a score dependent on a ranked list such as Area Under the ROC-Curve (AUC) or Average Precision (AP) is more relevant. These two measures differ in that AP emphasizes the beginning of a list more than does the AUC. In this thesis we will use and evaluate several classifiers and performance metrics.

1.2 Organization of the Thesis

This thesis is concerned with the theory and practicalities of visual word model. The theoretical intentions behind such a model may be clear, however, the road to hell . . . i.e. an unsuccessful practical application, is often paved with good intentions. Consider the simple idea of averaging the performance score on a rotating hold-out set to estimate scene classification performance in video. The practical application of this simple idea is studied in Chapter 2. We investigate the effect of the narrative context in video on classifier performance estimation and how the estimation itself affects the final classification performance. Furthermore, we introduce a new intermediate performance measure and experimentally evaluate our approach for two different classifiers on a large video collection.

In Chapter 3 we present a scene classification method by incorporating several robust elements in the codebook model. We explore alternatives to only using the histogram of visual word counts and also use a codebook vocabulary of semantically meaningful elements to express image context like vegetation, water, sky, etc. These meaningful elements are expressed by color invariant features that take advantage of natural image statistics for a compact representation. We evaluate on a large video set and five image sets, where we demonstrate robustness by training our method on one set while evaluating it on another set.

Efficiency by a compact representation is the subject of Chapter 4. In this chapter, we argue that indexing large image and video collections in practice benefits from a compact image representation, i.e., a compact visual word vocabulary. To this end, we compare various improvements to the codebook model under a compactness constraint. We experimentally compare the improvements with the standard codebook implementation for two classifiers on a large video collection.

In Chapter 5 we exploit context in feature space by soft-assignment of image features to visual words in the codebook model. We evaluate 4 types of feature assignment and investigate the effect of image feature dimensionality, the size of the visual word vocabulary and the size of the data set. We rigorously evaluate on five image sets.

An object recognition method is studied in Chapter 6. We use color invariant features with an entropy based similarity measure. We evaluate our method on a large image collection consists of 1,000 objects recorded under various imaging circumstances.

Chapter 2

Episode-Constrained Cross-Validation in Video Concept Retrieval[1]

2.1 Introduction

Machine learning techniques have proven to be a valuable addition to the repertoire of a multimedia researcher. Applications of machine learning techniques in multimedia are found in semantic video labeling [122], video shot detection [99], audio classification [72], scene recognition [135], sports analysis [24], and in many other areas. Moreover, multimedia researchers have contributed to specifically designed classifiers for multimedia analysis [38, 88].

Several machine learning techniques rely on accurate performance estimation [25]. The estimated performance may be used in finding the best parameters of a classification model and helps when deciding between different features. Thus, accurate performance estimation influences the quality of the machine learning method.

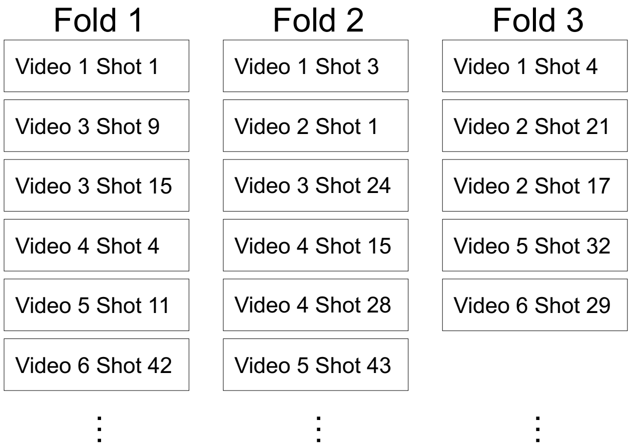
The central issue addressed in this Chapter is the following: *How is classification performance estimation affected by the narrative structure in multimedia data?* Much multimedia data is narrative in nature. For example, popular music has a verse and a chorus, multimedia presentations have slides designed with a message in mind, and shots in video data may be part of a storyline. Such narratives typically build a story by repeating similar elements. In separating narrative data in a test and training set, these highly similar elements may easily end up in both the test and the training set. Hence, commonly used classifier performance estimation techniques need special care when applied to multimedia classification.

In this Chapter we exploit the narrative structure present in multimedia data to achieve accurate classification performance estimation. We show that more accurate performance estimation increases the final classification performance. Furthermore, we investigate how unbiased performance indicators can be constructed, resulting in unbiased and accurate estimation of classification performance in a narrative. As an instantiation of narrative multimedia data we will focus on semantic concept detectors in video. However, the described techniques readily apply to other types of data that share a narrative structure.

The idea of exploiting the narrative structure in video is not novel [122, 50, 59, 151, 152], though using narrative units for unbiased classification performance estimation is novel to the best of our knowledge. Our earlier work [137] also noted the influence of narrative structure on classification performance estimation. This current Chapter, however, provides a more in-depth analysis of this earlier work while also presenting a new unbiased performance indicator for narrative data.

The organization of this Chapter is as follows. The next section revisits standard classifier evaluation techniques. Then, section 2.3 introduces an evaluation technique that respects narrative structure in video concept retrieval. This narrative structure introduces unbalanced data, which is discussed in section 2.4. Section 2.5 presents the experimental setup followed by the results in section 2.6 and the conclusions in section 2.7.

[1] Published in *IEEE Transactions on Multimedia* [139].



Fold 1	Fold 2	Fold 3
Video 1 Shot 1	Video 1 Shot 3	Video 1 Shot 4
Video 3 Shot 9	Video 2 Shot 1	Video 2 Shot 21
Video 3 Shot 15	Video 3 Shot 24	Video 2 Shot 17
Video 4 Shot 4	Video 4 Shot 15	Video 5 Shot 32
Video 5 Shot 11	Video 4 Shot 28	Video 6 Shot 29
Video 6 Shot 42	Video 5 Shot 43	
⋮	⋮	⋮

Figure 2.1: An example of partitioning a video set by using shot based 3-fold cross-validation.

2.2 Classifier Performance Evaluation

Correct classification error estimation not only provides a quantitative assessment of the classifier, it also influences classifier performance. Classifier performance depends on the quality of the classifier model, which in its turn relies on the input features and classifier parameters. These classifier parameters and features are typically tuned by maximizing the estimated performance over various input features and parameter settings. For example in a semantic video concept retrieval task, Snoek et al. [122] use the estimated classifier performance to select the best low level features. Furthermore, they find the best parameters for a Support Vector Machine (SVM) by maximizing the estimated classifier performance. In their framework, inaccurate classifier performance estimation might result in choosing the wrong features, or in sub-optimal parameter settings. Hence, classifier performance estimation affects the selected classifier model, and thus the quality of the tuned classifier.

Estimating classification performance is typically done by training a classifier on one set, and testing the classifier on an independent hold-out set. Thus, a straightforward approach to classifier performance estimation is keeping a random sample of the available data in an unseen hold-out set. This hold-out set should be as large as possible, to accurately represent the class variation that may be expected. However, keeping a large part of the data from the training set gives the classifier less data to train on. Hence, a balance between the size of the training set and the size of the hold-out set must be struck.

In contrast to a single hold-out set, the cross-validation method rotates the hold-out set over all available data. Cross-validation randomly splits the available data in X folds, where each of these X folds is once used as a hold-out set. The performance estimates on all rotating hold-out folds are averaged, yielding an estimate of the classifier performance. The cross-validation procedure may be repeated R times, to minimize the effect of the random partitioning. An example of cross-validation for a set of shots in a video is shown in figure 2.1. The advantage of using cross-validation is the combination of a large training set with several hold-out sets. Therefore, cross-validation is the standard procedure for classification performance estimation [25].

2.3 Cross-Validation in Video Classification

Machine learning is heavily used in semantic video indexing [88, 122]. The aim of semantic video indexing is retrieving all relevant shots in a dataset to a given semantic concept. Some examples of semantic concepts are *Airplane, Car, Computer Screen, Bill Clinton, Military Vehicle, Sports.* Machine learning techniques, and specifically classifiers, are commonly used to rank a list of shots according to their probability of being relevant to a semantic concept. These machine-indexed semantic concepts provide a user with automated tools to browse, explore, and find relevant shots in

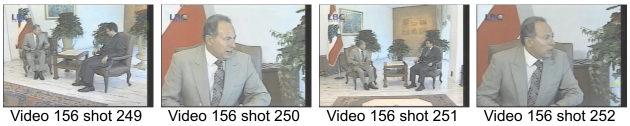


Figure 2.2: An example of narrative structure in video: four consecutive shots showing an interview with the former Lebanese President Mr. Lahoud.

a large collection of video. With growing digital video collections, there is a need for automatic concept detection systems, providing instant access to digital collections. Therefore, machine learning techniques are vital to automatic video indexing.

For semantic video concept indexing, a video is typically represented as a set of single shots [91, 122]. However, a video document is the end result of an authoring process [122], where shots are used to convey a message. For example, a topic in news video, may consist of several similar shots, as shown in figure 2.2. This temporal co-occurrence of similar shots in a topic may be exploited for video indexing [59, 151, 152]. Nevertheless, the video indexing task is oriented towards single shots, whereas a semantic concept might span several shots.

The granularity difference between the indexing task that focuses on single shots, and semantic concepts that may span several shots requires special care in estimating retrieval performance. Consider figure 2.2, and note the high similarity between shot 250 and shot 252. The similarity between these two shots can be expected, since they are part of the same narrative structure. However, the retrieval task focuses on single shots, and does not take this semantic relation between shots into account. Therefore, the common practice [91, 122] of estimating retrieval performance by cross-validation on shots is biased. Cross-validation on shots will mix shots in a single topic to different folds while randomly partitioning the data. Thus, shots that belong to the same semantic concept will be present both in the training set and in the rotating hold-out set. This leaking of near-identical information creates a dependency between the training set and the hold-out set, which will manifest in too optimistic estimates for retrieval performance. Moreover, if cross-validation is used for classifier parameter tuning, the parameters will be biased towards near-duplicate data and might consequently fail to find the best parameters for true independent hold-out data. Therefore, the narrative structure of video data should be taken into account when estimating retrieval performance.

In order to preserve the narrative relation between shots in a semantic concept, we propose an episode-constrained version of cross-validation. In contrast to a shot based partitioning of the video data, an episode-constrained partitioning aims to keep shots together if they are part of the same episode. In the context of a semantic concept retrieval task, an episode ideally consists of

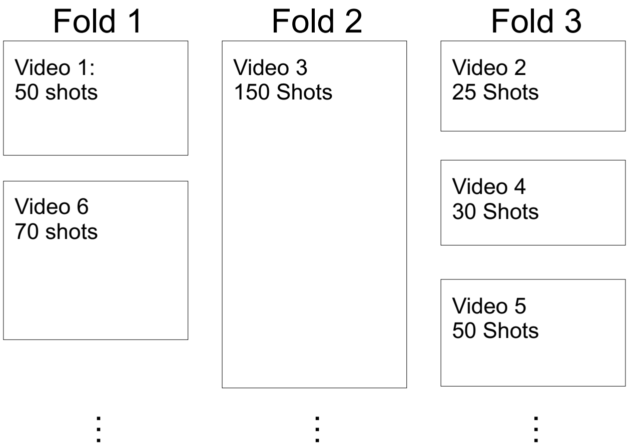


Figure 2.3: An example of a partitioning a video set by using episode-constrained 3-fold cross-validation.

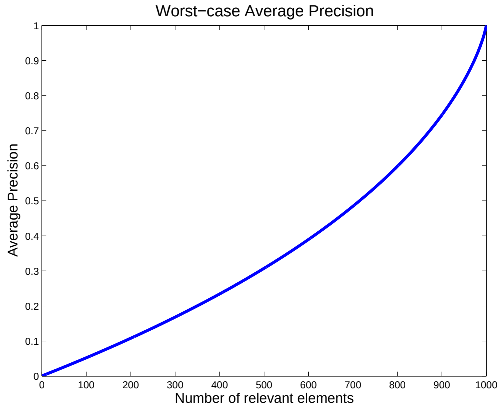


Figure 2.4: Average precision for a worst-case retrieved list of 1000 elements, where all relevant items are found at the bottom of the list. Note that the worst-case average precision score increases with the number of relevant items.

all constituent shots of the concept at hand. However, video story segmentation is an unsolved problem [50, 144]. Therefore, we resolve to using whole videos as atomic episodes. With videos as atomic elements, all shots in a video are kept together, preventing the leaking of near-identical information to the hold-out set. Whereas the traditional method randomly distributes shots, our method randomly distributes videos. An example of episode-constrained cross-validation for a video set is shown in figure 2.3. The episode-constrained version of cross-validation creates truly independent hold-out data, and will yield more accurate performance estimates of video concept classification.

2.4 Performance Estimation Between Unbalanced Sets

In semantic video retrieval, the performance measure of choice is average precision [91, 103, 122]. For a ranked list of elements, average precision denotes the area under the precision recall graph. Let $L_k = \{s_1, s_2, \dots, s_k\}$ be the top k ranked elements from the retrieved results set L, and let R denote the set of all relevant items, then average precision (AP) is defined as

$$\mathrm{AP}(L) \quad = \quad \frac{1}{|R|} \sum_{k=1}^{|L|} \frac{|L_k \cap R|}{k} I_R(s_k) \quad ,\text{for } |R| > 0, \tag{2.1}$$

where $|\cdot|$ denotes set cardinality and the indicator function $I_R(s_k) = 1$ if $s_k \in R$ and 0 otherwise. Average precision places a high emphasis on the top of the retrieved results list. The bottom of the results list is weighted less heavy and retrieval system benchmarks often truncate after a couple of thousand results. This practical approach to truncation and the high emphasis on the top retrieval results may explain the popularity of average precision in the video retrieval community.

Average precision describes the shape of the retrieved results list. However, average precision does not take the a-priori probability of relevant elements into account. Hence, average precision is not normalized for the number of relevant elements, and will give high scores when there are many relevant elements. Consider a worst-case retrieval system, that consistently places all relevant elements R at the bottom of the retrieved result list L. When the cardinality of L is fixed, $|L| = c$, the worst-case average precision (WAP) depends only on the number of relevant elements $|R|$, reducing equation 2.1 to

$$\mathrm{WAP}(|R|) \quad = \quad \frac{1}{|R|} \sum_{k=1}^{|R|} \frac{k}{(|L| - |R|) + k}, \text{for } |R| > 0. \tag{2.2}$$

Figure 2.4 illustrates the worst-case average precision for an increasing number of relevant elements. Note that a growing number of relevant elements results in an increasing a-priori average precision score. Thus, average precision scores are hard to compare between sets with a varying number of relevant elements because the average precision score is biased towards high-frequency relevant elements.

Given average precision as the performance measure for semantic video retrieval, it stands to reason to adopt average precision as the performance measure in cross-validation. In episode-constrained cross-validation, however, shots are kept together to prevent leaking of similar shots to a rotating test set. These atomic sets of shots hamper an equal distribution of the relevant shots over the cross-validation folds. For example, one news episode may contain several shots of a popular sports event, whereas other episodes may contain none. Hence, episode-constrained cross-validation yields an unbalanced distribution of relevant elements over the folds. Since the estimated performances on the folds are averaged to give a final cross-validation performance estimate, the folds that are randomly endowed with a high number of relevant-item episodes will dominate the cross-validation performance estimation. The effects of this will manifest itself in the classifier model selection that fit best to the fold that has the most relevant elements. Thus, in general, and for episode-constrained cross-validation in particular, an alternative to average precision is required that normalizes for unbalanced folds.

A performance measure for cross-validation should optimize average precision and allow equal weights when averaging cross-validation folds. Hence, this performance measure should scale between a fixed minimum and maximum, say 0 and 1, where 0 should represent the case where all relevant elements are retrieved at the bottom of the list, and 1 should indicate that all relevant elements are found at the top of the list. This normalization between 0 and 1 remedies the bias of average precision towards a high number of relevant elements. Besides normalization, the performance measure should guarantee that it optimizes the original average precision score. Any alternative to average precision as a performance measure should follow these criteria.

Several alternatives to average precision may be found in the literature. In classifier evaluation it is common to use receiver operating characteristic (ROC) curves for representing classification performance [25]. The ROC-curve shows the variation between the ratio of correctly classified positive elements and the incorrectly classified negative elements. As an alternative to average precision, the area under the ROC curve (AUC) may be maximized [38]. Maximization of the AUC optimizes the pairwise probability of retrieving a relevant element over a non-relevant element [21]. The AUC has the required property that an AUC value of 1 indicates perfect retrieval, and 0 denotes worst-case retrieval. However, optimizing the AUC does not guarantee to optimize average precision [23]. Other performance measures like R-precision [2], normalized average rank [86], normalized average precision [103], inferred average precision [154] or interpolated precision [102] may optimize average precision, however they do not scale between a fixed minimum and maximum. To the best of our knowledge, no performance measure exists that satisfies our demands. Hence, for a retrieved results set L, we propose an unbiased version of average precision which we name balanced average precision (BAP),

$$\mathrm{BAP}(L) \quad = \quad \frac{\mathrm{AP}(L) - \mathrm{WAP}(L)}{1 - \mathrm{WAP}(L)} \quad , \text{for WAP} < 1, \tag{2.3}$$

where AP and WAP refer to average precision and worst-case average precision in equations 2.1 and 2.2 respectively. The balanced average precision merely rescales the average precision where the worst possible result is set at 0, and the best possible results set at 1. Since balanced average precision is a monotone rescaling of average precision, optimizing this measure also optimizes the original average precision. Hence, balanced average precision allows a normalized comparison between sets with an unbalanced number of relevant elements, while maintaining all properties of average precision.

In figure 2.5 we show the relation between average precision (AP) and balanced average precision (BAP). The figure illustrates the BAP score corresponding to a given AP value for various ratios

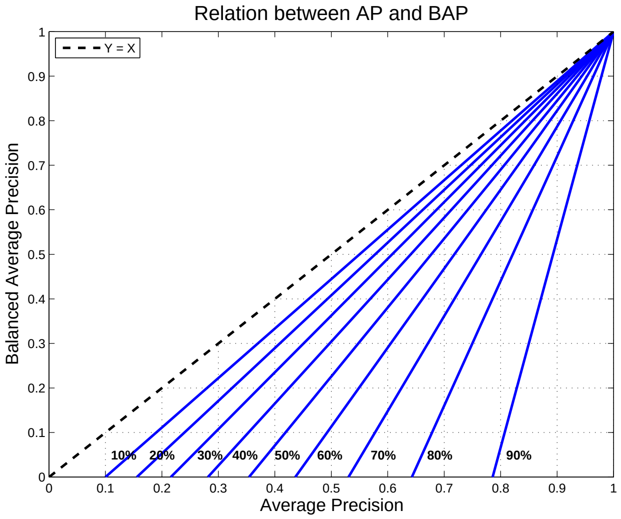


Figure 2.5: The relation between average precision and balanced average precision. The solid blue lines represent different ratios of the number of positive elements compared to the number of negative elements in a retrieved list. In this example, the ratios range from 10% to 90% positive elements. The dashed line $y = x$ is included as an AP self-reference.

between positive and negative elements in a list. For a fixed AP score, an increasing positive ratio yields a substantially smaller BAP score (vertical lines). Note that a larger difference between positive ratios yields a larger difference between BAP scores. For example, at an AP value of 0.8, the difference in BAP scores between the ratios of 80% and 90% is 0.35, whereas the difference between the ratios 70% and 80% is 0.13. For cross-validation, therefore, BAP will have more impact for folds with large differences between their positive elements ratios. What is more, the inequality between varying positive ratios increases, as the BAP score decreases (horizontal lines). For example, the difference between the ratio lines of 80% and 90% for a BAP value of 0.6 is 0.05, whereas this difference is 0.12 for a BAP score of 0.1. Hence, the effect of BAP becomes more pronounced for low classifier performance, i.e., with hard problems. We deem multimedia indexing a hard problem. Moreover, episode-constrained cross-validation increased the inequality between folds. Hence, we argue for using BAP for parameter estimation in multimedia classification.

2.5 Experimental Setup

We compare the episode-constrained version of cross-validation with the shot based version of cross-validation on a large corpus of news video: the Challenge Problem [121]. The Challenge Problem provides a benchmark framework for video indexing. The framework consists of visual features, text features, classifier models, a ground truth, and classification results for 101 semantic concepts[2] on 85 hours of international broadcast news data, from the TRECVID 2005/2006 benchmark [91]. The advantage of using the challenge framework is that the framework provides a standard set of features to the TRECVID data. Furthermore, the framework is well suited for our experiment, since there are a large number or shots, i.e. close to $45,000$, and an abundance of semantic concepts.

The Challenge data comes with a training set consisting of the first 70% of the video data, and a hold-out set containing the last 30% of the data. We use the training set for training both a k-nearest neighbor classifier (kNN) and a support vector machine classifier with an rbf-kernel [25]. We opted for the k-nearest neighbor classifier because of its simplicity, its generally decent performance, and the fact that it has a single tunable parameter. We included the SVM because it is a popular classifier which performs well on this data [121]. The features we use are the visual features [136]

[2] We did not evaluate the concept *baseball*, since all the examples in the training set of this concept are found in a single video.

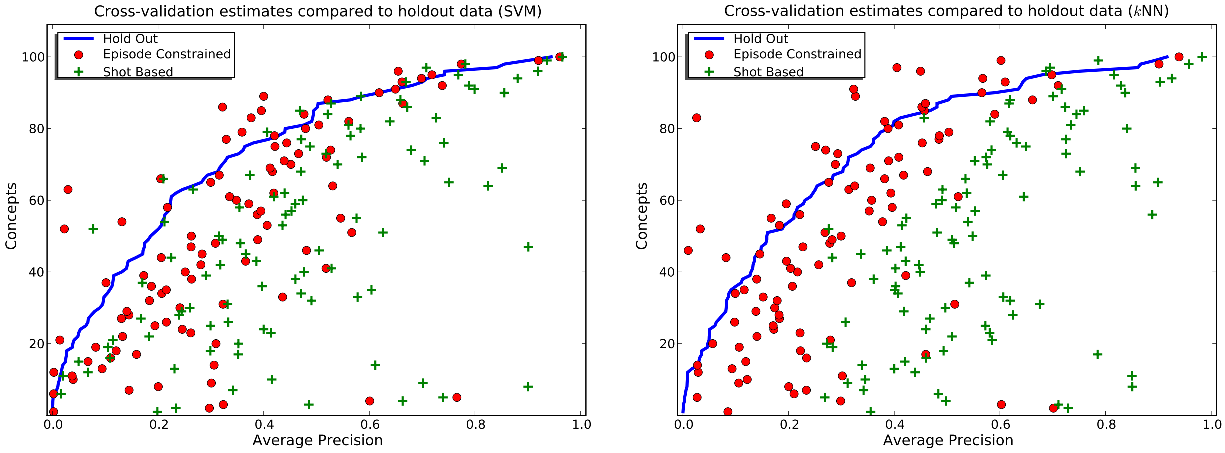


Figure 2.6: Performance estimates of episode-constrained and shot based cross-validation compared to the true hold-out performance (a) for the SVM classifier, and (b) for the kNN classifier.

that are provided with the Challenge framework.

2.6 Results

The focus of the experiment is on comparing episode-constrained cross-validation versus shot based cross-validation. To this end, we use both cross-validation methods to randomly partition the data in 10 folds. These 10 folds are subsequently used to estimate the best value for k for a kNN classifier, where $k \in \{1, 2, 3, 4, 5\}$. For the SVM classifier we preset the slack parameter C per class to the inverse of the class frequency and logarithmically tune the rbf-kernel size γ, where $\gamma \in \{1, 3.16, 10, 31.6, 100\}$. To evaluate the results, we computed the classification scores for all k and γ parameters on the hold-out set. The estimates and true hold-out average precision scores for the of the SVM and kNN classifier are displayed in figure 2.6.

2.6.1 Evaluating Episode Constrained Cross-Validation

The results in figure 2.6 clearly show the over-estimation of the average precision scores by the shot based cross-validation method. This over-estimation is more evident for the kNN classifier than for the SVM classifier. For the SVM classifier the episode-constrained estimation is closer to the true hold-out performance for 81 concepts, and in case of the kNN classifier this holds for 93 concepts. We show a more detailed figure for the kNN classifier in figure 2.7a. In this figure we show concepts with a large difference between their scores on hold-out or between the scores of the two cross-validation methods. Furthermore, we show the 7 concepts where shot based cross-validation gives a closer estimate to the hold-out performance than episode-constrained cross-validation. These 7 concepts either have very few examples or consist of shots that have near-copies in the hold-out set. Concepts with few examples (i.e. *Prisoner*) cannot be distributed completely over the 10 folds when videos are kept together. These concepts yield zero-scores for some folds, which in turn leads to a low fold average. The near copies in the hold-out set are due to commercials (*Bird, Fish, Cycling, Waterfall*) or due to little appearance variation (*Soccer, Nightvision*). Other concepts score significantly lower on the hold-out set because they have too little appearance overlap between the examples in the train set and the hold-out set (River, Motorbike, Mr. Nasrallah, Horse racing, Horse). The remaining concepts (*Cartoon, Drawing/Cartoon, Drawing*) are made up of highly repetitive shots within a video and therefore benefit most from episode-constrained cross-validation as can be seen by its accurate performance estimation compared to shot based cross-validation.

In figure 2.7a we show the estimated classifier parameters and the best parameters on the hold-out set. For space considerations we only show the kNN classifier, since it gives the best results.

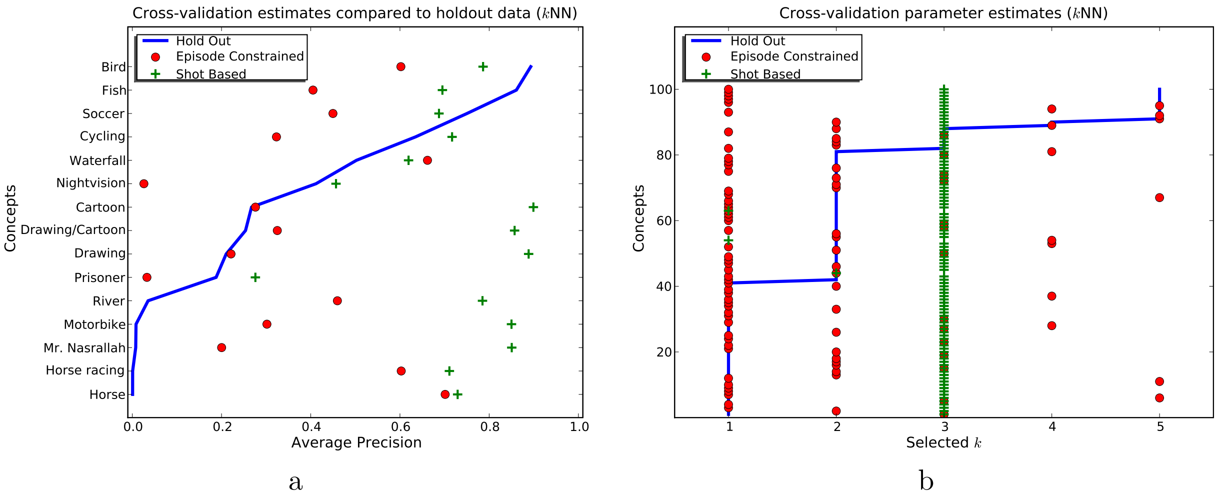


Figure 2.7: (a) Performance estimates of episode-constrained and shot based cross-validation compared to the true hold-out performance for some selected concepts with the kNN classifier. (b) Selected parameters of episode-constrained and shot based cross-validation compared to the true hold-out performance for the kNN classifier.

The first thing that is striking about the estimated parameters in figure 2.7b is the discrepancy between methods in selecting the best classifier parameter. The shot based cross-validation method for kNN selects $k = 3$ for 97 out of 100 concepts, whereas episode-constrained cross-validation correlates better with the best parameter of the hold-out set. The parameter estimates influence the final classification performance, and we summarize this in table 2.1. In this table we present the mean performance in average precision over all concepts, for both cross-validation methods and for both classifiers. We show the estimated results on training data, and the results on hold-out data where we tune the classifier parameter by selecting the maximum performance according to the cross-validation method at hand.

In analyzing table 2.1, we focus on two points: 1) the accuracy in estimating classifier performance and 2) the final classification performance. Starting with point 1, we consider the difference between the estimated performance on training data and the reported performance on hold-out data. For shot based cross-validation there is considerable difference between the estimated performance on training data and the performance on hold-out data. Specifically, the difference is 0.386 for the kNN classifier, and 0.273 for the SVM classifier. In contrast, for episode-constrained cross-validation the difference between training data and hold-out data is only 0.097 for the kNN, and 0.135 for SVM. This clearly shows that the estimated performance of the episode-constrained cross-validation is more accurate than the performance estimate based on shots. Continuing with the issue of final classification performance, we compare the performance on hold-out data for both methods. An analysis of the hold-out results per concept shows that episode-constrained cross-validation yields equal or better results for 85 concept with kNN and for 79 concepts for SVM. Averaged over all concepts, the episode-constrained method outperforms the shot based method by 14% for kNN, and 5% for SVM, as shown in table 2.1. The smaller improvement in the case of the SVM is due to a large performance increase when near-duplicates are present in the hold-out set. Since near-duplicates are very similar, the SVM with its parameters tuned by shot based cross-validation is very well tuned to these duplicates. The large performance increase for near-duplicates leads to a disproportional increase in the average value over all concepts. The near-duplicates mostly consist of commercials: *Bird* (+0.09), *Waterfall* (+0.10), *NightVision* (+0.19), *SwimmingPool* (+0.09), *Beach* (+0.06). Nevertheless, for the SVM the performance for 79 out of 100 concepts improves by using episode-constrained cross-validation. Therefore these results show that performance estimation with episode-constrained cross-validation is considerably more accurate than using shot based cross-validation, and that this improvement in performance estimation

	Shot Based		Episode-Constrained	
	kNN	SVM	kNN	SVM
Training set	0.573	0.474	0.310	0.345
Hold-out set	0.187	0.201	0.213	0.210

Table 2.1: The mean performance in AP over all concepts using the estimated parameters as selected by each method.

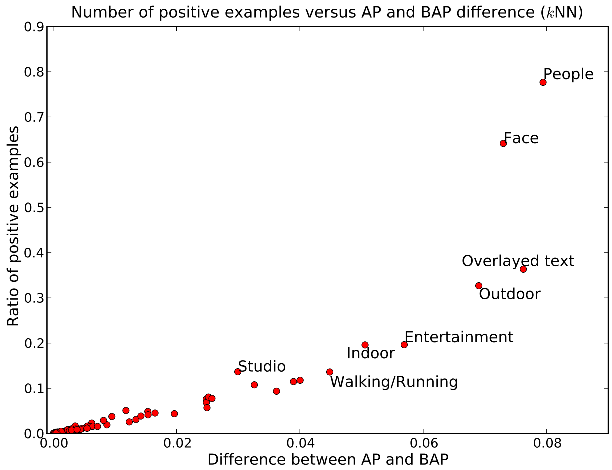


Figure 2.8: The difference per concept between estimated average precision (AP) and balanced average precision (BAP) on training data.

directly translates to an improvement in final classification performance.

2.6.2 The Influence of Balanced Average Precision

Here, we evaluate the assumptions and motivation of using balanced average precision. Balanced average precision allows a fair comparison between collections with an unbalanced number of relevant elements. We assumed that unbalanced collections are more likely to occur with episode constrained cross-validation, since atomic sets of shots hamper an equal distribution of relevant elements over the cross-validation folds. In order to test this hypothesis, we compare the spread of the relevant elements over the folds for both episode-constrained cross-validation and the traditional shot based cross-validation. Specifically, we compute the standard deviation of the number of relevant elements per fold, averaged over all concepts. Both methods of cross-validation have on average 135.21 relevant elements per fold, where the average standard deviation of the shot based and episode-constrained cross-validation method is 0.40 and 30.51 respectively. The difference between both standard deviations clearly shows that episode-constrained cross-validation creates significantly more unbalanced folds than shot based cross-validation. Hence, the motivation of using balanced average precision with episode-constrained cross-validation is sound.

The unbalanced folds in episode-constrained cross-validation necessitate the use of balanced average precision. However, the difference between balanced average precision (BAP) and traditional average precision (AP) may not necessarily prove significant. We evaluate this significance on the Challenge Problem. We employ episode-constrained cross-validation for classifier parameter selection and compare the scores of average precision versus balanced average precision. The results on the Challenge Problem show no difference in parameter selection for both the kNN as the SVM classifier. Hence, for this dataset there is no difference between average precision and balanced average precision. In figure 2.8 we show the difference between AP and BAP compared to the ratio of positive examples for a concept. As illustrated in figure 2.8, there are 89 out of 100 concepts with less than 10% positive examples. Such concepts with relatively few positive examples are less

Fold	AP	BAP	% Relevant shots
1	0.919	0.863	63
2	0.917	0.864	60
3	0.900	0.824	65
4	0.911	0.841	66
5	0.867	0.779	61
6	0.906	0.830	66
7	0.873	0.785	62
8	0.892	0.820	61
9	0.896	0.810	67
10	0.930	0.866	70

Table 2.2: Average precision (AP) balanced average precision (BAP) scores and the percentage of relevant shots in each fold for the concept *Face*.

affected by unbalanced data. As we have shown in figure 2.5, the benefit of BAP comes into its own with larger number of positive examples. As an example, the scores on the cross-validation folds for the concept *Face* are given in table 2.2. This table shows that the over-estimation bias in average precision does occur, however not often enough. For example, when comparing the scores for fold 1 and fold 2, the AP in fold 1 is higher than the AP in fold 2, whereas the BAP for fold 1 is lower than the BAP for fold 2. The same holds for fold 8 and 9. Therefore, despite that there is no difference between AP and BAP for parameter selection on this dataset, the unbalanced data does have a biased effect on average precision. Thus, when using episode-constrained cross-validation balanced average precision is preferred over average precision.

2.7 Conclusions

In this Chapter, we compare two methods of cross-validation for estimating classification performance for semantic concept detection in video. The traditional method of cross-validation is based on shots, whereas we propose a method based on episodes. An episode-constrained method for cross-validation prevents the leaking of similar shots to the rotating hold-out set. We use a whole video as an episode. However, video story segmentation [50, 144] seems a likely alternative to obtain natural episodes. Since episode-constrained cross-validation tends to produce sets with an unbalanced number of relevant items, we introduce balanced average precision. Balanced average precision is an unbiased alternative to average precision. In contrast to average precision, balanced average precision normalizes for the number of relevant items and is therefore a theoretically better choice when dealing with sets that contain an unbalanced number of relevant elements. Experimental results show that the bias of average precision for unbalanced data does occur. However, in our dataset, balanced average precision performs equal to average precision because of the low ratio of positive examples in this dataset. Further experimental evaluation show that the episode-constrained method yields a more accurate estimate of the classifier performance than the shot based method. Moreover, when cross-validation is used for parameter optimization, the episode-constrained method is better able to estimate the optimal classifier parameters, resulting in higher performance on validation data compared to the traditional shot based cross-validation.

Chapter 3

Visual Scene Categorization by Learning Image Statistics in Context[1]

3.1 Introduction

Often, real world images only make sense when captured in context. For example consider an image of a harbor, a city skyline, or a conference meeting. Such scenes are captured more by the ensemble of objects, rather than by individual objects. Therefore, scene recognition differs from object recognition [32, 34, 71, 79] in that not only the foreground is the focus of recognition. Object recognition concentrates on the important task of detecting features relevant to one instance of an object, preventing as much as possible the inclusion of background features. Here we address the problem of scene categorization, *including* background and surrounding objects, that is, the context. Hence, we aim to contribute to content based image and video analysis by establishing a robust method for the learning and subsequent classification of scene categories.

Instead of using image features directly for scene categorization [131], several approaches [33, 73, 94, 101, 124, 127, 130, 145] make use of an intermediate image description step. This intermediate step consists of labeling a part of the image by its best representative out of a predefined codebook vocabulary. Using a codebook allows for density estimation [130], latent class analysis [33, 101, 124], and low level semantic grouping [73, 94, 127, 145]. An inherent problem of the codebook approach is choosing the vocabulary. If the vocabulary is too large, each part of the image will match to a single, unique, vocabulary element, which defies the purpose of a codebook. On the other hand, if the vocabulary is too small, several different image parts will be represented by the same vocabulary element. Thus, the codebook vocabulary determines the expressiveness and the discriminatory power of the method. In contrast, we propose to use the similarity to all codebook vocabulary elements, retaining expressiveness and discriminatory power.

In this Chapter, we exploit the statistical information locally available in images to categorize the scene. As shown by Torralba and Oliva [94, 127], scene categorization has strong correlation with the statistical structure within the image. Here, we provide a method for scene categorization, which does not need the input to be centered and oriented in a similar direction. Furthermore, the proposed method is robust over different datasets. We used one set of annotated video sequences to model video categories, object images, and photo stock collections. Moreover, the experiments are conducted with at least 50 categories using over ten thousand images. To our knowledge, this is the largest experimental evaluation, in number of categories and number of images, present in the literature.

The outline of the Chapter is as follows. The next section will give an overview of the visual features which effectively capture local image statistics. Subsequently, section 3.3 shows our method for learning context from local image statistics by learning the similarities of proto-concepts within images. Section 3.4 experimentally demonstrates our method on 4 dataset: 1) We show categorization results for 50 categories recognized on a large video collection of 160 hours of video. 2)

[1] Published in *CVPR International Workshop on Semantic Learning Applications in Multimedia* [136].

To show the generality of our approach, we provide a comparison with state-of-the-art by learning and recognizing the 101 object categories in the Caltech collection. 3) To compare against the state-of-the-art we participated in the Pascal VOC object recognition challenge 4) to show the robustness of our approach, we will use the proto-concepts extracted from the video data to learn 89 categories from the Corel photo collection (16,500 images), and recognizing the learned categories in a completely different photo stock (ArtExplosion, 62,000 images). Finally, section 3.5 concludes the Chapter.

3.2 Visual Features

Modeling visual data heavily relies on qualitative features. Good features describe the relevant information in an image while reducing the amount of data representing the image. To achieve this goal, we use Weibull-based features [42]. By using Weibull-based features, we combine color invariance with natural image statistics resulting in an effective but compact description of local image content. Color invariance aims to remove accidental lighting conditions, while natural image statistics efficiently represent image data.

3.2.1 Natural Image Statistics

The statistical content of the scene provides robust cues for scene recognition [94, 127]. Hence, there is a direct relation between scene structure, and image statistics. In this Chapter, we exploit the statistical information locally available in images to categorize the scene. An example of such a categorization may be "close-up, indoor, outdoor, panorama". At a higher level of semantics, one may aim at categorizing the sort of objects in the image: "anchorman, explosion, boats, rural, city view, traffic jam". As will be demonstrated in this Chapter, both categorizations have strong correlations with the statistical structure of the scene.

We capture the local statistics of the image by applying Weibull-based features [42] where natural image statistics is used to effectively model texture information. For sake of completeness, we provide a short overview of Weibull-based features.

Texture is described by the distribution of edges for a certain region in an image. Hence, a histogram of a Gaussian derivative filter is used to represent the edge statistics. Since there are more non-edge pixels then there are edge pixels, a histogram of edge responses for natural images always has a peak around zero, i.e.: many pixels have no edge responses. Additionally, the shape of the tails of the distribution is often in-between a power-law and a Gaussian distribution. The tail emphasizes the long-range correlation between edge pixels in the image. A heavy power-law tail indicates a strongly contrasting object-background edge, whereas a Gaussian tail indicates a noisy, high-frequency texture region. The complete range of image statistics in natural textures can be well modeled with an integrated Weibull distribution [42]. This distribution is given by

$$\frac{\gamma}{2\gamma^{\frac{1}{\gamma}}\beta\Gamma(\frac{1}{\gamma})}\exp\left\{-\frac{1}{\gamma}\left|\frac{r-\mu}{\beta}\right|^{\gamma}\right\} \qquad , \tag{3.1}$$

where r is the edge response to the Gaussian derivative filter and $\Gamma(\cdot)$ is the complete Gamma function, $\Gamma(x) = \int_0^\infty t^{x-1}e^{-1}dt$. The parameter β denotes the width of the distribution, the parameter γ represents the peakness of the distribution, and the parameter μ denotes the origin of the distribution. See figure 3.1 for examples of the integrated Weibull distribution.

The integrated Weibull distribution can be estimated from a histogram of filter responses with a maximum likelihood estimator (MLE). The parameters μ, β and γ are estimated by taking the derivatives of the integrated Weibull distribution to the respective parameters and setting them to zero. The parameters β and γ are dependant on each other, therefore a binary search scheme is utilized to estimate the best β and γ combination.

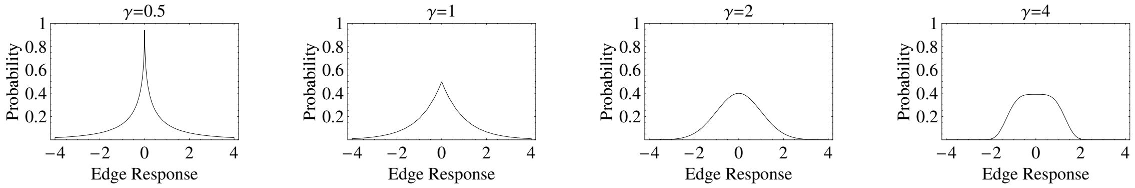


Figure 3.1: Some examples of the integrated Weibull distribution for $\beta = 1$, $\mu = 0$, varying values for $\gamma \in \{\frac{1}{2}, 1, 2, 4\}$.

Since the integrated Weibull distribution characterizes edge responses, the parameters of the distribution correspond to different image properties. The β parameter represents the width of the distribution. A high value of β corresponds to a wide distribution which indicates an image with high contrast. The γ parameter denotes the slope of the distribution. A low value of γ (< 1) represents a highly peaked distribution, which corresponds to an image with smooth surfaces. A medium value of γ ($1 < \gamma < 2$) indicates a smooth distribution, which represents Gaussian noise-like images. A high value of γ (> 2) is an indicator of a histogram that does not follow a Weibull distribution. Specifically, an image with a regular pattern, for example the beams of the American flag, produces a histogram that has multiple peaks. The MLE estimator of the integrated Weibull distribution will represent multiple peaks in the histogram by a smooth and flat distribution, represented by a high value of γ. The μ parameter represents the mode of the distribution. The position of the mode is influenced by uneven illumination and colored illumination. Hence, to achieve color constancy the values for μ may be ignored.

To assess the similarity between two integrated Weibull distributions, a goodness-of-fit test is utilized. The measure is based on the integrated squared error between the two cumulative distributions, which is obtained by the Cramér-von Mises statistic,

$$C^2 = \int_0^1 [F(x) - G(x)]^2 \, dF(x) \quad , \tag{3.2}$$

where F is the test distribution, and G represents the target distribution, where both are cumulative distributions. For two Weibull distributions with parameters β_F, γ_F and β_G, γ_G a first order Taylor approximation yields the log difference between the parameters. Therefore, we define a measure of similarity between two Weibull distributions is given by the ratio of the parameters,

$$C^2(F, G) = \sqrt{\frac{\min(\beta_F, \beta_G)}{\max(\beta_F, \beta_G)} \frac{\min(\gamma_F, \gamma_G)}{\max(\gamma_F, \gamma_G)}} \quad . \tag{3.3}$$

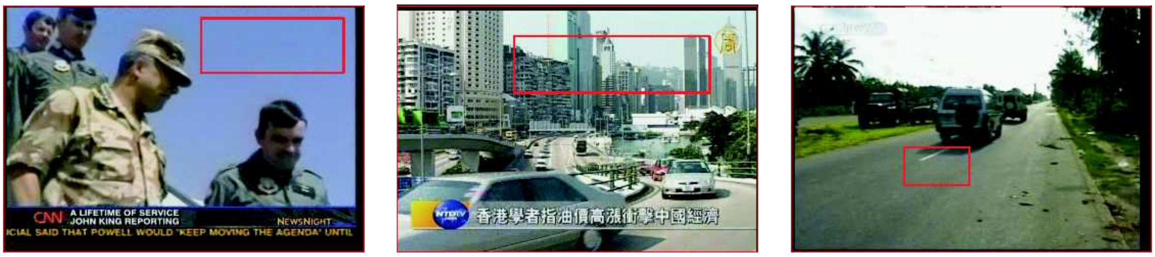
In summary, Weibull-based features provide a texture descriptor based on edges. Moreover, the features rely heavily on natural image statistics to compactly represent the visual information. For a more detailed elaboration on Weibull-based features, see [42].

3.2.2 Color Invariant Edge Detection

Here we combine color invariant edge responses with natural image statistics to end up with color invariant Weibull-based features. Color invariance aims to remove accidental lighting conditions, while Weibull-based features efficiently represent image statistics.

We first decorrelate the RGB channels by a linear transformation to an opponent color representation. Advantage of the use of an opponent color space is that color values are decorrelated. Hence, for a distinctive image content descriptor, we may as well use the marginal, one-dimensional, distributions for each of the color channels. This in contrast to the histogram of the full 2D chromatic or 3D color space (see e.g. [26, 45]).

Further decorrelation of color information can be achieved by using photometric invariant edge detectors. The invariant W (notation from [44])) measures all intensity fluctuations except for


CNN
A LIFETIME OF SERVICE
JOHN KING REPORTING
NEWSNIGHT
ICIAL SAID THAT POWELL WOULD "KEEP MOVING THE AGENDA" UNTIL
香港學者指油價高漲衝擊中國經濟

Sky Building Road

Figure 3.2: Three examples of annotated regions in video.

overall intensity level. That is, edges due to shading, cast shadow, and albedo changes of the object surface. These invariants are equivalent to Gaussian derivative filters for color images, where 6 orthogonal derivatives may be distinguished. W_x, W_y detect edges in intensity, whereas $W_{\lambda x}, W_{\lambda y}$ and $W_{\lambda\lambda x}, W_{\lambda\lambda y}$ detect edges in the two orthogonal chromatic color components.

Thus, color invariant edge responses, are invariant to changes in intensity, and decorrelate the RGB channels, allowing the weibulls to be computed on marginal densities.

3.3 Contextures: Regional Texture Descriptors and their Context

Building towards semantic access to image collections, we aim to decompose complex scenes in proto-concepts like vegetation, water, fire, sky etc. These proto-concepts provide a first step to automatic access to image content [145]. Given a fixed vocabulary of proto-concepts, we assign a similarity score to all proto-concepts for all regions in an image. Different combinations of a similarity histogram of proto-concepts provide a sufficient characterization of a complex scene. We introduce the notion of contextures, where global texture and local texture information and their context are used to describe visual scene information.

By using the similarity to all vocabulary elements, we introduce an alternative to codebook approaches [33, 101, 124, 130, 145]. A codebook approach uses the single, best matching vocabulary element to represent an image patch. For example, given a blue area, the codebook approach must choose between water and sky, leaving no room for uncertainty. We propose to use the distances to all vocabulary elements. Hence, we model the uncertainty of assigning an image patch to each vocabulary elements. By using similarities to the whole vocabulary, our approach is able to model scenes that consist of elements not in the codebook vocabulary.

3.3.1 Region Annotation of Proto-Concepts

In order to recognize concepts based on low-level visual analysis, we annotated 15 different proto-concepts: building (321), car (192), charts (52), crowd (270), desert (82), fire (67), US-flag (98), maps (44), mountain (41), road (143), sky (291), smoke (64), snow (24), vegetation (242), water (108), where the number in brackets indicates the number of annotation samples of that concept. These proto-concepts are chosen by their relevance for concept detection in the TRECVID video benchmark. Although they seem to be tuned to the problem at hand, we will show these concepts to generalize (including the annotation effort) to various datasets. Fig. 3.2 shows an example of some regional annotations. We use the TRECVID 2005 [91] common annotation effort as a basis for selecting relevant shots containing the proto-concepts. In those shots, we annotated rectangular regions where the proto-concept is visible for at least 20 frames.

For each of the proto concepts, visual characteristics are captured by their Weibull-based features as described above.

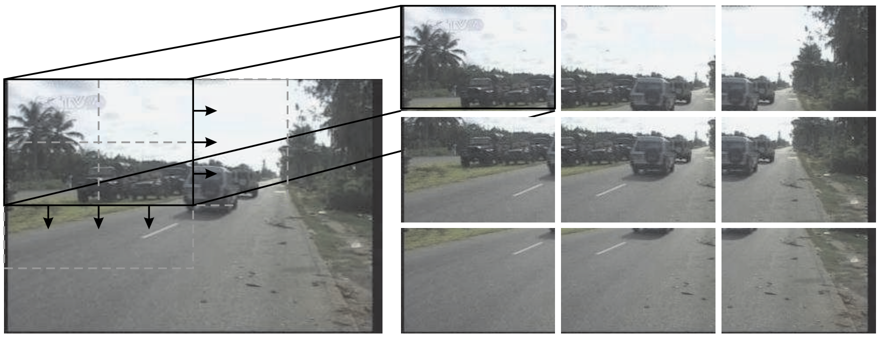


Figure 3.3: An example of dividing an image up in overlapping regions. Here, the region size is a $\frac{1}{2}$ of the image size for both the x- and y-dimension. The regions are uniformly sampled across the image with a step size of half a region. Sampling in this manner identifies nine overlapping regions.

3.3.2 Region descriptors

The visual detectors aim to decompose an image in similarities to proto-concepts like vegetation, water, fire, sky etc. To achieve this goal, an image is divided up in several overlapping rectangular regions. The regions are uniformly sampled across the image, with a step size of half a region, see figure 3.3 for an example. The region size has to be large enough to assess statistical relevance, and small enough to capture local textures in an image. We utilize a multi-scale approach, using small and large regions.

A visual scene is characterized by both global as well as local information. For example, a picture with an aircraft in mid air might be described as "sky, with a hole in it", sky being globally present in the image except for a local distortion: the aircraft. To model this type of information, we use a proto-concept occurrence histogram where each bin is a proto-concept. The values in the histogram are the similarity responses of each proto-concept, to the regions in the image.

We use the proto-concept occurrence histogram to characterize both global and local texture information. Global information is described by computing an occurrence histogram accumulated over all regions in the image. Local information is taken into account by constructing another occurrence histogram for only the response of the best matching region. For each proto-concept, or bin, b the accumulated occurrence histogram and the best occurrence histogram are constructed by,

$$H_{accu}(b) = \sum_{r \in R(im)} \sum_{a \in A(b)} C^2(a,r) \quad , \tag{3.4}$$

$$H_{best}(b) = \arg\max_{r \in R(im)} \sum_{a \in A(b)} C^2(a,r) \quad , \tag{3.5}$$

where $R(im)$ denotes the set of regions in image im, A(b) represents the set of stored annotations for proto-concept b, and C^2 is the Cramér-von Mises statistic as introduced in equation 3.2. We denote a proto-concept occurrence histogram of an image as a contexture for that image. We have chosen this name, as our method incorporates texture features in a context. The texture features are given by the use of Weibull-based features, using color invariance and natural image statistics. Furthermore, context is taken into account by the combination of both local and global region combinations.

The contexture H_{accu} counts the relative amount of proto-concepts present in a scene, hence *how much* of a proto-concept is present in a scene. The contexture H_{accu} is important in characterizing, for example, airplanes and boats. In these cases, the accumulated histogram indicates the presence of a large water body or a large area of sky. The contexture H_{best} only indicates the presence of proto-concepts, hence indicates *which* proto-concepts are present in a scene. In this way, constellations of proto-concept indicate scene type without specifying the relative area each proto-concept should occupy. This is of importance in characterizing, for example, military actions in the middle east, where the combined presence of road, desert, and fire, turns out to be very effective. Note that, by using occurrence histograms and dense sampling over the image, the proposed method is translation invariant, thus, the exact layout of the scene is not strictly enforced. Opposed to [94], placing objects in the centre of the scene, and strictly aligning them in a similar direction is not necessary for our categorization scheme.

In contrast to codebook approaches, our method is not limited to the visual categories that can be described by the vocabulary of proto-concepts. Not every image contains proto-concepts like 'sky', 'vegetation', 'water'. Scenes where the specific proto-concepts do not occur can nevertheless be described by contextures. This is the case, since the similarity to a proto-concept is used, not the proto-concept itself. A robust and consistent similarity measure will give similar values for similar scenes. For scenes that belong to the same visual category, there is some common visual denominator that ties the scenes to the category. Hence, there will be a correlation between the contextures of scenes that belong to the same category. For example, an office scene might consist of large surfaces with sharp edges (desks) and multicolored highly textured and oriented regions (books). The similarity to proto-concepts like 'sky' and 'vegetation' will not be high since none of the proto-concepts are present. However, the responses of the proto-concepts will be the same for another office scene, because this new scene will consist of similar regions. Thus, a scene can be expressed in a degree of similarity to a vocabulary of proto-concepts, without containing any of the proto-concepts.

Learning of scene categories is approached by default machine learning techniques. The contextures are extracted from example images, human labeled to belong to a given category, and subsequently fed into a support vector machine (SVM) with a radial basis function for scene category learning.

3.4 Experiments

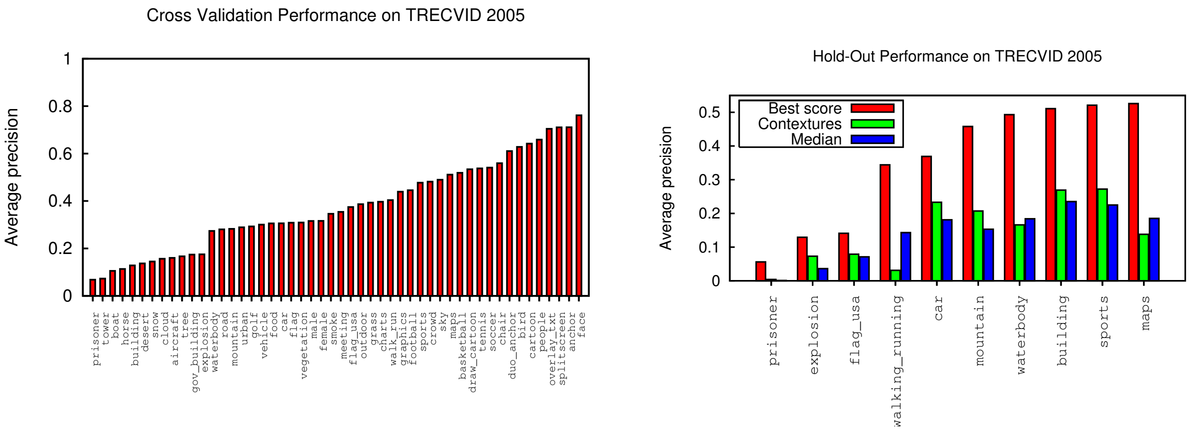
Contextures can be computed for different parameter settings. Specifically, we calculate the contextures at scales $\sigma = 1$ and $\sigma = 3$ of the Gaussian filter. Furthermore, we use two different region sizes, with ratios of $\frac{1}{2}$ and $\frac{1}{6}$ of the x-dimension and y-dimensions of the image. The combination of all these parameters yields a single vector, which is used for scene classification.

3.4.1 TRECVID video benchmarks

The TRECVID video benchmark 2005 [91] provides nearly 170 hours of news video (English: CNN, NBC, MSNBC; Chinese: CCTV4, NTDTV; Arabic: LBC). The goal is to retrieve shots from this collection, which are relevant to a predefined topic. The National Institute of Standards and Technology (NIST) provides the video collection to all participants, and scores the returned rankings by human evaluation.

Video retrieval is evaluated by the relevance of a shot, while contextures are based on one image. To generalize our approach to shot level, we extract 1 frame per second out of the video, and then aggregate the frames that belong to the same shot. We use two ways to aggregate frames: 1) average the contexture responses for all extracted frames in a shot and 2) keep the maximum response of all frames in a shot. This aggregation strategy accounts for information about the whole shots, and information about accidental frames, which might occur with high camera motion. However, since we do not use keyframes, we lose information about exactly identical shots, like commercials.

We learned 50 categories on the video data, shown in figure 3.4(a). The results provided here gives an impression of the quality of visual only detection by using our method of scene categorization, compared to state-of-the-art video retrieval. For all 50 visual concepts we extracted from the video, 10 categories are evaluated by NIST. In figure 3.4(b) we provide the average precision for these 10 categories for our method against the best and the median result for all



(a)

(b)

Figure 3.4: (a) Performance measured in average precision of 50 visual-only detectors on TRECVID data. The score was computed by three fold cross-validation. (b) Best, median and our average precision scores for the 10 concepts on TRECVID 2005 hold-out data, evaluated by NIST.

33 other participants. We do not get the best results however obtain competitive results to all participants.

Overall, the proposed scene categorization turns out to work effectively for 1) scenes where spatial context is uniform, like individual sports (soccer, tennis, basketball, football), 2) typical studio settings (anchor, face, spitscreen), and 3) well constrained environments (e.g., “chairs” and “tables” coincides with interview settings or political items in news). Performance for combinations of these categories are not well learned from examples alone (see e.g. sports), and need a higher level aggregation step. Furthermore, natural scene categories are well represented by the proposed scheme, for example mountains, waterbody, vegetation, smoke. Visual inspection shows that the scene categorization is well able to generalize learned concepts to an unseen test set. Note that for the 10 evaluated concepts, TRECVID results for at least 3 concepts (waterbody, cars, mountains) are dominated by commercials (identical copy detection), for which we did not make an additional effort.

3.4.2 Caltech 101 object categories

In the previous section we gave an impression of our scene categorization on a large collection of video data. From the training set of the TREC video collection, the proto-concept annotations have been extracted. Hence, the proto-concepts are tuned to the type of data (compression, quality), and possibly include domain specific information. An important research question is if the learned similarity histograms of proto-concepts, at the heart of our method, easy generalize to other domains and image qualities. Here, we compare performance on a standard collection of web-images: the Caltech 101 object categories.

In figure 3.5 we compare classification performance against Serre et al. [110]. For recognition of each single categories against a background class of Google images, (one vs background), performance is not as good as by Serre et al. This can be explained by the fact that the Caltech collection contains several manipulation artifacts, in that objects have been centered and orientation has been normalized within each category. Furthermore, several computer graphics and cartoons are included in object categories, and, more important, convey a large portion of the background class. Hence, our natural image statistics based description is not too adequate here. However, a perfect classification of foreground-background gives no indication of performance if an unknown scene has to be classified. You can have perfect one-vs-background, and at the same time being poor in separating all 101 classes. We expect our method to perform better under one-against-all

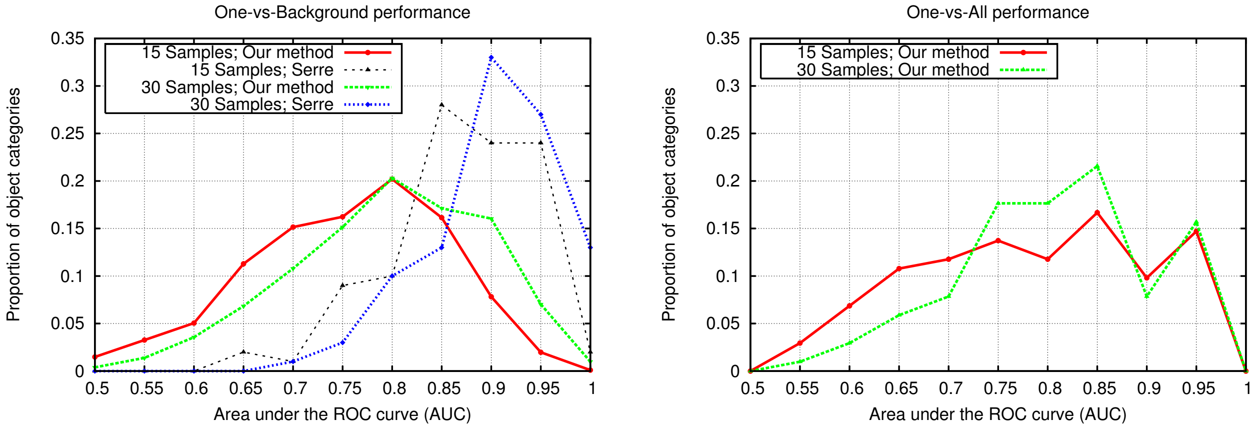


Figure 3.5: Performance histogram of one class vs. background class on the Caltech 101 object dataset for different numbers of training examples.

since in that case the number of cartoon images becomes less dominant.

Our performance on multiclass classification for 15 training samples is 33.2% correct classification, and for 30 training samples 42.3% correct classification (chance 1/101 is below 1%). Compared to the paper by Fei Fei et al. [32], who reach 16% correct classification for 15 examples. Serre et al. [110] has 35% for 15 examples, and 42% for 30 examples, comparable to our results. Holub et al. [58] obtain 40% classification accuracy for 20 training examples. Berg et al. [8] reach best performance of 45% with 15 examples.

Note that we obtain a similar performance as the methods cited above, with a limited feature set derived from only 12 Gaussian derivative filters. Hence, our method generalizes generalizes beyond the original domain of video to web images.

3.4.3 Pascal VOC datasets

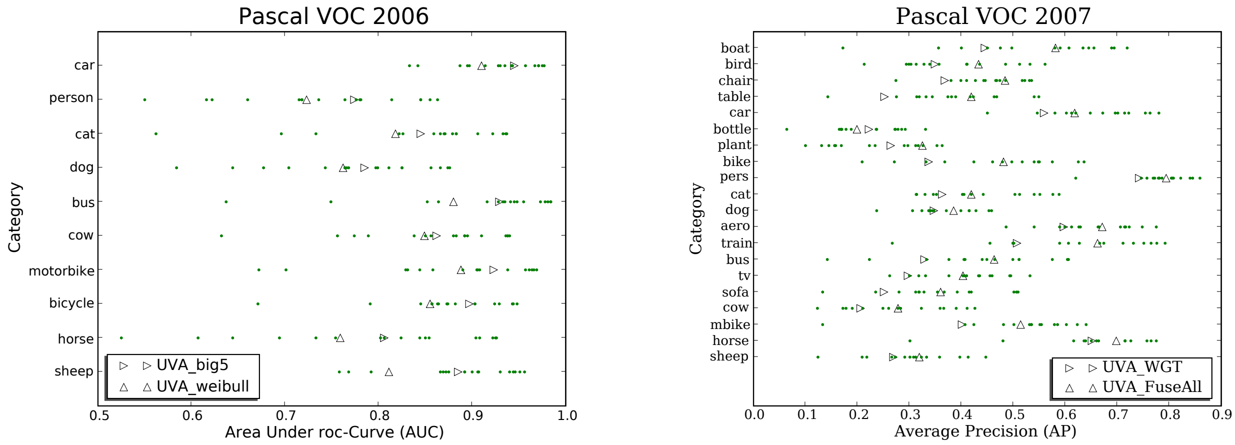
We performed additional categorization experiments on the 2006 and 2007 Pascal VOC object categorization datasets [30, 28]. These datasets contain 5,304 images in the 2006 collection and 9,963 images in the 2007 set. For both of these sets, half of the collection is used for training, and the other half for testing. Note that at the time of the competitions the ground truth for the test-set was not available.

VOC 2006 evaluation

For the Pascal VOC 2006 challenge [30], we submitted two methods. The first method is the Weibull-features with contextures as presented earlier. The second method uses a combination of several detectors and descriptors. The output of several image descriptors on the train+val set is clustered with a radius-based clustering algorithm. These clusters are subsequently used to characterize an image in the whole set. The four detectors consist of: 1) An overlapping 2D grid, 2) Maximally Stable Extremal Regions (mser), 3) Harris Laplacian and 4) Hessian Affine.

The five image descriptors consist of: 1) Wiccest Features, 2) Sift, 3) Spin, 4) Gloh, and 5) Shape Context.

Based on cross validation performance on the train set, we selected the best representative for each descriptor. The best results were given by these five: mser.spin + grid.weibull + mser.shapeCtx + harlap.sift + hesaff.gloh. Early fusion of the image characterizations based on the clusters were used to train a Support Vector Machine classifier on the train set, which was used to predict scores on the test set. The results are given in figure 3.6. The results show that the combination of methods always outperforms the single method. Furthermore, it can be seen that our approach is not the best, but performs competitively



a

b

Figure 3.6: Comparing Wiccest-only features with a fusion of popular descriptors and detectors in the VOC 2006 and 2007 challenge. (a) VOC 2006, UVA_weibull denotes the Wiccest features, whereas the UVA_big5 indicate the fusion of the five image descriptors. (b) VOC 2007, UVA_WGT denotes the Wiccest features, whereas the UVA_FuseAll indicate the results for late fusion.

Africa Agricltr (Architct Agricltr)(Agriculture) Alaska Architecture (ArchitI ArchitII ArctEuro ArctUrbn ArctWrld)(Architecture) Arizona Austrlia Aviation (Aviation WarPlane)(Aircraft Air_Show) Ballet Balloon (Balloon1 Balloon2)(Balloons) Boats (Boats Nautical TallShip)(Nautical Boats_&_Ships) Canada Caribean Car (Car_Brit Car_Old1 Car_Old2 Car_Perf Car_Race Car_Rare Car_BGs)(Auto_Racing Automobiles Classic_Autos_&_Motorcycles) Castle (CastleI CastleII Castles)(Castles) China (China ChinaII)(China) Religion (Churches RelgnI RelgnII RelgnIII Worship)(Religion Synagogues_&_Churches) Colorado Sci-Tech (Indstry Industry Sci&Tech ComTech)(Science_&_Industry Industrial) Work (Work)(People_at_Work) CstaRica (CstaRica)(Costa_Rica_&_Guatemala) Cuisine (Cuisine Cuisines Food Food1 Food2)(Food) CzechRep (CzechRep)(Czechoslovakia) Sunsets (Sunsets Dawn&Dsk SkiesI SkiesII)(Sunset_&_Sky Clouds_&_Sky Sunrise_&_Sunset) Desert (DesertI DesertII)(Desert) Egypt (Egypt1 Egypt2)(Egypt Egyptian_Art_&_Architecture) England Garden (Estate Plants Gardens FlwrBeds)(Gardens) Europe (Europe Europe_E Europe_S Euro_Scn)(Europe) Evrglade Fireworks (Firewrk1 FireWrk2)(Fireworks) Flags Florida Flower (Flower1 Flower2 FlowerI FlowerII)(Plants Flowers) Forest (Forest Forests)(Trees_&_Plants Trees) France Golf Greece Hawii (Hawii)(Hawaii Hawaii) HongKong Horses (Horses Rodeo)(Horse_Sports) India (India_I India_II)(India) Interior (Interior)(Interiors) Italy Japan (Japan Japan1 Japan2)(Japan) Jewelry Kenya Korea Landmark (Landmark Landmrks)(Landmarks Cityscapes Real_Estate) Ruins (Ruins Ruins1 Ruins2 Ruins3)(Ancient_Ruins) LightHouse (LightHse)(Lighthouses) London MiddleEast (MdleEast)(Middle_East) Mexico (Mexico MexicoCt)(Mexico) Mountain (Mountain)(Mountain_Scenes) Music (Music)(Musical_Instruments) Namibia NwMexico (NwMexico)(New_Mexico) NwZealnd (NwZealnd)(New_Zealand) Oregon People (People1 People2)(People People_General) Quebec Signs (Rd_Signs)(Universal_Signs Signs) Recreation (Recreatn)(Recreation) Road (Road&Hwy)(Roads_of_the_World Highways_&_Byways) Rock-Gem (Rock&Gem)(Rocks_&_Gems) RockForm (RockForm)(Rockscapes) Rural Russia San_Fran (San_Fran)(San_Francisco) Scotland Sculpture (Sculpt SculptI SculptII)(Statuary_&_Objects) Seasons (Seasons)(The_Four_Seasons) SE_Asia (SE_Asia)(Asia) SF_Doors (SF_Doors)(Doors_&_Windows) Space (Space Space1 Space2)(Space Science) Sport (SportsI SportsII)(Sports) SubSea (SubSea1 SubSea2 SubSea3 SubSeaII)(SubSea_I Under_the_Sea) Thailand Train (Train1 Train2 Train3)(Trains_&_Tracks Trains) Transport (Trans1 TransII)(Transportation) Utah Vietnam (Vietnam)(Vietnam_Voyage) Virginia Wash_DC (Wash_DC)(Washington_DC) Waterscape (Waterfal WaterScn Waterway Waves)(Water Waterscapes) WetSport (WetSport Windsurf)(Water_Sports) Winter (Winter)(Winterscapes) Yemen Zimbabwe

Figure 3.7: The 89 base concepts, with corresponding categories in Corel and Artexplosion. The main concept is followed by its constituent categories in brackets. The concepts in brackets are the corresponding categories in Corel and ArtExplosion, respectively.

VOC 2007 evaluation

For the Pascal VOC 2007 challenge [28], we again focus on two methods. The first is again the Weibull-features with contextures as used before. The second method, however, uses late fusion of several descriptors. This late-fusion consists again of several interest point detectors and image descriptors. The results are shown in figure 3.6. Again, the fusion method outperforms the single method, and our performance is competitive with the state-of-the-art.

3.4.4 Corel vs. ArtExplosion

To further evaluate the robustness of our approach, we applied scene categorization on a photo stock. In this experiment, we investigate if scene categories learned from one collection can be applied to a different collection. Note that, from a machine learning perspective, this is a more challenging task then obtaining a training and test set by subdividing a homogeneous collection. We use the Corel and ArtExplosion commercially available photo stock, and take the intersection in categories between the two as dataset, see figure 3.7. Hence, we have 89 categories, on one side 16,499 Corel images, ranging from 99 to 700 examples per category; on the other side 62,072

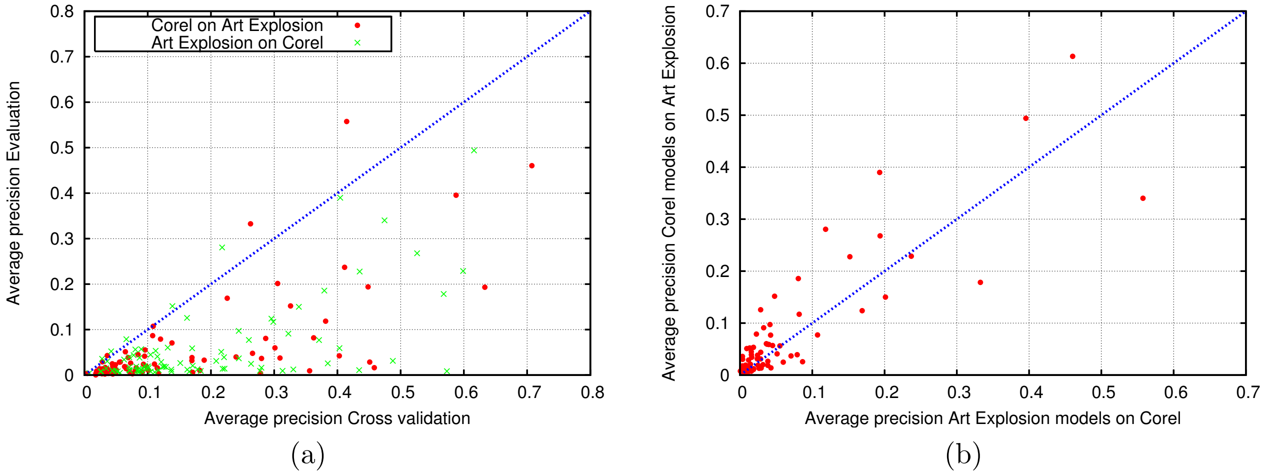


Figure 3.8: (a) Performance measured in average precision on training and evaluation sets. The x-axis is the inter-set cross validation performance, where the y-axis displays the performance scored on the other set. (b) Performance of models evaluated on different sets.

ArtExplosion images ranging from 26 to 4,896 examples per category.

We learned the categories for the Corel collection and ArtExplosion collection separately, and applied the models learned from the one collection to retrieve the categories from the other collection, see figure 3.8(a) and figure 3.8(b). Main result is that geographical locations (countries, cities) are not performing well: for 43 location concepts, there are 39 performing below 0.1 average precision. The remaining 4 locations are (average precision on cross validation between brackets): Italy (0.11); Yemen (0.12); Egypt (0.16); Utah (0.26). The relative high scores are explained by a large overlap in similar places photographed in both Corel and ArtExplosion. Hence, we draw the conclusion that geographical locations can only be categorized by learning and retrieving typical landmarks.

To evaluate categories which do perform well, we made a human judged ground truth of the top-100 results of the non-geographical locations categories. The results are given in figure 3.9, and some examples are shown in figure 3.10. Note that in these 46 categories, still 11,000 Corel images and 50,465 ArtExplosion images are available. For the top 100 results the Corel models evaluated on ArtExplosion score on average better than the ArtExplosion models evaluated on Corel. On average, the Corel on ArtExplosion measured with strict category membership has 21% correct, while manually counting the output of the methods shows it has 33% correct. Conversely, the ArtExplosion evaluated on Corel, with strict categories has 17% correct, and with manual counting of the output shows 24% correct. Categories that are consistently well performing are: architecture, people, wetsport, waterscape, mountain, subsea, flags, balloon, signs, boats, forest, aviation, fireworks, flower, and sunset. Note that building, water, flag, sky, vegetation are proto-concepts learned from the TRECVID video collection. These concepts appear to be transposed to the stock photo collection, increasing performance for related categories.

3.5 Conclusions

In this Chapter we have presented scene category classification by learning the occurrence of proto-concepts in images. We compactly represent these proto-concepts by using color invariance and natural image statistics properties. By exploiting similarity responses as opposed to strict selection of a codebook vocabulary, we have been able to generalize these proto-concepts to be applicable in general image collections. We have demonstrated the applicability of our approach in a) learning 50 scene categories from a large collection of news video data; b) a collection of 101 categories of web images; c) two instances of the Pascal VOC object recognition challenge and d) two large

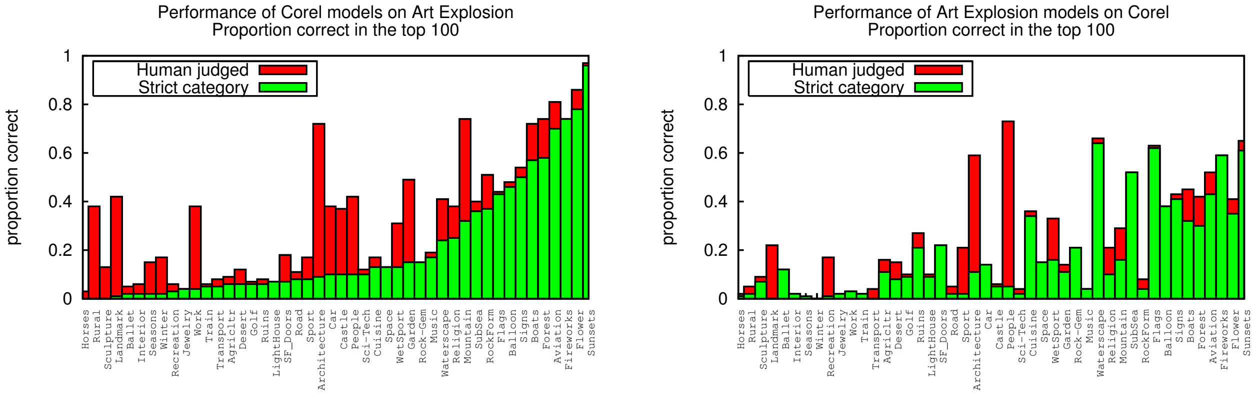


Figure 3.9: Percentage correct classification in the Top 100 results for 46 categories. The ground truth is contrasted with the given categories, where countries and cities are not included as a ground truth can not be established.

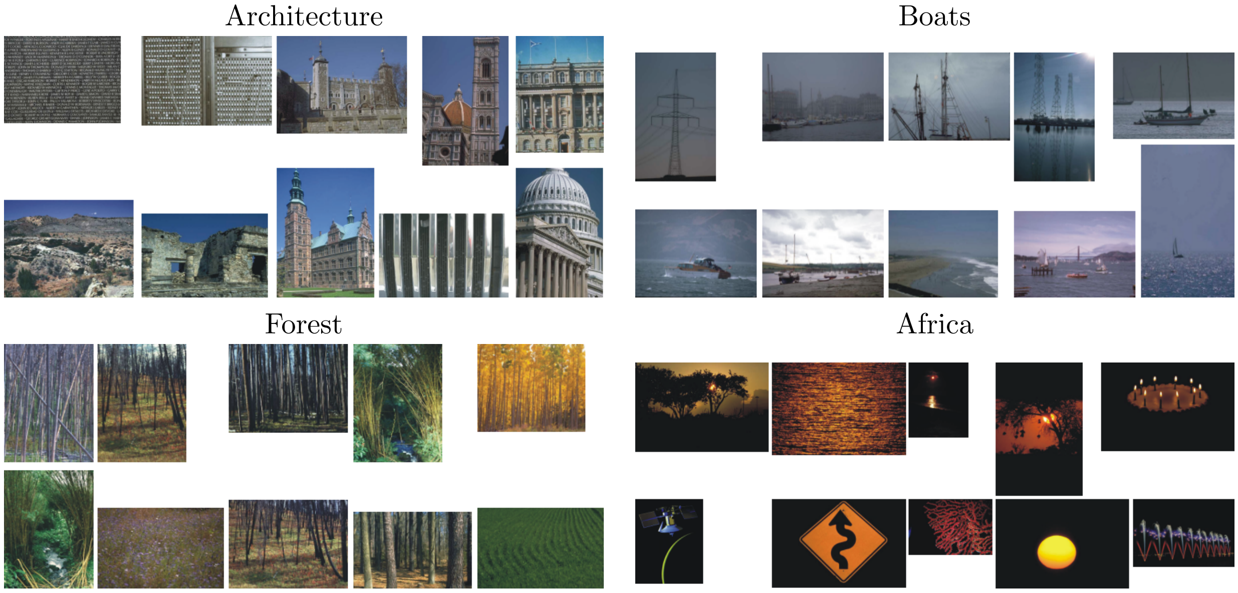


Figure 3.10: Examples of top 10 results. Only for Africa, according to the categories, none are correct.

collections of photo-stock images, comprising 89 categories, where categories are learned from one and categorized from the other.

In conclusion, we have provided an effective scheme for scene categorization. An important contribution is scalability, showing that the proposed scheme is effective in capturing visual characteristics for a large class of concepts, over a wide variety of image sets. Where specific methods may have better performance for specific datasets, we have shown a method which is neither tuned nor optimized in parameters for each collection, other than the TRECVID video dataset. Hence, the method has proven to robustly categorize scenes from learned context.

Chapter 4

Comparing Compact Codebooks for Visual Categorization [1]

4.1 Introduction

Today, digital video is ubiquitous. This omnipresence of digital video material spurs research in automatic content-based indexing. However, given the sheer quantity of available digital video, the applicability and quality of current video indexing algorithms severly depends on their efficiency [52, 108]. One approach to achieve efficiency is by means of a compact, yet powerful representation of the visual data. To this end, this Chapter compares various methods which obtain compact and expressive models for video indexing.

As an instantiation of video indexing, we focus on automatic concept categorization [64, 88, 119, 120, 147]. Applications are mainly found in content-based retrieval and browsing. The goal of concept categorization is to rank shots according to their relevance to a set of predetermined semantic concepts. Some examples of these concepts are *airplane, beach, explosion, George Bush, people walking, etc.*

Many visual concepts are captured as a typical contextual arrangement of objects [4, 57, 66, 87, 94, 127]. For example, consider an image of a beach, a city skyline, or a conference meeting. Such concepts are portrayed by a composition of the image as a whole, rather than characterized by one specific part in the image. Moreover, the background context of an object may provide considerable recognition cues. Consider figure 4.1 where an object is cut out of its surroundings. Without the background information, recognition becomes ambiguous even for humans. Alternatively, in figure 4.2(a), a white patch is placed over the object, where the identity of a hidden object may be derived with high accuracy from the context and nothing but the context. Hence, the background context of an object can be more informative than the object itself. Therefore, in this Chapter we model the whole image for concept categorization, purposely including the context provided by the background.

We describe visual concepts in context with the codebook, or bag-of-visual-words, model. The codebook model is inspired by a word-document representation as used in text retrieval [105]. An schematic of the codebook model is given in figure 4.3. The codebook model treats an image as a distribution of local features, where each feature is labeled as a discrete visual prototype. These prototypes, or codewords, are defined beforehand in a given vocabulary, which may be obtained by unsupervised clustering [13, 33, 62, 67, 95, 100, 114, 125], or manual, supervised annotation [14, 82, 136, 146]. Given a vocabulary, the codebook model allows visual categorization by representing an image by a histogram of codeword counts. The codebook model yields a distribution over codewords that models the whole image, making this model well-suited for describing context. This Chapter strives towards efficient concept categorization by investigating qualitative and compact codebooks.

[1] Published in *Computer Vision and Image Understanding* [138].

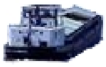


Figure 4.1: *Example of an object that is ambiguous without context.*

4.1.1 Contribution

In this Chapter, we experimentally evaluate various codebook methods to obtain a small, compact, vocabulary that discriminates well between classes. The size of the vocabulary is linked to the discriminative power of the model. A too small vocabulary does not discriminate well between concept categories [140]. Hence, current state-of-the-art methods typically use several thousands of codewords [132, 74]. In a practical application, however, it may not be feasible to use such large number of codewords. Practical objections to a large vocabulary are its storage requirements, working memory usage, and the computation time to train a classifier. Moreover, it has recently been shown that a too large vocabulary severely deteriorates the performance of the codebook model [140]. Therefore, we selected four state-of-the-art methods that each individually focus on improving performance and evaluate these algorithms under a compactness constraint. The compactness constraint is typically ignored by systems who focus solely on performance. The four compacting methods consist of 1) global vocabulary clustering; 2) concept-specific vocabulary clustering; 3) annotating a semantic vocabulary and 4) soft-assignment of image features to codewords. Methods 1-3 deal with vocabulary building, where method 2 is a variant of method 1. Method 4 is a generic approach to increase the expressive power of the codebook vocabulary. We evaluate each of these methods against each other, on a large shared dataset over two different feature types, and two different classifiers.

This Chapter is organized as follows. In the next section we give an overview of the related literature. In section 4.3 we describe the four evaluated methods. We present our experimental setup in section 4.4, whereas we highlight the results in section 4.5. Section 4.6 concludes the Chapter.

4.2 Related Work

Several techniques exist for efficiently retrieving high-dimensional image features in large image collections. Nistér and Stewénius [92] use hierarchical k-means clustering to quantize local image

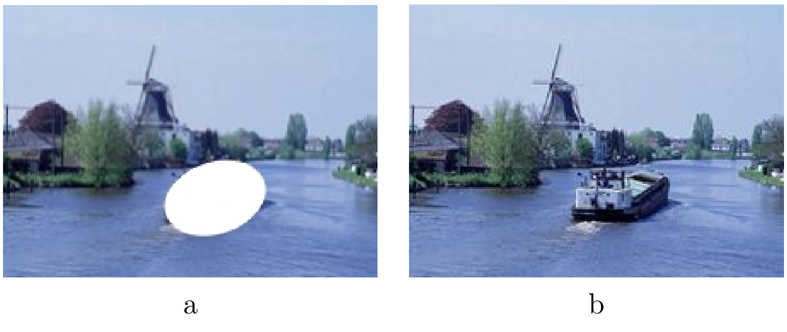


Figure 4.2: *Example showing the influence of context. (a) The surroundings of an object, (b) the whole image. Note that the category of the hidden object in (a) can easily be inferred from the context.*

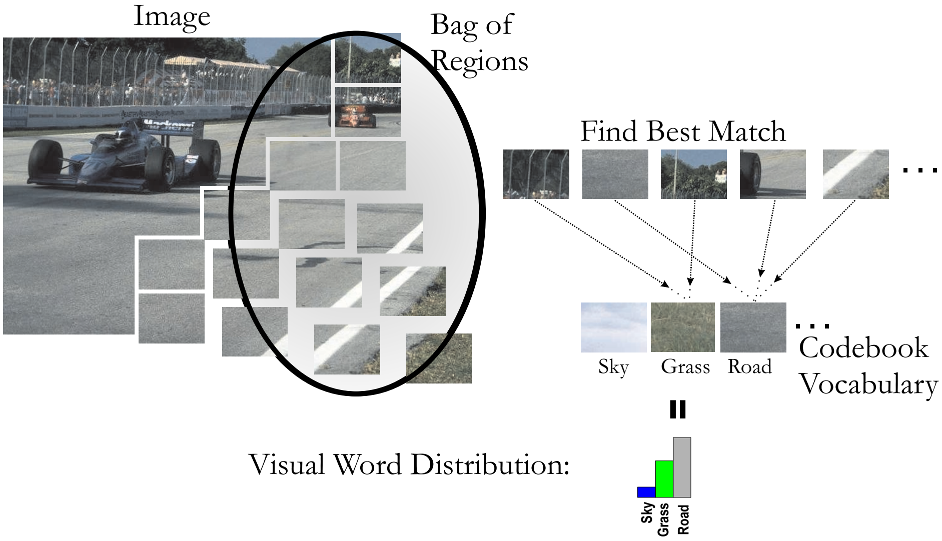


Figure 4.3: *An example of the visual word, or codebook model. An image is represented as a bag-of-regions where each region is represented by the best fitting codeword in the vocabulary. The distribution of the codeword-counts yields the image model.*

features in a vocabulary tree. This vocabulary tree demonstrates efficient feature retrieval in as many as 1 million images. A tree structure is also used by [78] who obtains efficiency gains by reducing the dimensionality of the features by a truncated Mahalanobis metric. Moreover, novel quantization method based on randomized trees is used by [97]. In contrast to a tree structure, Grauman and Darrell [48] present an approximate hashing scheme based on pyramid matching. The pyramid matching allows multi-resolution image matching while the hashing technique allows sub-linear retrieval in large collections of features. Hashing is also used by Kise *et al.* [65] who show that a simple binary representation of feature vectors can result in an efficient approximate nearest neighbor algorithm. Tree and hashing algorithms are well-suited for assigning features to extremely large vocabularies, with millions of centroids. These algorithms, however, do not consider categorization. They focus on recognition of (close to) exact image and feature matches. For categorization with the codebook model, a vocabulary of a million codewords is no longer practical when training a classifier, and a tree-structure does not help out there. The classifier is still left with storing a feature vector of a million codewords for each image. Therefore, we focus on compact vocabularies for efficiency.

A compact codebook model can be achieved by modeling codeword co-occurrence. Under the assumption that frequent co-occurring codewords describe similar information, the vocabulary size may be reduced by merging these codewords. Codeword co-occurrence is typically modeled by a generative probabilistic model [10, 56]. To this end, Fei-Fei and Perona [33] introduce a Bayesian hierarchical model for scene categorization. Their goal is a generative model that best represents the distribution of codewords in each concept category. They improve on latent dirichlet allocation [10] by introducing a category variable for classification. The proposed algorithm is tested on a dataset of 13 natural concept categories where it outperforms the traditional codebook model by nearly 30%. The work by Fei-Fei and Perona is extended by Quelhas *et al.* [100], who investigate the influence of training data size. Moreover, Bosch *et al.* [13] show that probabilistic latent semantic analysis improves upon latent dirichlet allocation. Further contributions using co-occurrence codebook models are by [125]. Typically, a generative model is built on top of a codebook model. Hence, the techniques proposed in this Chapter can easily be extended with co-occurrence modeling. The extra modeling step requires ample additional processing which is less practical for large datasets. Moreover, an additional step makes it harder to evaluate which part of our algorithm is responsible for what. Therefore, in this Chapter, we focus on compact codebook models, without introducing additional co-occurrence modeling steps.

Apart from co-occurrence modeling, a compact codebook may be achieved directly by reducing the vocabulary size or by carefully selecting the vocabulary elements. Such a careful selection can be achieved with a semantic vocabulary [14, 136, 82, 146] that describes an image in meaningful codewords. A semantic vocabulary can be constructed by manually selecting image patches with meaningful labels, for example *sky, water* or *vegetation.* The idea of meaningful codewords, is that they allow a compact, discriminative, and semantic image representation. In contrast to annotating a vocabulary, Jurie and Triggs [62] compare clustering techniques to obtain a vocabulary. Specifically, they show that radius-based clustering outperforms the popular k-means clustering algorithm. Furthermore, Winn *et al.* [150] concentrate on a global codebook vocabulary, whereas Perronnin *et al.* [95] focus on concept-specific vocabularies.

In this Chapter we concentrate on compact vocabulary construction while trying to retain the ability to discriminate well between concept categories. Note that this is more general than vocabularies that are built by a discriminative criterion [84]. Such methods assume that the discriminative ability of a single feature carries over to the whole vocabulary. Hence, a vocabulary created by discriminative criteria of single features also aims at a final vocabulary which is discriminative between concept categories.

Instead of reducing the size of a vocabulary, the expressive power of the vocabulary may be increased. With higher expressive power, a vocabulary needs less codewords to obtain similar performance which in turn leads to a more compact vocabulary. The expressive power can be increased by disposing of the hard-assignment of a single codeword to a single image features. Instead of using hard-assignment, some weight may be given to related codewords. To this end, Tuytelaars and Schmid [128] and Jiang *et al.* [61] assign weights to neighboring visual words. Whereas a visual word weighting scheme based on feature similarity is used in Agarwal and Triggs [1] and in our previous work [136, 140]. This soft-assignment increases the expressiveness of a vocabulary. We will test the influence of soft-assignment on vocabulary compactness. In the next section we will present the details of the method.

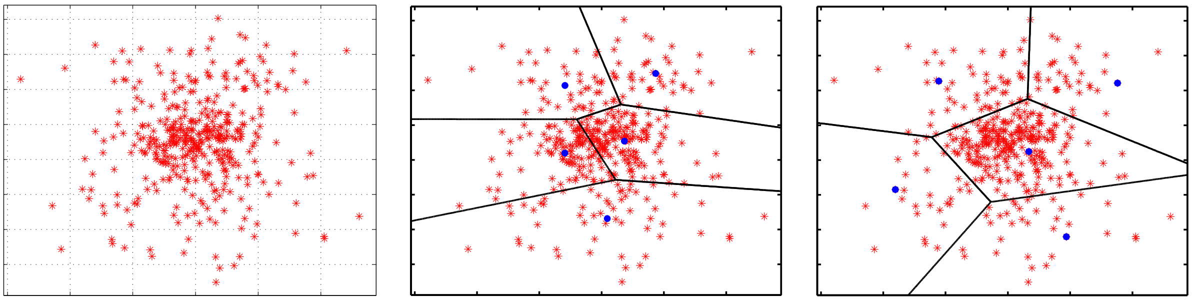
4.3 Compact Codebook Models

In the codebook model, the vocabulary plays a central role. The expressive power of the vocabulary determines the quality of the model, whereas the size of the vocabulary controls the complexity of the model. Therefore, vocabulary construction directly influences model complexity. We identify two methods for constructing a vocabulary: a data-driven approach characterized by unsupervised clustering and a semantic approach which relies on annotation. Besides the construction of the vocabulary, the expressive power may be increased. To this end, we consider replacing the hard-assignment of codewords to image features with soft-assignment. This soft-assignment aims for a more powerful vocabulary, which in turn leads to a more compact model.

4.3.1 Codebook Compactness by a Clustered Vocabulary

A codebook vocabulary consists of discrete visual codewords, which are described by high-dimensional features. In order to obtain discrete codewords, the continuous high-dimensional feature space needs to be discretized. A common approach to discretizing a continuous feature space is by uniform histogram binning. However, in a high-dimensional feature space a histogram with a fixed bin size for each dimension will create an exponentially large number of bins. Moreover, since feature spaces are rarely uniformly distributed, many of these bins will be empty [128]. We illustrate the partitioning of a continuous feature space with a uniform histogram in figure 4.4(a).

An alternative to a uniform partitioning of the high-dimensional feature space is unsupervised clustering. The benefit of using clusters as codewords is a small vocabulary size without empty bins. A popular clustering approach for finding codewords is k-means [13, 33, 62, 67, 95, 100, 125]. K-means is an unsupervised clustering algorithm that tries to minimize the variance between k clusters and the training data, where k is a parameter of the algorithm. The advantages of k-



(a) Histogram (b) K-means (c) Radius-based

Figure 4.4: *Three examples of continuous space partitioning, using (a) a uniform histogram, (b) k-means clustering, and (c) radius-based clustering. Note the empty bins in the histogram, the cluster centers in densely populated areas of k-means, and the uniform partitioning of radius-based clustering.*

means are its simple and efficient implementation. However, the disadvantage of k-means is that the algorithm is variance-based. Thus, the algorithm will award more clusters to high-frequency areas of the feature space, leaving less clusters for the remaining areas. Since frequently occurring features are not necessarily informative, this over-sampling of dense regions is inappropriate. For example, in analogy of text retrieval, the most frequent occurring words in English are the so called function words like *the*, *a*, and *it*, despite their high frequency these function words convey little information about the content of a document. Therefore a codebook vocabulary based on variance-based clustering may not be as expressive as it could be.

In contrast to variance-based clustering, Jurie and Triggs [62] argue that the codewords for a codebook vocabulary are better represented by radius-based clustering. Radius-based clustering assigns all features within a fixed radius of similarity r to one cluster, where r is a parameter of the algorithm. This radius denotes the maximum threshold between features that may be considered similar. As such, the radius determines whether two patches describe the same codeword. Hence, the influence of the radius parameter r on the codebook model is clear where the number of clusters, k, in k-means clustering is typically chosen arbitrary. The difference between radius-based clustering and k-means is illustrated in figure 4.4(b) and figure 4.4(c). Note that the codewords found by k-means populate the densest part of the feature space, whereas the radius-based method finds codewords that each represent a distinct part of the feature space. Hence, radius-based clustering results in a non-empty, uniform sampling of a continuous feature space. Therefore, we will adopt radius-based clustering for data-driven codebook vocabulary creation.

Concept-specific Vocabulary

A vocabulary formed by unsupervised clustering offers us the opportunity to construct a different, tuned, vocabulary for each concept [67, 95]. This tuning endows each concept with its own unique vocabulary. For example, it might be beneficial to model the concept *boat* with a different vocabulary than the concept *office*, since scenes with a boat will contain water and sky, whereas office scenes hold tables and chairs. The idea behind concept-specific vocabularies is to obtain a reduced vocabulary, while retaining expressive power. We will experimentally compare the compactness and expressiveness of the concept-specific vocabularies against a global vocabulary obtained by clustering the whole feature space.

4.3.2 Codebook Compactness by a Semantic Vocabulary

Whereas the previous section described a clustering approach for obtaining a codebook vocabulary, this section will focus on a semantic vocabulary. The use of semantic codewords builds on the principle of compositionality, stating that the meaning of an image can be derived from the meaning of the constituent parts of the image [14, 82, 136, 146]. For example, an *outdoor* image is likely to contain *vegetation, water*, or *sky.* A semantic vocabulary consists of meaningful codewords. Therefore, the creation of the vocabulary requires a human annotator. This annotation step typically consists of drawing bounding boxes around a meaningful patch of pixels [136, 146]. The rationale behind meaningful codewords is that local image semantics will propagate to the global codebook image model, leading to compact visual models

Both the semantic vocabulary and the clustered vocabulary have specific advantages and disadvantages. The semantic vocabulary approach is based on manual selection of visually meaningful codewords. However, this approach has the underlying assumption that images can be decomposed in these semantic codewords, which may not hold for all images. For example, an *indoor* image is unlikely to contain any *sky* or *buildings.* In contrast to semantic labeling, clustering uses statistics to determine descriptive codewords. However, these codewords lack any meaningful interpretation. Such an interpretation may be important since humans typically decompose complex scenes into meaningful elements. Both approaches of acquiring a vocabulary of low-level descriptors have their merits. We will experimentally compare both methods to determine their compactness and expressiveness.

4.3.3 Codebook Compactness by Soft-Assignment

In order to take the continuous nature of image patches into account, we have proposed [136] to base the codebook model on a degree of similarity between patches. Similarity between patches is a more suitable representation than assigning only one visual word to an image patch. Labeling an image patch with the single best visual word ignores all ambiguity regarding the meaning of the image patch. In contrast, assigning a degree of similarity to an image patch will model the inherent uncertainty of the image patch. For example, instead of labeling a blue pixel patch as *sky*, the patch is better represented by saying that its similarity to *sky* is 0.9, and its similarity to *water* is 0.8. By using soft-assignment to model the uncertainty of the meaning of an image patch, we foresee improved expressive and discriminative power while maintaining a constant vocabulary size [136]. To evaluate this claim we will test soft-assignment versus hard-assignment as used in the traditional codebook model. If this claim is sound, the vocabulary size may be reduced, which in turn yields a more compact codebook.

Soft-assignment is easily incorporated in the codebook model. For each codeword, or bin, b in the vocabulary V the traditional codebook model constructs the distribution of codewords over an image by

$$H(b) = \sum_{r \in R(im)} \begin{cases} 1 & \text{if } b = \arg\max_{v \in V}(S(v,r)). \\ 0 & \text{otherwise,} \end{cases} \tag{4.1}$$

Here, $R(im)$ denotes the set of regions in image im, and $S(v,r)$ is the similarity between a codeword v and region r. The similarity $S(b,r)$ is specific to the type of image features that are used. The similarities are given with the image features in appendix 4.A. The similarities allow replacing hard-assignment with soft-assignment by

$$H(b) = \sum_{r \in R(im)} S(b,r). \tag{4.2}$$

This soft-assignment weights each codeword according to the similarity of an image region to this codeword. Figure 4.5 illustrates this advantage.

4.4 Experimental Setup

The experiments focus on the relation between codebook compactness and codebook quality. Codebook compactness is given by the size of the vocabulary, whereas codebook quality is measured by its categorization performance. To reduce dependency on a single visual feature, we show results

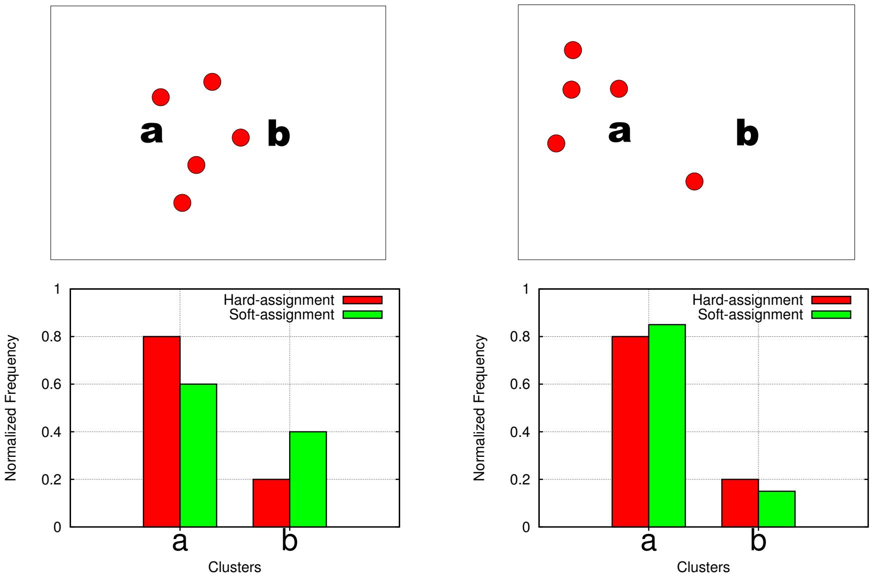


Figure 4.5: *Two examples indicating the difference between hard-assignment and soft-assignment of codewords to image features. The first row shows two images with each 5 samples (dots) around two codewords 'a' and 'b'. The second row displays the normalized occurence histograms of hard-assignment and soft-assignment for both images. Note that hard-assignment is identical for both examples, whereas soft-assignment is sensitive to the position of the samples.*

over two visual features (Wiccest features and Gabor features, see appendix 4.A). Furthermore, we investigate the effect of the linear and light-weight Fisher classifier against a computationally more intensive non-linear SVM classifier. We identify three experiments:

- **Experiment 1:** Soft-Assignment versus Hard-Assignment;
- **Experiment 2:** Semantic Vocabulary versus Globally-clustered Vocabulary;
- **Experiment 3:** Semantic Vocabulary versus Concept-specific clustered Vocabulary;

The experiments are conduced on a large video dataset where each shot is annotated if a concept is present. This fixed ground-truth allows repeatable experiments.

4.4.1 Video Datasets

The experiments are evaluated on the TREVID 2005 development set [115]. This video set contains nearly 85 hours of English, Chinese and Arabic news video. In addition to the video data, we use the standard ground truth provided by the MediaMill Challenge [121]. This ground truth defines 101 semantic concepts with shot labels for each category, where the video data is split in 70% for training, and the remaining 30% for testing. In total there are 43,907 shots, where 30,630 are in the training set, and 13,277 in the testing set. The shots are indexed by their representative keyframe, as defined by the MediaMill Challenge. We selected the MediaMill Challenge because it is a realistic and challenging dataset with a shared ground truth, allowing repeatable experiments. In figure 4.6 we show some concepts defined by the MediaMill Challenge. Note the wide variety of concepts, i.e.: Graphics (*Drawing, Maps, Weather*), objects (*Bird, Chair, Flag USA*), scenes (*Duo-anchor, Meeting, Night fire, River, Sky, splitscreen, Studio, Tennis*), persons (*Anchor, Mr. Lahoud, Prisoner*), and emotional (*Entertainment*). The video data is a realistic subset of broadcast news, containing commercials, e.g. (*Bird, River*), and concepts with little variation in their appearance for this set, e.g. (*Night fire, Tennis, Chair, Weather, Anchor*). In contrast to simplified datasets recorded in a laboratory setting [85], the MediaMill Challenge allows a more truthful extrapolation of our conclusions to other real-world datasets.

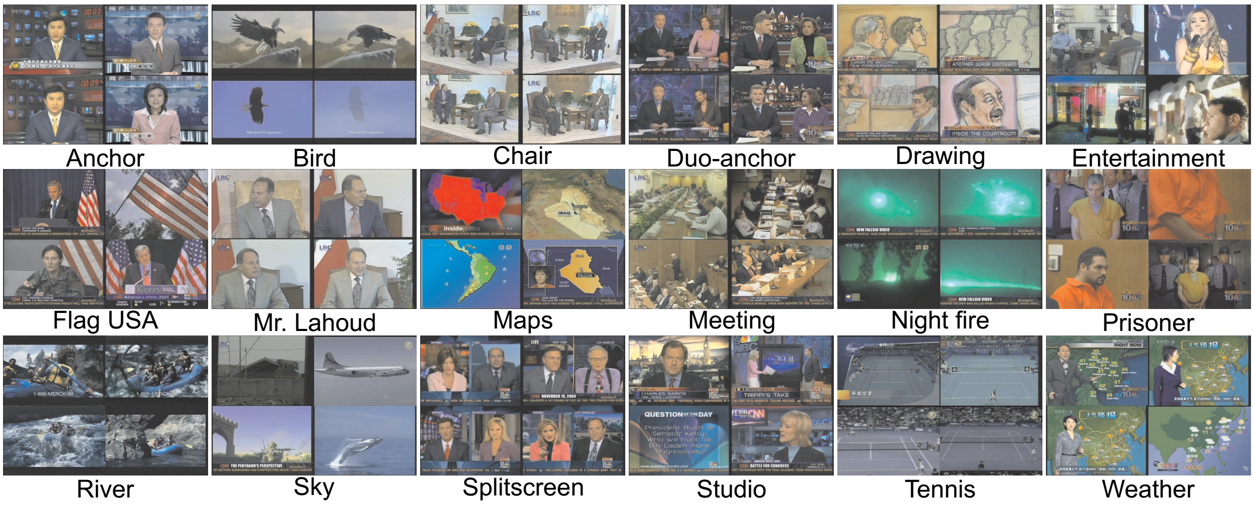


Figure 4.6: *Some examples of the concepts defined by the MediaMill Challenge, which we use to evaluate categorization performance.*

4.4.2 Visual Categorization Implementation

Image Features

To evaluate if a method generalizes over visual features, we conduct all experiments with two different image features: Wiccest and Gabor. Wiccest features rely on natural image statistics which makes them well suited to describe natural images. On the other hand, Gabor features respond to regular textures and color planes, which is beneficial for man-made structures. Both these image features measure colored texture, where the Gabor features also takes non-textured color into account. Each feature is calculated on two scales, making them sensitive to differently scaled textures. We selected texture features because of their ability to describe the foreground as well as the contextual background of an image. More details about the image features are in appendix 4.A.

Image Sampling

The codebook model represent an image as a distribution over codewords. To build this distribution, several regions are sampled from an image. Since grid-based sampling is shown to outperform interest points in scene categorization [33, 62], we use a grid for region sampling. Specifically, this grid is constructed by dividing an image in several overlapping rectangular regions. The regions are uniformly sampled across the image, with a step size of half a region. We use two different region sizes, with ratios of $\frac{1}{2}$ and $\frac{1}{6}$ of both the x-dimension and y-dimensions of the image.

4.4.3 Compact Codebook Models Implementation

Semantic Vocabulary

A semantic vocabulary consists of meaningful elements, obtained by annotation. We use the semantic vocabulary by [136]. This vocabulary consists of 15 different codewords, namely: building (321), car (192), charts (52), crowd (270), sand/rock (82), fire (67), flag USA (98), maps (44), mountain (41), road (143), sky (291), smoke (64), snow (24), vegetation (242), water (108), where the number in brackets indicates the number of annotation samples of that concept. We use the train set as a basis for selecting relevant shots containing the codewords. In those shots, we annotate rectangular regions where the codeword is visible for at least 20 frames. Note that a vocabulary of 15 codewords, evaluated for two scales and two region sizes will yield a descriptor of $4 \times 15 = 60$ elements.

Globally-clustered Vocabulary

A globally-clustered vocabulary is created on all image features in the train set. We build a such a global vocabulary by radius-based clustering. Radius-based clustering aims to cover the feature space with clusters of a fixed similarity radius. Hence, the algorithm yields an even distribution of visual words over the feature space and has been shown to outperform the popular k-means algorithm [62]. Whereas Jurie and Triggs [62] use mean-shift with a Gaussian kernel to find the densest-point, we maximize the number of features within its radius r for efficiency reasons.

Since each image features is calculated at two scales for two region sizes there are 4 image descriptors per feature. We cluster each descriptor separately, yielding 4 different clustering steps. The final vocabulary consists of the resulting clusters for a single radius as found by all these four clustering steps. Note that the number of clusters may vary per scale and region size combination.

Concept-Specific Clustered Vocabulary

A concept-specific vocabulary is designed for a single concept. Such a specific vocabulary may be found by limiting the radius-based clustering algorithm to images in a single class only. This makes the resulting clusters depend on only that subset of the feature space which is relevant for the concept. Note that the images are labeled globally, whereas the clustering is based on local codewords. The clustering step itself is identical to the globally-clustered vocabulary, and is performed separately for each of the four feature scale and region size combinations.

4.4.4 Supervised Machine Learning Implementation

Automatic concept categorization in video requires machine learning techniques. For each semantic concept, we aim for a ranking of shots relevant to this concept. To evaluate this ranking, we employ two classifiers: a strong and computationally intensive SVM classifier and a weak but fast Fisher classifier. Fisher's linear discriminant [35] projects high-dimensional features to a one-dimensional line that aims to maximize class separation.The most important reason why we use Fisher's linear discriminant is its fair categorization performance with high efficiency. This efficiency is mostly due to its linearity and the benefit that this classifier has no parameters to tune. The other classifier is the popular discriminative maximum-margin SVM classifier. The reason for choosing an SVM is because it generally gives good results on this type of data [121]. For the SVM we use a non-linear χ^2 kernel, where we use episode constrained cross-validation [139] to tune the best C-slack parameter.

4.4.5 Evaluation Criteria

We evaluate compactness and categorization performance. Compactness is measured in by the size of the codebook vocabulary. For measuring categorization performance, we adopt average precision from the Challenge framework. Average precision is a single-valued measure that summarizes the recall-precision curve. If $L_k = \{s_1, s_2, \ldots, s_k\}$ are the top k ranked elements from the retrieved results set L, and let R denote the set of all relevant items, then average precision (AP) is defined as

$$\mathrm{AP}(L) = \frac{1}{|R|} \sum_{k=1}^{|L|} \frac{|L_k \cap R|}{k} I_R(s_k) \ , \tag{4.3}$$

where $|\cdot|$ denotes set cardinality and $I_R(s_k) = 1$ if $s_k \in R$ and 0 otherwise. In our experiments we compute AP over the whole result set.

Average precision measures the categorization performance for a single concept. The MediaMill Challenge, however, defines 101 concepts. As the performance measure over multiple concepts, we report the mean average precision (MAP), given by the average precision averaged over all concepts.

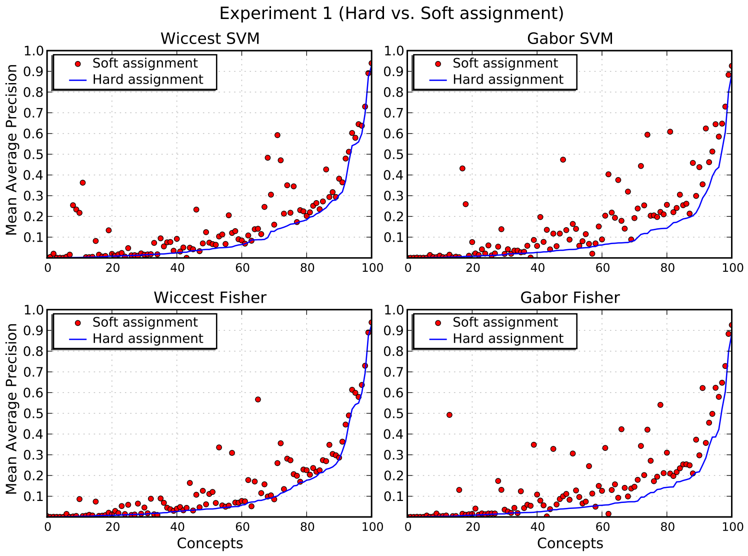


Figure 4.7: *Comparing hard-assignment versus soft-assignment for all 101 concepts, over two different visual features with a semantic vocabulary.*

	Experiment 1			
	Wiccest		Gabor	
	SVM	Fisher	SVM	Fisher
Hard-Assignment	0.120	0.113	0.100	0.097
Soft-Assignment	0.179	0.157	0.187	0.175

Table 4.1: *The mean average precision over all 101 concepts in experiment 1. Results are shown for hard-assignment versus soft-assignment for Wiccest features and Gabor features and the Fisher and SVM classifier, using a semantic vocabulary. Note that soft-assignment outperforms hard-assignment for both feature types and for both classifiers.*

4.5 Experimental Results

4.5.1 Experiment 1: Soft-Assignment vs. Hard-Assignment

The first experiment compares soft-assignment with hard-assignment in the codebook model for a semantic vocabulary over two classifiers and over the two visual features. In appendix 4.A we detail both features and their respective soft assignment functions. In figure 4.7 we show the results for the Wiccest and Gabor features. The figure illustrates that performance for nearly all concepts improves by using soft-assignment. This improvement is in line with the expectations in [136]. In the few cases where soft-assignment is outperformed by hard-assignment, the performance difference is marginal. On average over the two features and two classifiers there are 92 ± 2.71 concepts that increase and 8.75 ± 2.87 concepts that decrease. Over both features and both classifiers there are 78 of the 101 concepts that always improve. In contrast, there is no concept whose performance always decreases. For the four feature-classifier combinations, there are 28 concepts that decrease in performance for at least one of these combinations. Note that this is the absolute worst-case performance. In contrast, all 101 concepts are found to increase at least once or more in the four feature-classifier combinations. The average performance over all 101 concepts for the two visual features is shown in table 4.1. The table shows that using soft-assignment improves performance for both feature types and for both classifiers.

The difference per concept between soft-assignment and hard-assignment is given in figure 4.8. Here we show the five most increasing concepts and the five most decreasing concepts by replacing

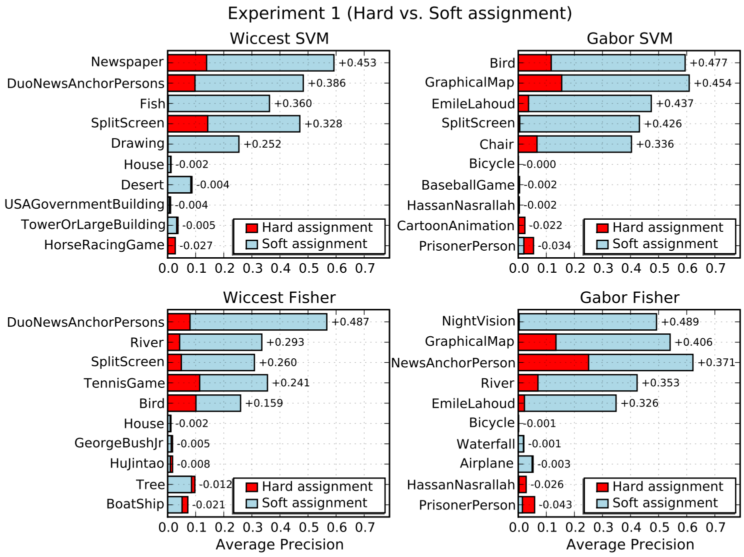


Figure 4.8: *The difference between soft-assignment and hard-assignment for the top and bottom five concepts in experiment 1.*

hard-assignment with soft-assignment. Note that the performance gain by the improving concepts is several magnitudes higher than the decrease in performance. There are four concepts that consistently decrease in the bottom five. The concepts *PrisonerPerson, HassanNasrallah, Bicycle* are found in the bottom five of the Gabor features for both the Fisher as the SVM classifier. These concepts are sensitive to exact color matching. The *Bicycle* concept is a sparse but repetitive commercial, and the *PrisonerPerson, HassanNasrallah* concepts contain shots of highly discriminative colors, like an orange prisoner uniform. Since the gabor features take the color of an image patch into account, these features are more effected than the Wiccest features. The six concepts *Bird, River, DuoNewsAnchorPersons, GraphicalMap, EmileLahoud, SplitScreen* consistently increase in the top five. Of these six concepts the concepts *GraphicalMap* and *EmileLahoud* are found in the Gabor features top 5 for both the Fisher as the SVM classifier. In this case the concepts are again typically colorful, such as the many variations of a *GraphicalMap*, or a colorful flag in the background of Mr. *EmileLahoud*. In this case, however, performance increases. We deem that this is the case because there is significant variation in the colors. By using soft-assignment this variation is better modeled. The concept *DuoNewsAnchorPersons* increases for the Wiccest features in both the SVM as in the Fisher classifier. Again, we attribute the gain of soft-assignment to slight variation between the examples. With slight variation in the images, hard assignment may choose complete different visual words, whereas soft-assignment proves robust. The concept *SplitScreen* is found in the top five of three feature-classifier combinations. Only the Gabor-Fisher does not have this concept in the top five. This concept is characterized by a strong artificial edge in the middle of the screen. Besides this edge, there is some variation on the people present in the screens. Again, soft-assignment seems to be able do deal better with this variation. The concept *Bird* improves for Wiccest-Fisher and for Gabor-SVM. This concept is a repetitive commercial. We attribute the reason why static or near-copies benefit most to the fact that minor changes in the image content results in minor changes in the soft-assignment approach. In contrast, minor image content changes in the traditional codebook model may give rise to completely different codewords stemming from the hard-assignment in this method. In figure 4.6 we show example images for some concepts.

4.5.2 Experiment 2: Semantic Vocabulary vs. Globally-clustered Vocabulary

As a second experiment, we focus on the difference between a semantic vocabulary and a clustered vocabulary. In figure 4.9 we show the results with hard-assignment and soft-assignment over the two features and over the two classifiers. This figure shows that increasing the number of visual words

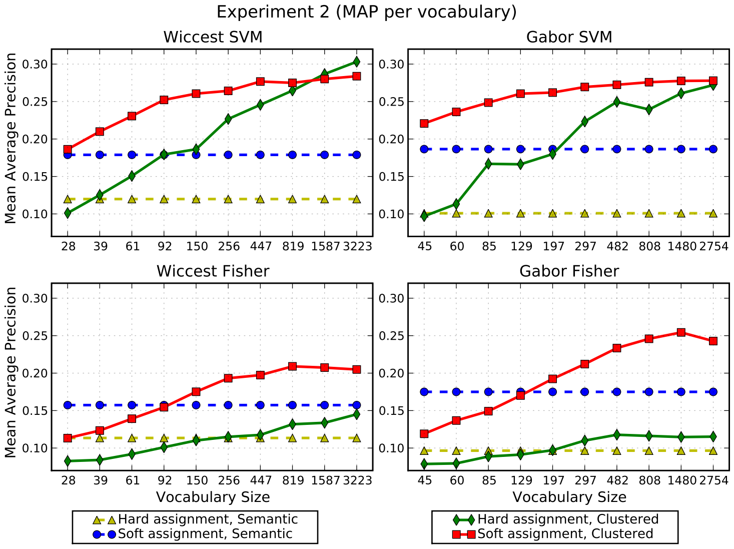


Figure 4.9: *Comparing a semantic codebook vocabulary with a globally-clustered codebook vocabulary for hard-assignment and soft-assignment. Results are shown in mean average precision over 101 concepts. The semantic vocabulary is the same as in experiment 1. Note that the Wiccest and the Gabor features have different vocabulary sizes. This is the case, because the number of clusters depends on the similarity function of the visual features (see appendix 4.A).*

increases the performance. Moreover, the figure shows a clear advantage of using an SVM classifier over the Fisher classifier. Nevertheless, for Gabor features with a vocabulary of 1480 codewords the Fisher classifier proves competitive to an SVM classifier. Note that a larger vocabulary not always yields the best results. For example, for the Fisher classifier with soft-assignment, the largest vocabulary is not the best performing one. Furthermore, the figure shows that for Wiccest features and a Fisher classifier the performance difference between a semantic and a clustered vocabulary is only slightly in favor of the semantic vocabulary when both vocabularies have an equal number of visual words (±60). In contrast, for Gabor features a semantic vocabulary is more beneficial, yielding a higher performance for a lower number of codewords. We credit this difference between the Wiccest and the Gabor features to the difference in dimensionality between the features. The Wiccest features use only 12 numbers, whereas the Gabor features consist of histograms of 101 bins. Since the feature-space of the Gabor descriptor is much higher in dimensionality, it is harder to fill this space, let alone find discriminative visual words. In contrast to clustering, a semantic vocabulary is given by manual annotation. This annotation step introduces meaningful visual words without the need to partition a high-dimensional feature space. Nevertheless, a fixed sized semantic vocabulary is outperformed by a clustered vocabulary for both features. This performance gain comes at a price, paid by an exponentially growing visual word vocabulary, leading to a more complex, and therefore less compact model. Comparing the results of a semantic vocabulary and a clustered vocabulary for the SVM classifier, shows a clear advantage for a clustered vocabulary. The clustered vocabulary already outperforms a semantic vocabulary with half the number of codewords in the case of Wiccest features. Moreover, for the Wiccest features the hard-assignment method outperforms the soft-assignment method for large vocabularies. In the case of the Gabor features, the hard-assignment performance equal to soft-assignment for large vocabularies. Nevertheless, for an SVM classifier, soft-assignment proves robust over the size of the vocabulary. Soft-assignment clearly outperforms hard-assignment for compact vocabularies.

In figure 4.10 we show per concept the vocabulary size which gives the best performance. Moreover, we show the contours of the areas that perform within 90% of the best score. When comparing soft-assignment versus hard-assignment, it can be seen that for soft-assignment there are more areas where the performance is within 90% of the best score. Hence, soft-assignment seems

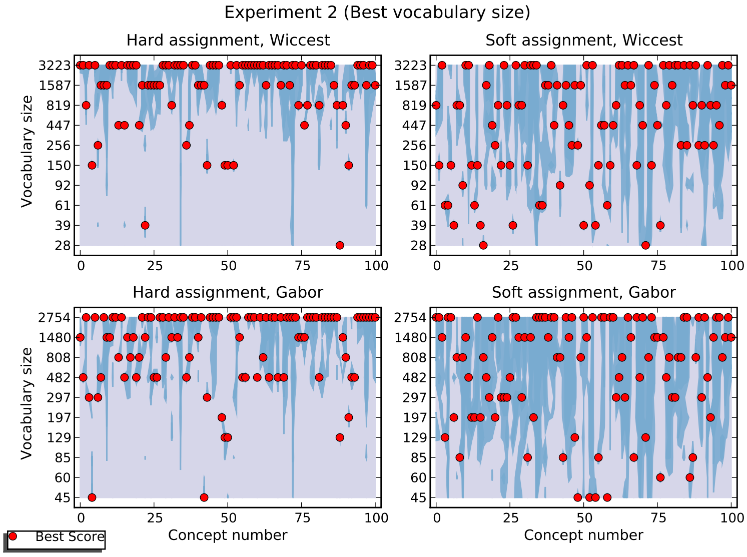


Figure 4.10: *The red dots indicate the best performing vocabulary size for each concept. The contours highlight the area within 90% of the best performance.*

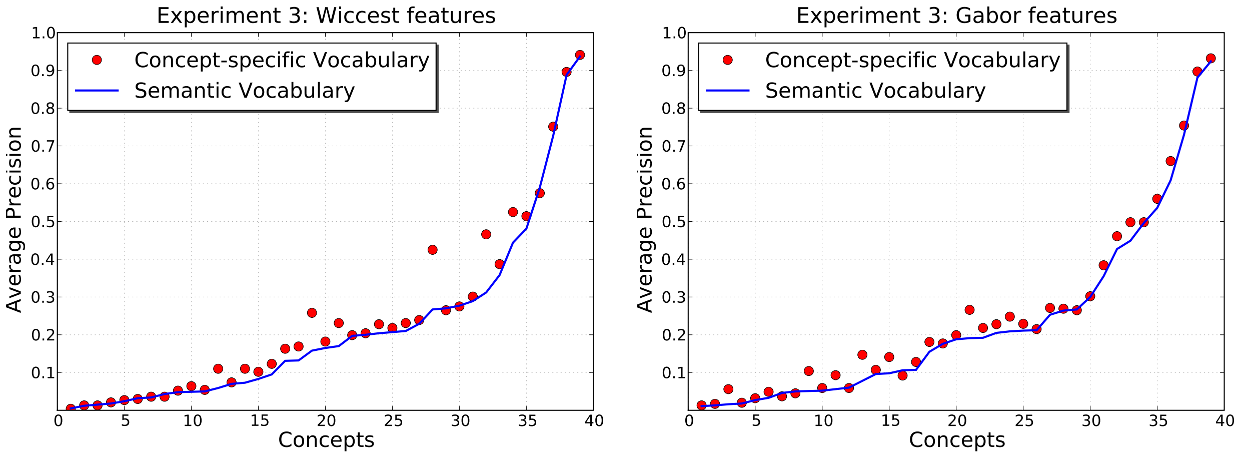


Figure 4.11: *Comparing a semantic vocabulary with a concept-specific vocabulary, both using soft-assignment.*

more robust to the size of the vocabulary. Furthermore, the figure shows that soft- assignment has more variation in the size of the best vocabulary than hard-assignment. Hence, soft-assignment seems the better choice for compact vocabularies. Moreover, as the variation in the size of the best vocabulary suggests, it may prove beneficial to tune a vocabulary per concept, instead of using a global vocabulary. This tuning per concept is explored in the next section.

4.5.3 Experiment 3: Semantic Vocabulary vs. Concept-specific Clustered Vocabulary

In an attempt to create more compact vocabularies while keeping performance on par, we evaluate individual vocabularies that are tuned to the specific concept at hand. These concept-specific vocabularies are created by restricting the radius-based clustering algorithm to the positive examples of a semantic concept. To constrain the computations, we limit this experiment to the Fisher classifier only and to the 39 concepts that were used in the TRECVID 2006 benchmark. Moreover, we select a fixed radius for the clustering algorithm: $r = 1.2$ for the Wiccest features and $r = 4.5$ for the Gabor features. These radii are selected with the intention to closely match the performance

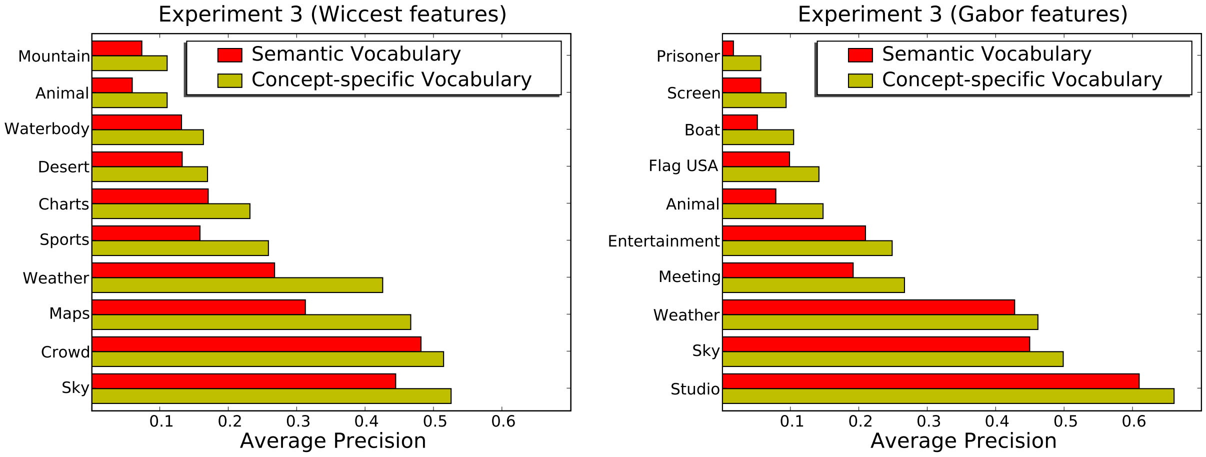


Figure 4.12: *The 10 concepts that benefit most from a concept-specific vocabulary over a semantic vocabulary.*

	Experiment 1 Semantic		Experiment 2 Clustered		Experiment 3 Concept-Specific	
Feature	Size	MAP	Size	MAP	Size	MAP
Wiccest	60	0.219	205	0.251	128.7	0.244
Gabor	60	0.235	249	0.270	118.5	0.254

Table 4.2: *The number of codewords used to obtain the same performance over three types of vocabularies: semantic (Experiment 1), clustered (Experiment 2), and concept-specific (Experiment 3). The size of the codeword vocabulary is shown, with the mean average precision in brackets for Wiccest features and Gabor features using soft-assignment. In the case of the concept-specific vocabulary, we show the average number of codewords, since this varies per concept.*

of the semantic vocabulary.

The performance differences between the semantic vocabulary and the concept-specific vocabularies for the Wiccest and Gabor features using soft-assignment are shown in figure 4.11. Note that the performance of both methods is closely aligned. Nevertheless, there are a few concepts that perform better with a concept-specific vocabulary. The top ten of the concepts that increase most are shown in figure 4.12. Some video frames containing these concepts are shown in figure 4.6. In the top ten, there are three concepts (*animal, weather, sky*) that increase for both features. The other features that improve per visual feature seem related to the feature type. The Wiccest features are related to edge statistics as found in natural images, and the concepts that improve are related to natural scenes (*animal, mountain, waterbody, desert, sports, sky, crowd*). Furthermore, it is striking that seven concepts out of the top ten for the Wiccest features consist of elements that are also used in the semantic vocabulary (*mountain, waterbody, desert, charts, maps, sky, crowd*). We speculate that this is the case because the improved concepts for the Wiccest features focus on natural images, and the semantic vocabulary consists mainly of naturally occurring codewords. In the case of Gabor features, that are more related to color and texture frequency, the concepts that improve may rely on colored texture for discrimination (*prisoner, flag USA, meeting, entertainment, weather, studio*). Nevertheless, disregarding those few outliers who outperform the semantic vocabulary, both vocabulary types perform more or less equal, as intended.

In table 4.2 we show the number of codewords used to achieve more or less the same performance. The number of codewords for the concept-specific vocabulary was found by increasing the radius of the clustering algorithm, until the performance of the concept-specific clustered vocabulary was reached. The results show that an annotated vocabulary has the most compact descriptor, with only 60 visual words. In contrast, the globally-clustered vocabulary requires at least three times more

Method	Manual	Computational		Compact		Performance	
		Strong	Weak	Strong	Weak	Strong	Weak
Semantic	−	±	+	−	+	−	±
Globally clustered	+	−	±	+	−	+	+
Concept-specific clustered	+	−	−	+	+	+	+

Table 4.3: *Summary of the four evaluated methods to obtain a compact and expressive codebook. We indicate if a method requires manual annotation effort, computation effort, and if the method yields compact models, with good performance. We distinguish between a strong classifier such as an SVM and a weak classifier such as Fisher's linear discriminant. A + denotes* ***good****, − indicates* ***bad****, and ± is* ***medium****. Note that soft-assignment is performed after vocabulary creation, thus it is not affected by annotation nor clustering.*

visual words than a semantic vocabulary. The individually clustered concept-specific vocabularies require two times the number of codewords than a semantic vocabulary. However, those concept-specific vocabularies are still only half the size of a globally clustered vocabulary. Hence, while a semantic vocabulary proves the most descriptive, the concept-specific clustered vocabularies yield a more powerful descriptor than a globally clustered vocabulary.

4.5.4 Summary of Experimental Results

We summarize the results in table 4.3. The first observation we can make is that soft-assignment typically outperforms hard-assignment in the codebook method. This improvement has been shown for two different visual features and for both a semantic vocabulary and a clustered vocabulary over two classifiers. Only for a very large vocabulary and an SVM classifier hard-assignment may improve over soft-assignment. Furthermore, the semantic vocabulary which requires manual annotation work has been shown to provide a competitive vocabulary when a weak classifier is used. In the case of the Fisher classifier it yields excellent performance with a minimum number of visual words leading to compact and expressive codebooks. For the Fisher classifier, a clustered vocabulary outperforms a semantic vocabulary when the number of visual words is high enough. However, this high number of visual words leads to less compact models, which may be infeasible for large video datasets. In the case of a strong classifier, the results show that clustered vocabularies outperform a semantic vocabulary. However, an SVM classifier takes more effort to train, with additional complication with cross-validation for parameter tuning [139]. Additional results indicate that the number of visual words in a clustered vocabulary may be reduced by tuning this vocabulary to each concept. These tuned vocabularies retain categorization performance while maintaining a reasonably compact vocabulary.

4.6 Conclusions

Given the vast amount of visual information available today, the applicability of automatic visual indexing algorithms is constrained by their efficiency. Accordingly, this Chapter focuses on compact, and thus efficient, models for visual concept categorization. We considered the codebook algorithm where model complexity is determined by the size of the vocabulary. We structurally compared four approaches that lead to compact and expressive codebooks. Specifically, we compared three methods to create a compact vocabulary: 1) global clustering, 2) concept-specific clustering and 3) a semantic vocabulary. The fourth approach increases expressive power by soft-assignment of codewords to image features. We experimentally compared these four methods on a large and standard video collection. The results show that soft-assignment improves the expressive power of the vocabulary, leading to increased categorization performance without sacrificing vocabulary compactness. Further experiments showed that a semantic vocabulary leads to compact vocabu-

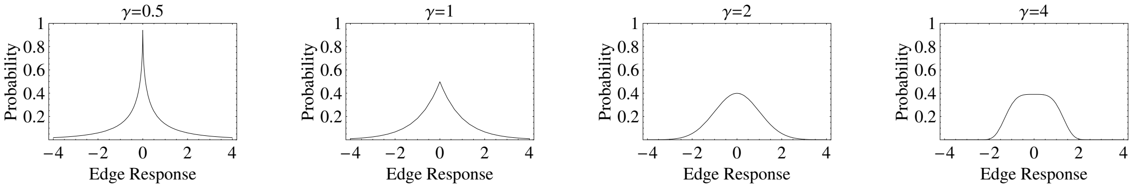


Figure 4.13: *Some examples of the integrated Weibull distribution for $\beta = 1$, $\mu = 0$, varying values for $\gamma \in \{\frac{1}{2}, 1, 2, 4\}$.*

laries, while retaining reasonable categorization performance. A concept-specific vocabulary leads to reasonable compact vocabularies, while providing fair visual categorization performance. Given these results, the best method depends at the application at hand. In this Chapter we presented a guideline for selecting a method given the size of the video dataset, the desirability of manual annotation, the amount of available computing power and the desired categorization performance.

4.A Appendix: Image Features

4.A.1 Wiccest Features

Wiccest features [40] utilize natural image statistics to effectively model texture information. Texture may be described by the distribution of edges at a certain region in an image. Hence, a histogram of a Gaussian derivative filter is used to represent the edge statistics. The histogram describes image statistics in natural textures, which are well modeled with an integrated Weibull distribution [40]. This distribution is given by

$$f(r) = \frac{\gamma}{2\gamma^{\frac{1}{\gamma}}\beta\Gamma(\frac{1}{\gamma})} \exp\left\{-\frac{1}{\gamma}\left|\frac{r-\mu}{\beta}\right|^{\gamma}\right\}, \tag{4.4}$$

where r is the edge response to the Gaussian derivative filter and $\Gamma(\cdot)$ is the complete Gamma function, $\Gamma(x) = \int_0^\infty t^{x-1}e^{-1}dt$. The parameter β denotes the width of the distribution, γ represents the 'peakness' of the distribution, and μ denotes the mode of the distribution. See figure 4.13 for examples of the integrated Weibull distribution.

The Wiccest features for an image region consist of the Weibull parameters for the illumination invariant edges in the region at $\sigma = 1$ and $\sigma = 3$ of the Gaussian filter [136]. The β and γ values for the x-edges and y-edges of the three opponent color channels normalized by the intensity [44] yields a 12-dimensional descriptor. The similarity, $S_{\mathcal{W}}$, between two Wiccest features is given by the accumulated fraction between the respective β and γ parameters,

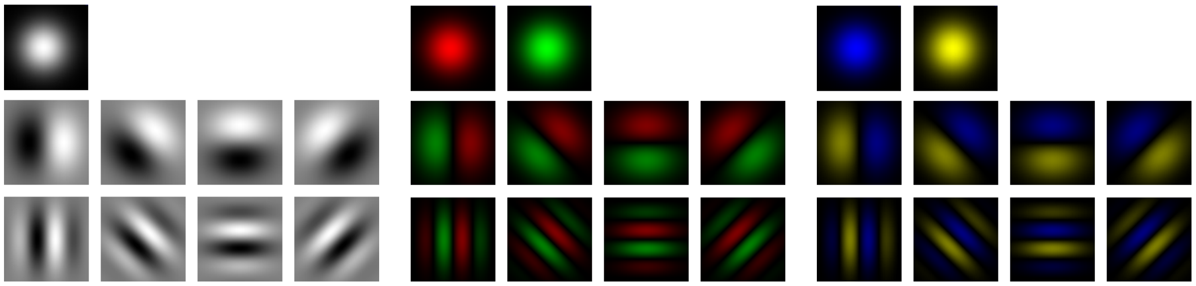
$$S_{\mathcal{W}}(F, G) = \sum \left(\frac{\min(\beta_F, \beta_G)}{\max(\beta_F, \beta_G)} \frac{\min(\gamma_F, \gamma_G)}{\max(\gamma_F, \gamma_G)} \right), \tag{4.5}$$

where F and G are Wiccest features.

4.A.2 Color Gabor Features

As an alternative to Wiccest features, one may use the popular Gabor filters. Gabor filters may be used to measure perceptual surface texture in an image [15]. Specifically, Gabor filters respond to regular patterns in a given orientation on a given scale and frequency. A 2D Gabor filter is given by

$$\widetilde{G}(x, y) = G_\sigma(x, y) \exp\left\{2\pi i \begin{pmatrix}\Omega_{x_0}\\ \Omega_{y_0}\end{pmatrix} \begin{pmatrix}x\\ y\end{pmatrix}\right\}, \quad i^2 = -1, \tag{4.6}$$



(a) Intensity channel (b) Red-Green channel (c) Blue-Yellow channel

Figure 4.14: *Some examples of the color Gabor filter with the chosen orientations, scales and frequencies.*

where $G_\sigma(x, y)$ is a Gaussian with a scale σ, $\sqrt{\Omega_{x_0}^2 + \Omega_{y_0}^2}$ is the radial center frequency and $\tan^{-1}(\frac{\Omega_{y_0}}{\Omega_{x_0}})$ the orientation. Note that a zero-frequency Gabor filter reduces to a Gaussian filter. An example of color Gabor filters is shown in figure 4.14. Illumination invariance is obtained by normalizing each Gabor filtered opponent-color channel by the intensity [55]. A histogram is constructed for each Gabor filtered color channel, where the Gabor similarity measure, $S_{\mathcal{G}}$, is given by histogram intersection,

$$S_{\mathcal{G}}(I, M) = \sum_{j=1}^{n} \min(I_j, M_j), \tag{4.7}$$

where I_j is bin j of the n-dimensional histogram of image I.

In the case of a Gabor filter, its parameters consist of orientation, scale and frequency. We follow Hoang *et al.* [55] and use four orientations, $0°, 45°, 90°, 135°$, and two fixed (scale, frequency) pairs: (2.828, 0.720), (1.414, 2.094), where we append zero frequency color to each scale. Furthermore, the histogram representation of the Gabor filters uses 101 bins for each Gabor filtered color channel.

Chapter 5

Visual Word Ambiguity[1]

Verbal descriptions of visual characteristics like a color or a texture are often ambiguous. For example, quantifying a texture with *"A predominantly smooth yellowish-red surface with a few cracks"* leaves considerable room for interpretation. One of the interpretations of the popular codebook model for automatic image classification [1, 5, 11, 13, 12, 14, 20, 33, 135, 136, 61, 62, 67, 68, 70, 74, 77, 82, 83, 93, 95, 96, 100, 132, 114, 125, 126, 128, 146, 150, 153] is that it expresses images in terms of visual words. The model represents high-dimensional image features by discrete and disjunct visual prototypes that are predefined in a vocabulary. The visual word analogy of the codebook model includes semantic modeling at the word level [14, 136, 82, 146] or at the topic level [1, 13, 14, 33, 67, 100, 125, 150]. Spatial image layout [1, 12, 14, 68, 83, 125] can be seen as modeling phrases, whereas visual vocabulary tuning [62, 67, 70, 95, 128, 150, 153] resembles modeling domain-specific terminology. These models incorporate image-specific properties within the conceptual visual word analogy. In this Chapter we introduce another aspect of the visual word analogy, namely the use of ambiguous linguistic quantifiers as *"some", "a few", "-ish", "predominantly", "much"*. Without such quantifiers to express ambiguity, the description of the aforementioned texture is reduced to *"A smooth red surface"*. We incorporate ambiguity in the codebook model by smoothly assigning continuous image features to discrete visual words. We show that ambiguity modeling leads to more expressive models that improve classification performance.

One inherent component of the codebook model is the assignment of image feature vectors to visual words in the vocabulary. Here, an important assumption is that a discrete visual word is a characteristic representative of an image feature. The continuous nature of visual appearance complicates selecting a representative visual word for an image feature. An image feature may have zero, one, or multiple candidates in the visual word vocabulary. With one candidate there is no ambiguity. Selecting a codeword from multiple realistic candidates gives rise to *visual word uncertainty*, whereas *visual word plausibility* refers to selecting a codeword without a suitable candidate in the vocabulary. Figure 5.1 illustrates these cases. Current methods assume that an image feature is well represented by its single, best representing visual word.

The contribution of this Chapter is an investigation of visual word ambiguity leading to explicit ambiguity modeling in the codebook model. We investigate the effect of ambiguity modeling on four aspects of the codebook model. First, we investigate the classification performance of various types of ambiguity modeling. Second, we look at vocabulary expressiveness by relating ambiguity modeling and the vocabulary size. Third, we consider the effect of ambiguity on the image feature dimensionality. Our fourth contribution investigates the connection between ambiguity modeling and the number of image categories. All contribution are thoroughly experimentally verified on five well-known image categorization datasets: 15 natural scenes, Caltech-101, Caltech-256, and Pascal VOC 2007/2008. Given the current drive of the state of the art to increase feature dimensionality, vocabulary size, and the number of image categories, we argue that our contributions play an important role in practical image classification.

This Chapter is organized as follows. The next section discusses the related literature on

[1] Published in *IEEE Transactions on Pattern Analysis and Machine Intelligence* [140].

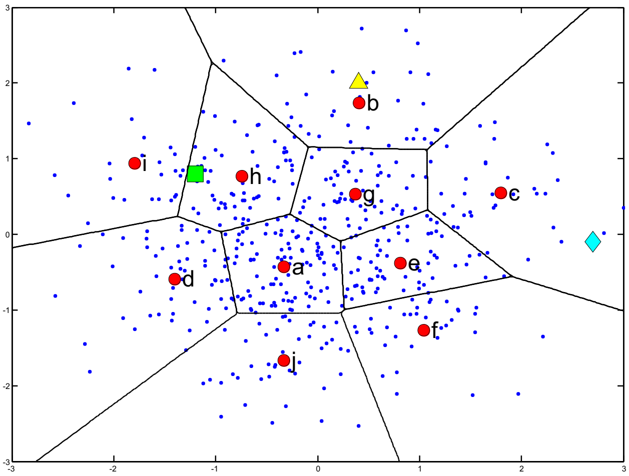


Figure 5.1: An example illustrating visual word ambiguity in the codebook model. The small dots represent image feature vectors. The labeled red circles are visual words found by unsupervised clustering. The triangle represents a data sample that is well suited to the codebook model. Visual word uncertainty is exemplified by the square, whereas visual word plausibility is illustrated by the diamond.

codebook-based scene classification. Section 5.2 introduces ambiguity in the codebook model. We show the performance and consequences of our method on five datasets in section 5.3, whereas section 5.4 concludes the Chapter. In this Chapter we use the terms *codeword* and *visual word* interchangeably.

5.1 Related Work

The visual word vocabulary in the codebook model may be constructed in various ways. Typically, a vocabulary is constructed by applying k-means clustering on image features [13, 68, 70, 74, 93, 100, 132, 114, 125, 150, 153]. K-means minimizes the variance between the clusters and the data, placing clusters near the most frequently occurring features. The most frequent features, as noted by Jurie and Triggs [62] and others [11, 100], are not necessarily the most discriminative. The discriminative power of the vocabulary may be improved by alternative clustering algorithms [62, 70], incorporating image class labels [150, 83, 153], or creating specifically tuned vocabularies for each image category as suggested by Perronnin *et al.* [95] and others [67, 128]. In contrast to clustering, a vocabulary may be obtained by manually labeling image patches with a semantic label [14, 136, 82, 146]. For example, Vogel *et al.* [146] construct a vocabulary by labeling image patches of *sky*, *water* or *grass*. The idea behind a semantic vocabulary is that the meaning of an image may be expressed in the meaning of its constituent visual words. Both the semantic and the clustered vocabulary creation methods may reduce visual word ambiguity by carefully selection the vocabulary. For example, when distinguishing a *sunset* from a *forest*, the ambiguity between the colors *pink* and *orange* is irrelevant, since both colors will be absent in a *forest*. Careful vocabulary selection, however, does not address visual word ambiguity itself.

In literature, visual word ambiguity modeling is used occasionally, often ad-hoc motivated, and rarely evaluated. Tuytelaars and Schmid [128] and Jiang *et al.* [61] assign an image feature to visual words that are neighbors in feature space. Alternatively, a probabilistic visual word voting scheme may be used [1, 5, 20, 77, 95, 96]. Here, each image feature contributes to multiple visual words relative to the posterior probability of the image feature given the visual word. Since multiple visual words are being considered, these methods cope with *visual word uncertainty*. These works recognize the importance of visual word uncertainty and show that it leads to increased classification performance. These works, however, lack a clear motivation for their type of ambiguity modeling, and ignore *visual word plausibility*. The plausibility of a visual word is employed by Boiman *et* *al.* [11] who use the distance to the single best neighbor in feature space. Their method cannot select multiple relevant visual words and therefore does not take *visual word uncertainty* into account. The uncertainty of a visual word as well as its plausibility are used by Jégou *et al.* [60]. The authors weight closer neighbors heavier than farther ones, without normalizing the scores to a posterior probability. Hence, multiple candidates can be selected, and implausible ones are given a low weight. None of these methods provide much motivation or evaluation for their choice of dealing with visual word ambiguity. In this Chapter, however, we motivate and evaluate several types of visual word ambiguity, extending our preliminary work [135] with state-of-the-art results on two additional datasets, an extensive evaluation of the vocabulary size, and ample extra analysis for all evaluated datasets.

Besides direct ambiguity modeling, ambiguity might be addressed by modeling visual word co-occurrences. Co-occurrence modeling may address ambiguity because it is likely that similar visual words with high ambiguity co-occur frequently. When these ambiguous visual words are grouped together their intra-ambiguity is resolved. Co-occurring visual word modeling is performed after assigning visual words to image features. Typically, co-occurrence is captured with a generative probabilistic model [9, 10, 56]. A generative codebook model [1, 13, 14, 33, 67, 77, 100, 125, 150] assumes that the visual words in an image are generated by underlying, latent, topics. These topics, in turn, characterize a distribution over the visual word vocabulary. With the assumption that similar visual words often co-occur, a generative model may deal with *visual word uncertainty* since similar visual words will be modeled by the same topic. Moreover, a generative model may take *visual word plausibility* into account because non-representative visual words will attain low probabilities. A generative probabilistic model, however, is dependent on large amount of visual word co-occurrence counts, or co-occurrence with the same other words, to properly model ambiguity. In contrast, directly modeling visual word ambiguity does not rely on such constraints. What is more, since a generative visual word model builds on top of visual word assignments, direct ambiguity modeling can be used as input for a generative model. In this Chapter we do not take generative models into account. A generative model on top of our ambiguity modeling would add another layer of complexity. This additional complexity complicates measuring the effect of direct ambiguity modeling. Since we are interested in measuring ambiguity, we concentrate on direct ambiguity modeling.

5.2 Visual Word Ambiguity by Kernel Codebooks

Given a vocabulary of codewords, the traditional codebook approach describes an image by a distribution over codewords. For each word w in the vocabulary V the traditional codebook model estimates the distribution of codewords in an image by

$$\mathrm{CB}(w) = \frac{1}{n}\sum_{i=1}^{n}\begin{cases} 1 & \text{if } w = \underset{v\in V}{\arg\min}(D(v, r_i)); \\ 0 & \text{otherwise,}\end{cases} \tag{5.1}$$

where n is the number of regions in an image, r_i is image region i, and $D(w, r_i)$ is the distance between a codeword w and region r_i. Typically, the regions r_i are detected interest regions, or densely sampled image patches. The codebook model represents an image by a histogram of word frequencies that describes the probability density over codewords.

A robust alternative to histograms for estimating a probability density function is kernel density estimation [113, 11]. Kernel density estimation uses a kernel function to smooth the local neighborhood of data samples. A one-dimensional estimator with kernel K and smoothing parameter σ is given by $\hat{f}(x) = \frac{1}{n}\sum_{i=1}^{n} K_\sigma\left(x - X_i\right)$, where n is the total number of samples and X_i is the value of sample i.

Kernel density estimation makes use of a kernel with a given shape and size. The kernel size determines the amount of smoothing between data samples whereas the shape of the kernel is related to the distance function between data samples [9, 143]. In this Chapter we use the SIFT descriptor

	Best Candidate	**Multiple Candidates**
Constant Weight	Traditional Codebook	Codeword Uncertainty
Kernel Weighted	Codeword Plausibility	Kernel Codebook

Table 5.1: The relationship between various forms of codeword ambiguity and their properties.

that draws on the Euclidian distance as its distance function [71]. The Euclidean distance assumes a Gaussian distribution of the SIFT features, with identity as the covariance. Hence, the Euclidian distance is paired with a Gaussian-shaped kernel $K_\sigma(x) = \frac{1}{\sqrt{2\pi}\sigma}\exp(-\frac{1}{2}\frac{x^2}{\sigma^2})$. The Gaussian kernel assumes that the variation between an image feature and a codeword is described by a normal distribution. This normal distribution has a smoothing parameter σ which represents the size of the kernel. This smoothing parameter determines the degree of similarity between data samples, and is dependent on the dataset, the feature dimensionality, and the range of the feature values. Note that we do not try to obtain the best fit of the data. In contrast, we aim to find the kernel size that discriminates best between classes. Therefore, we tune the kernel size discriminatively by cross-validation. Hence, the size of the kernel is dependent on the dataset and the image descriptor, whereas the shape of the kernel follows directly from the distance function.

In the codebook model, the histogram estimator of the codewords may be replaced by a kernel density estimator. Moreover, a suitable kernel (such as the Gaussian kernel) allows kernel density estimation to become part of the codewords, rather than the data samples. Specifically, when the kernel is symmetric, $K_\sigma(x-X_i) = K_\sigma(X_i-x)$, it trivially follows that there is no distinction between placing the kernel on the data sample or placing the kernel on a codeword. That is, if the centre of the kernel coincides with the codeword position, the kernel value at the data sample represents the same probability as if the centre of the kernel coincides with the data sample. Hence, a symmetric kernel allows for transferring the kernel from the data samples to the codewords, yielding a *kernel codebook*,

$$\mathrm{KCB}(w) \quad = \quad \frac{1}{n}\sum_{i=1}^{n} K_\sigma\left(D(w, r_i)\right), \tag{5.2}$$

where n is the number of regions in an image, r_i is image region i, $D(w, r_i)$ is the distance between a codeword w and region r_i, and σ is the smoothing parameter of kernel K. The outcome now is represented by a continuous variable rather than a discrete one.

In essence, a kernel codebook alleviates the hard mapping of features in an image region to the codeword vocabulary. This soft-assignment models two types of ambiguity between codewords: codeword uncertainty and codeword plausibility. Codeword uncertainty indicates that one image region may distribute probability mass to more than one codeword. Conversely, codeword plausibility signifies that an image feature may not be close enough to warrant representation by any relevant codeword in the vocabulary. Each of these two types of codeword ambiguity may be modeled individually. *Codeword uncertainty*,

$$\mathrm{UNC}(w) \quad = \quad \frac{1}{n}\sum_{i=1}^{n} \frac{K_\sigma\left(D\left(w, r_i\right)\right)}{\sum_{j=1}^{|V|} K_\sigma\left(D(v_j, r_i)\right)}, \tag{5.3}$$

normalizes the amount of probability mass to a total constant weight of 1 and is distributed over all relevant codewords. Relevancy is determined by the ratio of the kernel values for all codewords v in the vocabulary V. Thus, codeword uncertainty retains the ability to select multiple candidates, however does not take the plausibility of a codeword into account. In contrast, *codeword plausibility*,

$$\mathrm{PLA}(w) = \frac{1}{n}\sum_{i=1}^{n} \begin{cases} K_\sigma\left(D(w, r_i)\right) & \text{if } w = \arg\min\limits_{v \in V}(D(v, r_i)); \\ 0 & \text{otherwise,} \end{cases} \tag{5.4}$$

selects for an image region r_i the best fitting codeword w and assigns it probability mass proportional to the kernel value of that codeword. Hence, codeword plausibility will give a higher weight

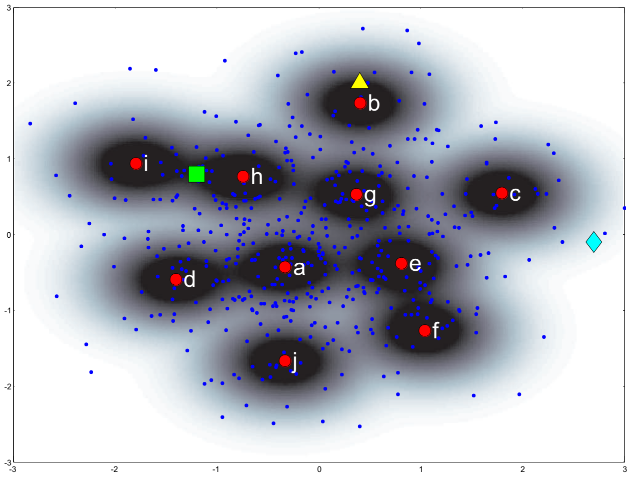


Figure 5.2: An example of the weight distribution of a kernel codebook with a Gaussian kernel, where the square, diamond and triangle represent the image features taken from figure 5.1.

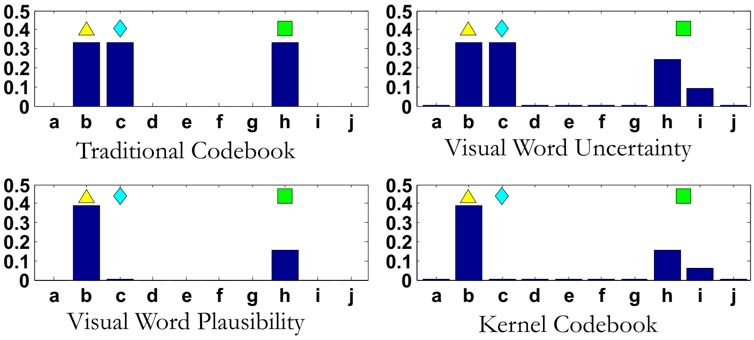
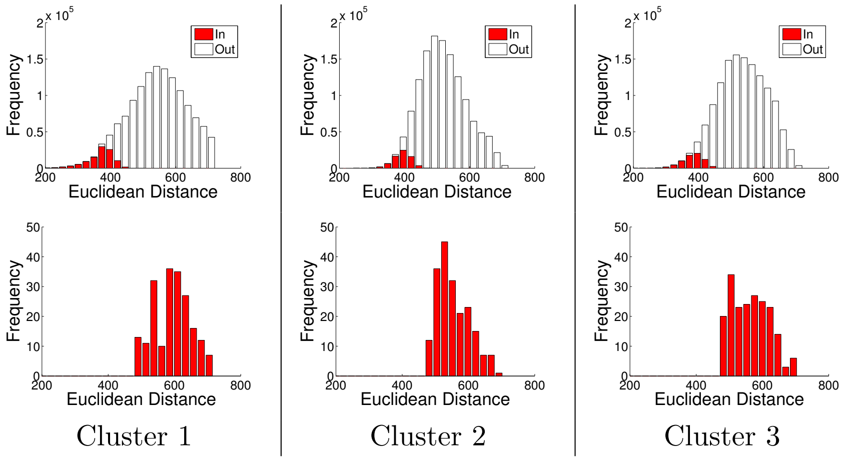


Figure 5.3: Summary of different types of codeword ambiguity, according to table 5.1. These distributions are based on the kernels shown in figure 5.2.

to more relevant data samples. However, it cannot select multiple codeword candidates. Note that the selection of a single codeword retains sparsity, which is advantageous for large datasets. The relation between codeword plausibility, codeword uncertainty, the kernel codebook model, and the traditional codebook model is summarized in table 5.1.

An example of the weight distributions of the various types of codeword ambiguity with a Gaussian kernel is shown in figure 5.2. Furthermore, in figure 5.3 we show an example of various codeword distributions corresponding to different types of codeword ambiguity. Note the weight difference in codewords for the data samples represented by the diamond and the square. Where the diamond contributes full weight in the traditional codebook, it barely adds any weight in the kernel codebook and codeword plausibility model. This may be advantageous, since it incorporates the implausibility of outliers. Furthermore, in the traditional codebook, the square adds weight to one single codeword, whereas the kernel codebook and codeword uncertainty adds weight to the two relevant codewords. In the latter two methods, the uncertainty between the two codewords is not assigned solely to the best fitting word, but divided over both codewords. Hence, the kernel codebook approach can be used to introduce various forms of ambiguity in the tradition codebook model. We will experimentally investigate the effects of all forms of codeword ambiguity in section 5.3.

In this Chapter we consider the kernel size fixed for all codewords. We have considered a variable kernel density estimator [113], where the smoothing factor σ varies per codeword. This variable smoothing factor could be determined by the variance of the image features that are assigned to each codeword by the clustering algorithm. However, varying the kernel size for each codeword yields an inhomogeneous feature space where distances are measured differently depending on their



Cluster 1 Cluster 2 Cluster 3

Figure 5.4: Histograms of Euclidean distances over 200 clusters, where each column represents a different cluster. The top row displays the distances from this cluster to all other points in the train set. The data samples that are closest to this cluster are indicated as *in* whereas the points that are assigned to other clusters are denoted *out*. The bottom row shows the distances from this cluster center to all other cluster centers.

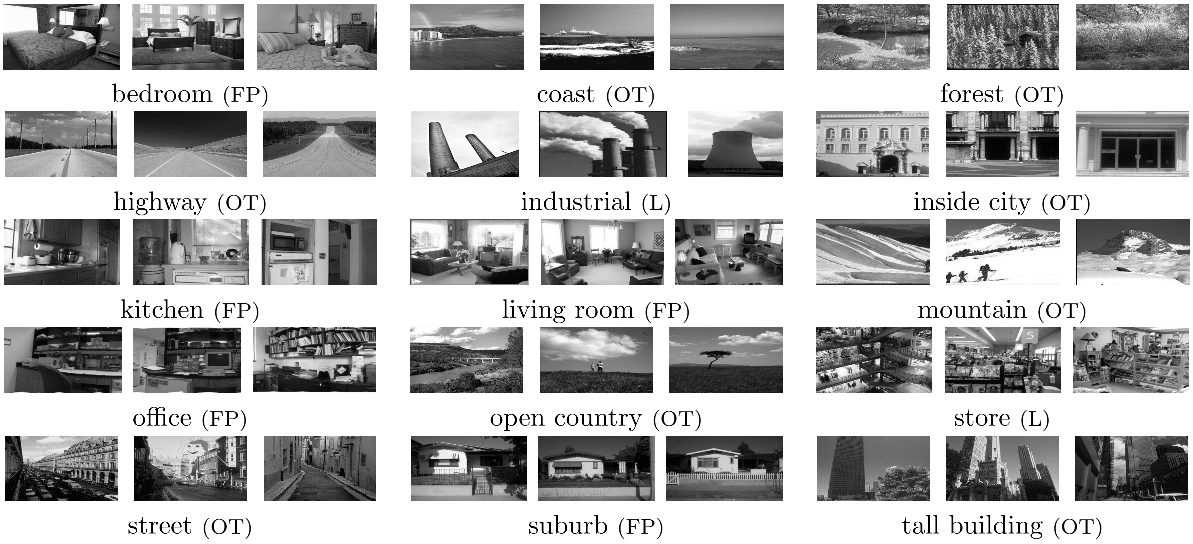
location in feature space. Essentially, a varying kernel size makes certain codewords more important than others. This difference in codeword importance may be justified, however, should be tied to the final classification performance. Since the classification performance is not taken into account by an unsupervised clustering algorithm, we adhere to a homogenous feature space by keeping the kernel size fixed for all codewords.

The ambiguity between codewords will be influenced by the number of words in the vocabulary. A large vocabulary allows a rich selection of visual words, increasing the likelihood that an image feature is well-represented. Moreover, in the case of a large vocabulary, the probability of multiple relevant visual words increases, suggesting the use of visual word uncertainty. On the other hand, when the vocabulary is small, essentially different image parts will be represented by the same vocabulary element. This misrepresentation may be alleviated by considering visual word plausibility. Hence, visual word ambiguity influences both small and large vocabularies. We extensively investigate the effect of the vocabulary size in section 5.3.

Since codewords are image descriptors in a high-dimensional feature space, we expect a relationship between codeword ambiguity and feature dimensionality. With a high-dimensional image descriptor, codeword ambiguity will become more significant. If we consider a codeword as a high-dimensional sphere in feature space, then most feature points in this sphere will lay on a thin shell near the surface. Hence, in a high-dimensional space, more feature points will be close to the boundary between codewords than in a lower-dimensional feature space. Thus, they introduce ambiguity between codewords. See Bishop's textbook on pattern recognition and machine learning [9, Chapter 1, pages 33–38] for a thorough explanation and illustration of the curse of dimensionality. Consequently, increasing the dimensionality of the image descriptor will in general increase the level of codeword ambiguity. In the next section we will experimentally investigate the effects of the dimensionality of the image descriptor.

5.3 Experiments

We experimentally compare codeword ambiguity modeling against the traditional codebook approach for five large and varied datasets: fifteen natural scene categories from Lazebnik *et al.* [68], Caltech-101 by Fei-Fei and Perona [32], Caltech-256 by Griffin *et al.* [49] and the Pascal VOC sets of 2007 [28] and 2008 [29]. We start our experiments with an in-depth analysis of our methods on the set of fifteen natural scene categories, after which we transpose these findings to the experiments on the two Caltech sets and the two issues of Pascal VOC. For our experimental setup we closely



bedroom (FP) coast (OT) forest (OT)

highway (OT) industrial (L) inside city (OT)

kitchen (FP) living room (FP) mountain (OT)

office (FP) open country (OT) store (L)

street (OT) suburb (FP) tall building (OT)

Figure 5.5: Example images from the Scene-15 dataset. Each category is labeled with the annotator, where (OT) denotes Oliva and Torralba [94], (FP) is Fei-Fei and Perona [33], and (L) refers to Lazebnik *et al.* [68].

follow Lazebnik *et al.* [68] as this setup has shown excellent performance on these datasets.

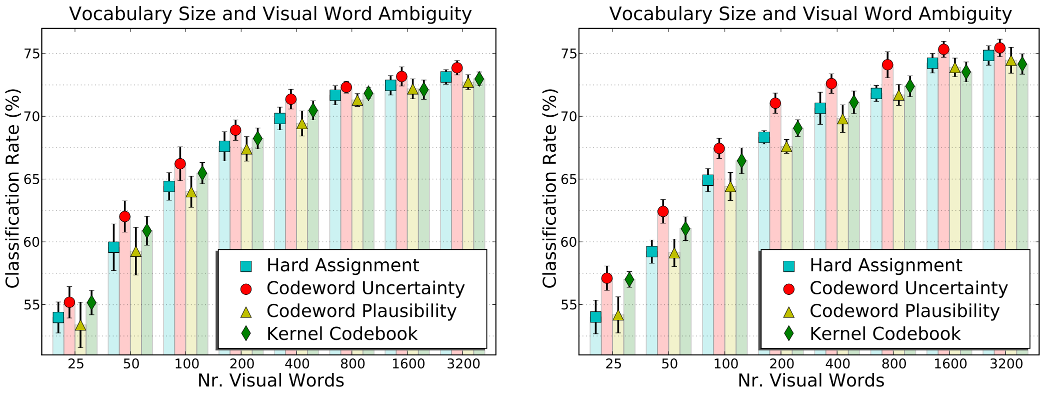
5.3.1 Experimental Setup

To obtain reliable results, we repeat the experimental process 10 times. We select 10 random subsets from the data to create 10 pairs of train and test data. For each of these pairs we create a codeword vocabulary on the train set. The exact same codeword vocabulary is used by both the codebook and the codeword ambiguity approaches to describe the train and the test set. For classification, we use an SVM with a histogram intersection kernel. Specifically, we use libSVM, and use the built in one-versus-one approach for multi-class classification. We use 10-fold cross-validation on the train set to tune parameters of the SVM and the size K_σ of the codebook kernel. The classification rate we report is the average of the per-category recognition rates which in turn are averaged over the 10 random test sets.

For image features we follow Lazebnik *et al.* [68], and use a SIFT descriptor sampled on a regular grid. A grid has been shown to outperform interest point detectors in image classification [33, 62, 93]. We compute all SIFT descriptors on overlapping 16x16 pixel patches, computed over a dense grid sampled every 8 pixels. Due to small implementation differences, our re-implementation of [68] performs slightly under their reported results. However, we use the same re-implementation for all methods of codeword ambiguity. Thus we do not bias any method by a slightly different implementation.

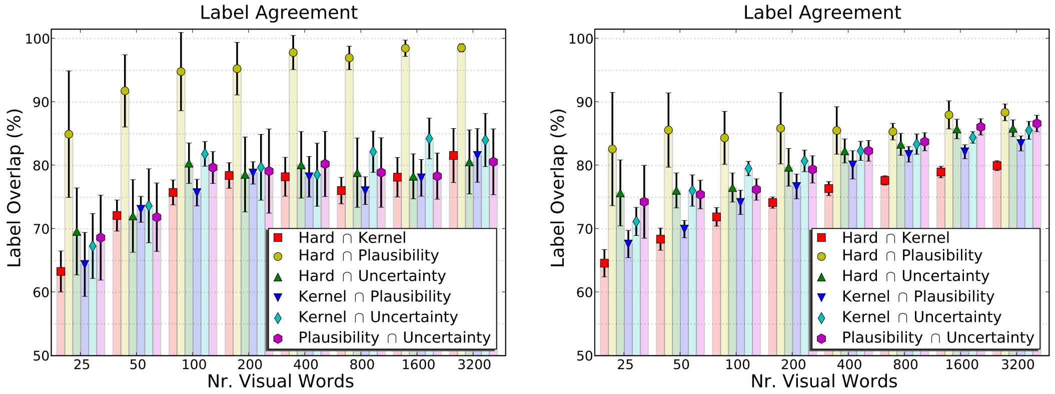
We create a codeword vocabulary by radius-based clustering. Radius-based clustering ensures an even distribution of codewords over the feature space and has been shown to outperform the popular k-means algorithm [62]. Whereas Jurie and Triggs [62] use mean-shift with a Gaussian kernel to find the densest-point, we maximize the number of data samples within its radius r for efficiency reasons.

In figure 5.4 we illustrate for 200 clusters the effect of clustering. We show the similarity distribution from cluster centers to other SIFT descriptors. The similarity distribution adheres to a Weibull shape, as expected [19]. For the Scene-15 dataset, the radius-based clustering algorithm used a radius of $r = 240$ to arrive at 200 clusters. Note, that this radius guarantees that the next cluster is at least a distance of $2r$ away, as can be seen in the bottom row of figure 5.4. Furthermore, note that for each cluster, the distance distribution from the cluster to all points (top row) is fairly similar to the distance distribution from this cluster to the other clusters (bottom row). This similarity suggests that the clusters give a good representation of the complete data.



Dimensionality: 12 Dimensionality: 128

Figure 5.6: Classification performance results of various types of codeword ambiguity for the Scene-15 dataset over various vocabulary sizes and feature dimensions.



Dimensionality: 12 Dimensionality: 128

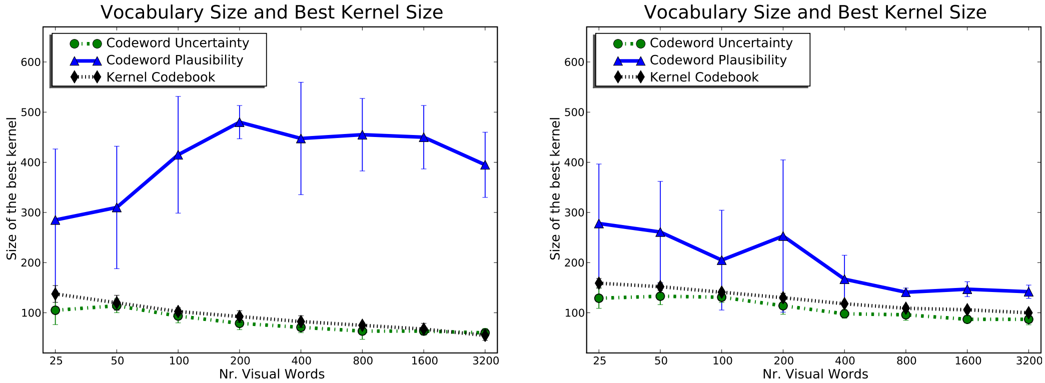
Figure 5.7: Analysis of the class label overlap as predicted by various types of codeword ambiguity for the Scene-15 dataset.

5.3.2 Experiment 1: In-depth Analysis on the Scene-15 Dataset

The first dataset we consider is the Scene-15 dataset, which is compiled by several researchers [33, 68, 94]. The Scene-15 dataset consists of 4,485 images spread over 15 categories. The fifteen scene categories contain 200 to 400 images each and range from natural scenes like mountains and forests to man-made environments like kitchens and offices. In figure 5.5 we show examples of the scene dataset. We use an identical experimental setup as Lazebnik *et al.* [68], and select 100 random images per category as a train set and the remaining images as the test set.

For the Scene-15 dataset, we analyze the types of codeword ambiguity, vocabulary size and feature dimensionality. To evaluate the effect of feature dimensionality on visual word ambiguity we project the 128 length SIFT descriptor to a lower dimensionality. This dimension reduction is achieved with principal component analysis, which reduces dimensionality by projecting the data on a reduced-dimensional basis while retaining the highest variance in the data. We compute a reduced basis on each complete training set, after which we project the train set and corresponding test set on this basis. We reduce the feature length from 128 dimensions to 12 dimensions. A projection to 60 dimensions shows very similar results (data not shown). In evaluating vocabulary size, we tune the radius in the radius-based clustering algorithm to construct eight differently sized vocabularies. The vocabulary sizes we consider are $\{25, 50, 100, 200, 400, 800, 1600, 3200\}$. The results for all types of codeword ambiguity evaluated for various vocabulary sizes and the two feature dimensionalities (12 and 128) are given in figure 5.6.

We start the analysis of the results in figure 5.6 with the various types of codeword ambiguity. The results show that codeword uncertainty consistently outperforms other types of ambiguity for



Dimensionality: 12

Dimensionality: 128

Figure 5.8: Analysis of the best kernel size, found with 10-fold cross-validation, used by various types of codeword ambiguity for the Scene-15 dataset.

all dimensions and all vocabulary sizes. This performance gain is not always significant, however. Nevertheless, for 128 dimensions and a vocabulary size of 200, codeword uncertainty (UNC) outperforms hard assignment with a vocabulary size of 400 and this trend holds for larger vocabulary size pairs: (200-UNC $>$ 400-HARD), (400-UNC $>$ 800-HARD), (800-UNC $\geq$ 1600-UNC) and (1600-UNC $>$ 3200-HARD). On the other end of the performance scale there is codeword plausibility, which always yields the worst results. The third option, a kernel codebook, outperforms hard assignment for smaller vocabulary sizes. For smaller vocabulary sizes the differences between codeword ambiguity types become more pronounced, whereas using a larger vocabulary dampens the differences between ambiguity types. And, as expected, the highest performance gain for codeword ambiguity is in a higher-dimensional feature space. When taking overall performance into account, the results indicate that a higher-dimensional descriptor yields the best results. Moreover, increasing the vocabulary size seems to asymptotically improve performance for all methods. We will investigate larger vocabularies in more detail, later.

To gain insight in the performance variation between the various types of codeword ambiguity we show the overlap percentage between the predicted category labels for all paired method in figures 5.7. The first thing that is striking in figure 5.7, is the high category label overlap between hard assignment and codeword plausibility. This high overlap may be explained by noting that codeword plausibility resembles hard assignment when the kernel size is sufficiently large. Inspecting the kernel sizes as found with cross-validation reveals that the kernel size for codeword plausibility is indeed large. The kernel size for codeword plausibility is typically 200 or larger, whereas the other types of codeword ambiguity range around 100. Furthermore, this label overlap between hard assignment and codeword plausibility is highest with a small number of dimensions. This may be due to the fact that a higher-dimensional space leaves more room for implausible features than a lower dimensional space. The kernel codebook and hard assignment pair have the least number of labels in common. This low label overlap may be expected, since these two types represent the extremes of the types of codeword ambiguity as shown in table 5.1. Further differences of label overlap can be seen between the low- and the high-dimensional feature space. In a high-dimensional feature space there tends to be less correlation between category labels. In a high-dimensional space, the differences between the types of ambiguity become more pronounced, reducing the label overlap. A further trend may be observed in the increased overlap for an increasing vocabulary size. Increasing the vocabulary size yields an increased performance, which requires more labels to be predicted correctly. We attribute the increase in label overlap to those images that are predicted correctly by a larger vocabulary. This link between increased performance and increased category label overlap also explains that the category label overlap is generally high between all types of codeword ambiguity.

To evaluate the influence of the kernel size, we show the kernel size found with 10-fold cross-

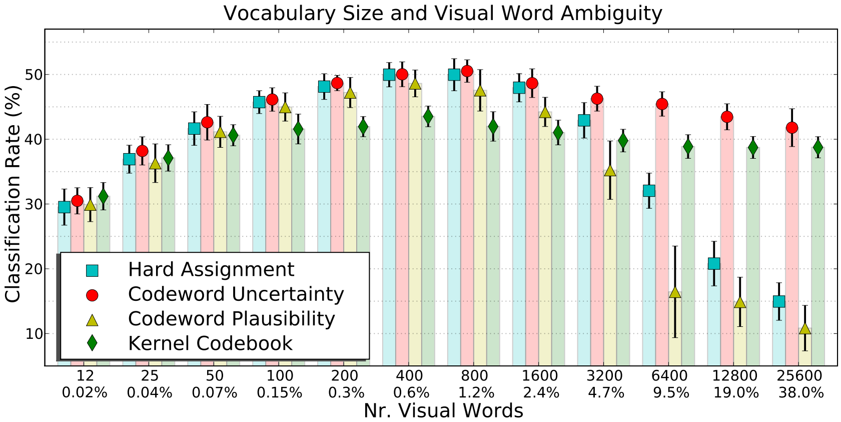


Figure 5.9: Classification performance results of various types of codeword ambiguity for the Scene-15 dataset, trained on 5 images per class. This figures illustrates the effect of relatively large vocabulary sizes compared to the total number of image features.

validation in figure 5.8. The figure shows the optimal kernel size for the various ambiguity types for the two feature dimensions and for an increasing vocabulary size. The kernel size for codeword uncertainty and the kernel codebook show a low variance over the 10 random repetitions. This indicates that these two types of codeword ambiguity have a stable, optimal kernel size. In contrast, the best kernel sizes for codeword plausibility fluctuate heavily over the 10 repetitions. Analyzing the scores, we found that increasing the kernel size of codeword plausibility beyond a sufficiently large value does not change the classification scores much. I.e., for large kernel sizes there are no implausible features left in the finite feature space. Therefore, sufficiently large kernels lead to similar classification performance without a clear optimum, resulting in high kernel size variance for codeword plausibility. In analyzing the kernel size over the number of vocabulary elements shows that a larger vocabulary leads to slightly smaller kernels. This may be expected, since a larger vocabulary is formed by a smaller radius between codewords. When considering the dimensionality of the descriptor, it shows that lower dimensional features use a smaller kernel. This is the case because low-dimensional features typically have a smaller Euclidean distance than high-dimensional features. In summary, the kernel size depends on the type of ambiguity, feature dimensionality and the number of codewords. Therefore, the optimal kernel size cannot be easily inferred from the data and should be found in a discriminative manner, linking it directly to classification performance as achieved with cross-validation.

As illustrated in figure 5.6, increasing the vocabulary size increases the classification performance and the performance of the four ambiguity types seems to converge. In figure 5.6, however, the vocabulary sizes are relatively small. The largest vocabulary in figure 5.6 has 3200 elements and comprises only 0.23% of all features. The behavior of relatively small vocabularies may not be identical to relatively large vocabularies. With vocabulary sizes that are relatively large compared to the total number of training image features, ambiguity type performance may diverge again. To evaluate this, we compared ambiguity type performance on the Scene-15 dataset over relatively large vocabularies.

To make the computation of relatively large vocabularies practically feasible, we reduced the total number of features in the training set. The number of features may be reduced by only extracting features on detected interest points in an image. However, interest point detection would deviate too much from our uniform experimental setup for the Scene-15 dataset. Hence, we keep extracting image features on a regular grid yet constrain the total number of image features by reducing the number of images per class as is also done by [13, 33, 49]. For this experiment, we randomly select 5 images for each of the 15 classes, using the remaining images for the test set. The average number of training feature over the 10 random repetitions amounts to a total of $67,408 \pm 348$ unique SIFT descriptors. Our experiment is not as much concerned with the

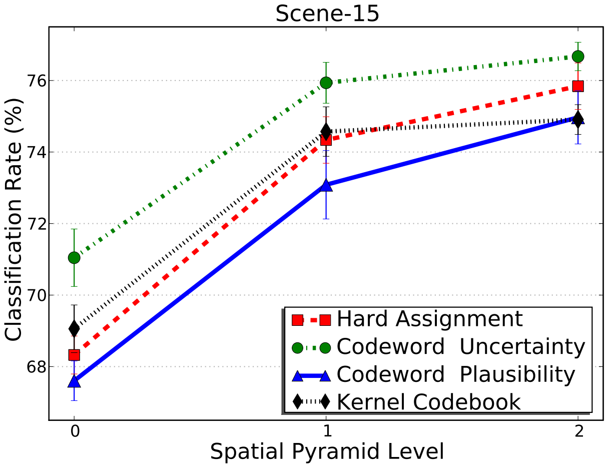


Figure 5.10: Classification performance on the Scene-15 dataset of various types of codeword ambiguity using the spatial pyramid.

total number of features per se, but with the ratio between the number of features and the size of the vocabulary. We want to measure the effect of *relatively* large vocabularies. We evaluated vocabulary sizes ranging from 12 (0.02%) to 25,600 (38%) unique visual words. The underlying assumption is that the results in this experiment trend will hold for various feature and vocabulary sizes, however with similar ratios.

The results for relatively large vocabularies are given in figure 5.9. Note that the performance for relatively small vocabularies show a similar trend as in figure 5.6. Hence, the results in figure 5.6 and figure 5.9 are in agreement. The main difference is the lower performance in figure 5.9 because only 5 images per class are used for training. In figure 5.9 it can be seen that for vocabulary sizes larger than 800 visual words (1.2%), the performance of all methods decreases. We attribute this performance decrease to the curse of dimensionality, albeit that we use a discriminative SVM classifier. In analyzing ambiguity types, it can be seen that for vocabulary sizes of 6,400 and higher, the performance of hard assignment and visual word plausibility severely deteriorates. This may be expected, since both of these ambiguity types can not select multiple suitable visual words. For example, in the extreme case of a vocabulary size equal to the number of image features, codeword plausibility and hard assignment map each training image feature to it's own unique visual word, reverting to exact feature matching. In contrast, the kernel codebook and codeword uncertainty methods both allow selecting multiple relevant visual words. When increasing the vocabulary size, the performance of these two types remains relatively stable, where codeword uncertainty is the better performer. As shown by this experiment, a larger vocabulary does not necessarily yield better results. Actually, a too large vocabulary severely deteriorates performance for codeword plausibility and hard-assignment. A kernel codebook and codeword uncertainty, however, only decrease slightly. Hence, for relatively large vocabularies visual word ambiguity modeling makes a significant difference.

To show the modularity of our approach and improve results we incorporate the spatial pyramid by Lazebnik *et al.* [68]. The spatial pyramid divides an image into a multi-level pyramid of increasingly fine subregions and computes a codebook descriptor for each subregion. The spatial pyramid has been shown to yield excellent performance [12, 68, 70]. We use the 128 dimensional features and a vocabulary of 200 codewords in accordance with Lazebnik *et al.* [68]. The results for the various forms of codeword ambiguity for the first two levels of the spatial pyramid are shown in figure 5.10. Our best result with codeword uncertainty is $76.7 \pm 0.4\%$, whereas hard assignment scores 75.8± 0.6%, both on level 2 of the pyramid. Codeword uncertainty at pyramid level 1 outperforms the traditional codebook at pyramid level 2, effectively saving a complete pyramid level. For the Scene-15 dataset, codeword uncertainty gives the highest improvement at level 0 of the spatial pyramid, which is identical to a codebook model without any spatial structure. Nevertheless,

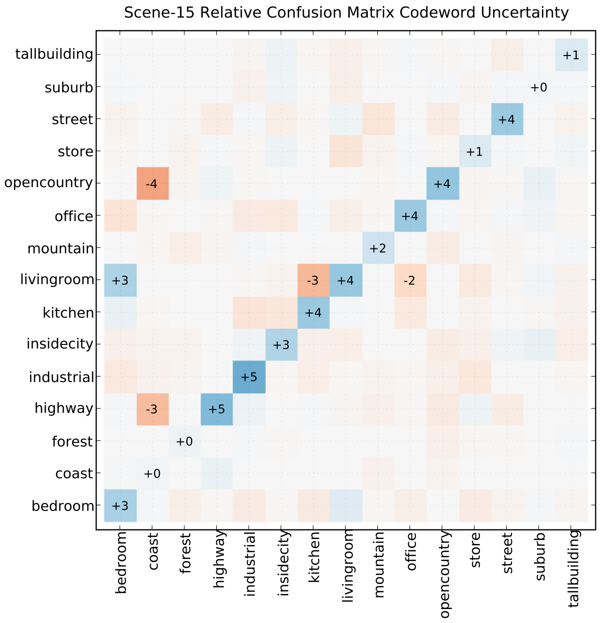


Figure 5.11: Relative confusion matrix of the Scene-15 dataset, for 200 codewords at level 0 of the pyramid, best viewed in color. The relative confusion denotes the increase (blue) or decrease (red) of the absolute classification score of codeword uncertainty compared to hard assignment matrix. We show the average classification percentage per category. The value at column x and row y represents the difference between codeword uncertainty and hard assignment in classifying images of category y as category x.

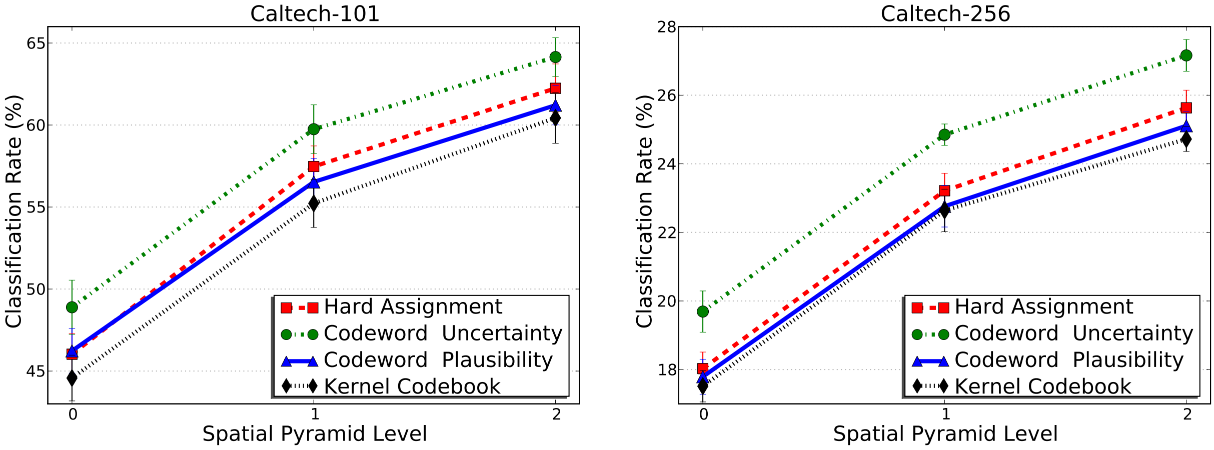
codeword uncertainty outperforms the hard assignment of the traditional codebook for all levels in the pyramid.

The relative confusion matrix of the Scene-15 dataset for 200 codewords at level 0 of the pyramid is shown in figure 5.11. The relative confusion denotes the absolute difference between entries in the confusion matrix of codeword uncertainty relative to the matrix of hard assignment. We focus on hard assignment versus codeword uncertainty, since uncertainty gives the highest improvement of the three types of visual word ambiguity. The non-diagonal entries that represent misclassification rates mostly decrease, or do not change much. The only pair with a higher confusion rate is the confusion between *livingroom* as *bedroom*. Nevertheless, this confusion is compensated by increased discriminative ability between *livingroom* and the categories *kitchen* and *office*. Further considerable confusion reduction is between *open country* as *coast* and *highway* as *coast*. Note that codeword uncertainty improves or matches the correct classification performance for all categories, given by the diagonal.

5.3.3 Experiment 2 and 3: Caltech-101 and Caltech-256

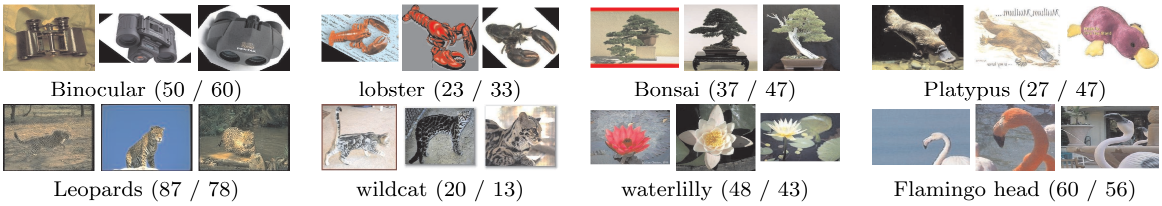
We conduct our second set of experiments on the Caltech-101 [32] and Caltech-256 [49] datasets. The Caltech-101 dataset contains 8,677 images, divided into 101 object categories, where the number of images in each category varies from 31 to 800 images. The Caltech-101 is a diverse dataset, however the obects are all centered, and artificially rotated to a common position. In figure 5.13 we show some example images of the Caltech-101 set. Some of the problems of Caltech-101 are solved by the Caltech-256 dataset. The Caltech-256 dataset holds 29,780 images in 256 categories where each category contains at least 80 images. The Caltech-256 dataset is still focused on single objects. However, in contrast to the Caltech-101 set, each image is not manually rotated to face one direction. In figure 5.14 we show some example images of the Caltech-256 set. We report classification performance on both Caltech sets.

Our experimental results for both the Caltech-101 as Caltech-256 are generated by 30 images



a b

Figure 5.12: Classification performance of various types of codeword ambiguity using the spatial pyramid. (a) Caltech-101 (b) Caltech-256.



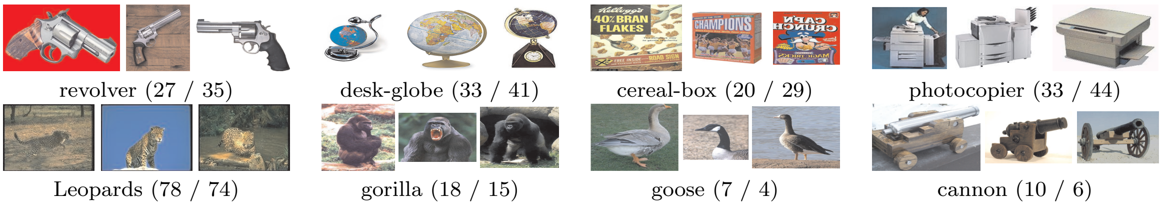
Binocular (50 / 60) lobster (23 / 33) Bonsai (37 / 47) Platypus (27 / 47)

Leopards (87 / 78) wildcat (20 / 13) waterlilly (48 / 43) Flamingo head (60 / 56)

Figure 5.13: Examples of the Caltech-101 set. Top: the top 4 categories where our method improves most, Bottom: the 4 categories where our method decreases performance. The numbers in brackets indicate the classification rate (hard / uncertainty).

per category for training. For testing, we employed 50 images per category for the Caltech 101, and 25 images per category for the Caltech-256. These number of train and test images are typically used for these sets [49, 68]. We use 128 dimensions, and compare the four types of visual word ambiguity. The average classification results per spatial pyramid level for Caltech-101 and Caltech-256 are shown in figure 5.12. These results on Caltech are similar to the results on the Scene-15 dataset. For both sets, the codeword uncertainty method outperforms the traditional codebook considerably in the light of the difficulty of the problem and the simplicity of the improvement. Our best result for Caltech-101 with codeword uncertainty is 64.1 ± 1.5%, whereas hard assignment scores 62.2± 1.2%, both on level 2 of the pyramid. For Caltech-256 our best result is 27.2± 0.4%, whereas hard assignment scores 25.63± 0.5%. The classification performance difference per category between hard assignment and codeword uncertainty are given in figure 5.15. For the Caltech-101 set, there are 86 categories that perform better, or equal with codeword uncertainty. In the case of the Caltech-256 set, there are 199 categories with better or equal performance.

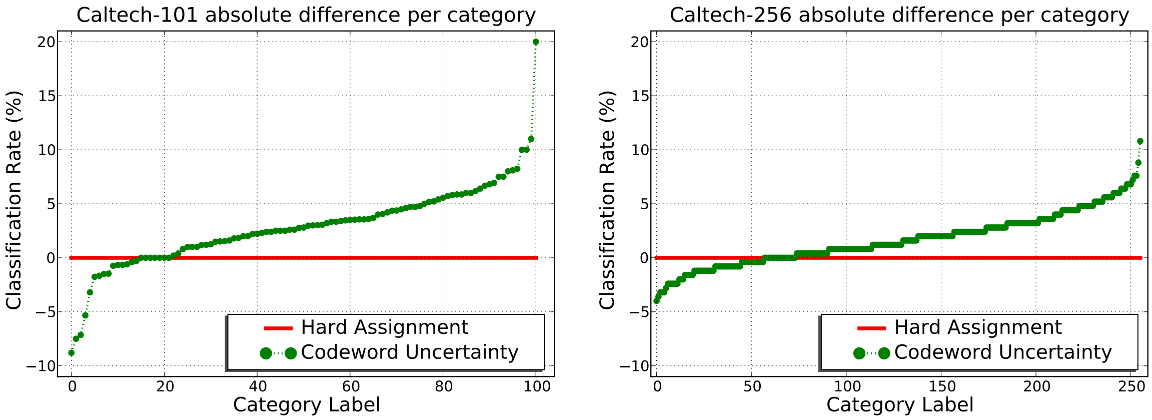
The relative confusion matrices of each Caltech dataset for 200 codewords at level 0 of the



revolver (27 / 35) desk-globe (33 / 41) cereal-box (20 / 29) photocopier (33 / 44)

Leopards (78 / 74) gorilla (18 / 15) goose (7 / 4) cannon (10 / 6)

Figure 5.14: Examples of the Caltech-256 set. Top: the top 4 categories where our method improves most, Bottom: the 4 categories where our method decreases performance most. The numbers in brackets indicate the classification rate (hard / uncertainty).



a

b

Figure 5.15: The classification performance difference per category between hard assignment and codeword uncertainty for (a) Caltech-101 and (b) Caltech-256.

pyramid are given in figure 5.16 and figure 5.17. The relative confusion denotes the difference between entries in the confusion matrix of codeword uncertainty compared to the matrix of hard assignment. Since the size of these datasets prohibits displaying the full confusion matrix, we show the four categories that increase most, and the four categories that decrease most by using codeword uncertainty over hard assignment. Moreover, for each of these categories we show their most confusing, and least confusing category. We focus on the difference between codeword uncertainty and hard assignment, since the former gives the best results and the latter is most commonly used in literature. Some examples of the classes that increase, and decrease most are given in figures 5.13 and 5.14.

We start the analyses of the relative confusing matrices of the Caltech datasets with the categories where performance decreases most. These object categories consist mostly of natural images that are captured including their contextual background. We deem this background as the reason for a decreased performance by ambiguity modeling. The background is very similar for several natural images. By incorporating ambiguity modeling this similarity is enhanced, leading to more confusion. In analyzing the categories that improve most, we observe that these categories mainly consist of man-made objects, and objects that are photographed without context. We conjecture that the reason why these classes benefit most from codeword ambiguity is that these object classes have little intra-class variation. Small variations may lead to completely different codewords when using the hard assignment as in the traditional codebook model. In contrast, our approach of ambiguity modeling will reserve weight for multiple, suitable codewords, leading to classification improvements.

5.3.4 Experiment 4: PASCAL VOC07-20 and VOC08-20 Datasets

As a final experiment, we consider the Pascal VOC 2007 [28] and 2008 challenge [29]. The VOC challenges consist of twenty object classes with 9,963 images in 2007 and in 10,057 images in 2008. These image sets are each split in half to a given train and test set. The Pascal VOC Challenge provides a yearly benchmark of object recognition algorithms. We follow the successful approach by Marszałek *et al.* [74], which was extended by Tahir and Van de Sande *et al.* [126]. Specifically, for each image we combine Harris-Laplace point sampling with densely sampling every 6 pixels. These points are subsequently represented by SIFT, and various color-SIFT descriptors [132]. The descriptors of the train set with around 5000 features per image are subsequently clustered by k-means to create a vocabulary of 4,000 codewords. This vocabulary is used in the codebook model at level 1 of Lazebniks spatial pyramid where we use a support vector machine with a χ^2 kernel for image classification. We fuse the classification scores for the various SIFT descriptors with a simple geometric mean. The final classification performance is measured in average precision,

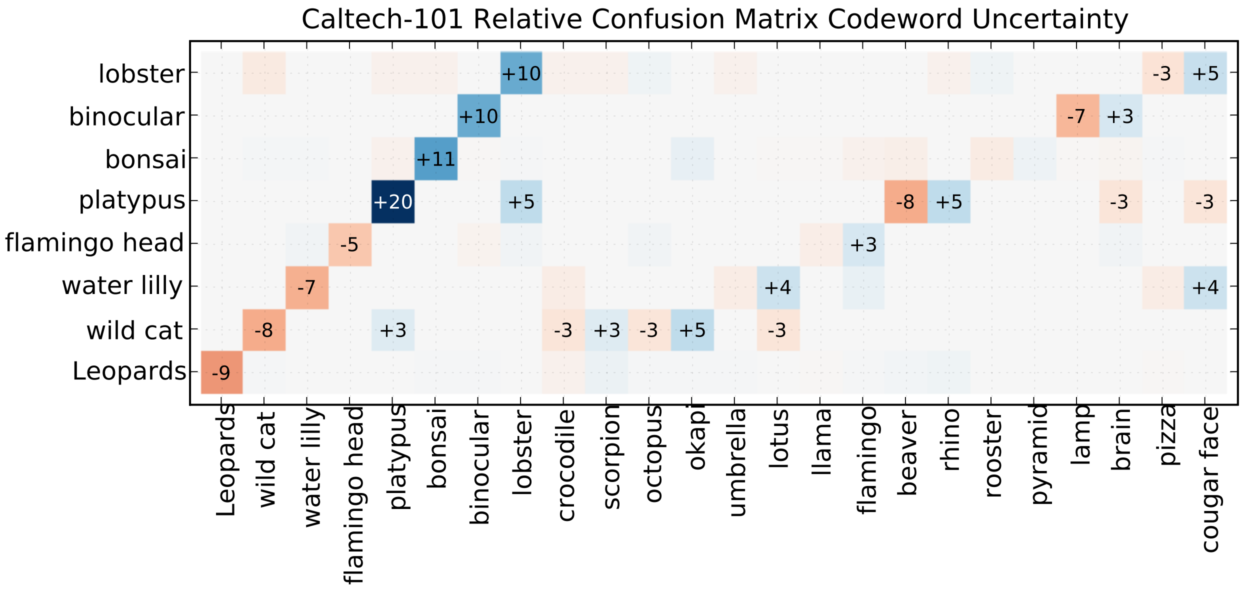


Figure 5.16: Relative confusion matrix for the Caltech-101 dataset, best viewed in color. We show the 4 categories that increase most, and the 4 categories that decrease performance most. Each of these 8 categories is paired with its most confusing and least confusing category. The value at column x and row y represents the difference between codeword uncertainty and hard assignment in classifying images of category y as category x.

which represents the area under the precision-recall graph.

We experimentally compared the traditional codebook model with codeword uncertainty and with the best two participating systems on the respective Pascal challenge. In figure 5.18 we show the results for both Pascal VOC 2007 and 2008. For Pascal VOC 2007 (VOC07-20) our implementation with codeword uncertainty performs best for 15 of the 20 object classes. The best method for the 5 other object classes is INRIA_Genetic. In terms of mean average precision over all object classes, our implementation with codeword uncertainty scores best with 0.605, followed by INRIA_Genetic with 0.594, hard assignment with 0.580 and XRCE with 0.556. Note that the traditional codebook model occupies the third place, whereas replacing hard assignment with codeword uncertainty yields the best result. Moreover, codeword uncertainty outperforms hard assignment for all 20 categories of VOC07-20. In the case of Pascal VOC 2008 (VOC08-20), SurreyUvA_SRKDA claims 9 categories, LEAR_shotgun wins 9, and codeword uncertainty is the best for 4 categories[2]. The best system in mean average precision is SurreyUvA_SRKDA with 0.549, followed by LEAR_shotgun with 0.545, codeword uncertainty with 0.541 and 0.521 for the traditional codebook. The SurreyUvA_SRKDA system with the best mean average precision already uses codeword uncertainty as a part of their method [126]. The main difference between SurreyUvA_SRKDA and our results presented here, is the use of a classifier with multiple kernel learning which is out of scope for this article. In comparing hard assignment with codeword uncertainty, the latter slightly decreases the performance for the category *cow*. For the other 19 categories of VOC08-20 the performance of codeword uncertainty is equal or better than hard assignment.

5.4 Discussion

This Chapter presented a principal improvement on the popular codebook model for scene classification. The traditional codebook model uses hard assignment to represent image features with codewords. We replaced this basic property of the codebook approach by introducing uncertainty

[2] This totals to 22 because 2 systems share the best score for the categories *person* and *bus*.

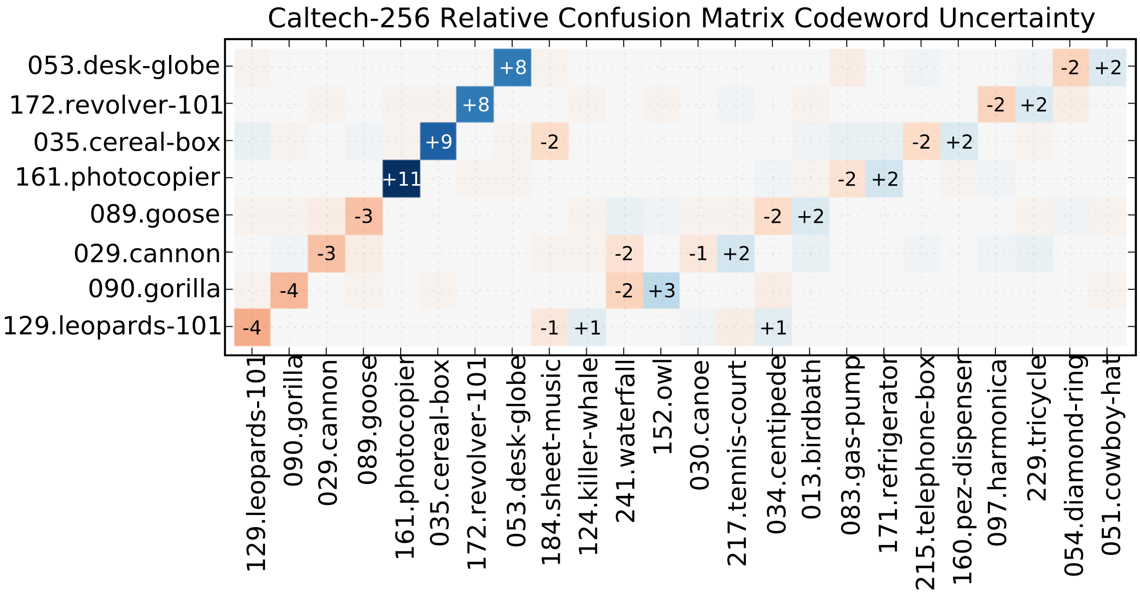


Figure 5.17: Relative confusion matrix for the Caltech-256 dataset, for the top 4 and bottom 4 categories as in figure 5.16. The value at column x and row y represents the difference between codeword uncertainty and hard assignment in classifying images of category y as category x.

modeling, which is appropriate as discrete feature vectors are only capable of capturing part of the intrinsic variation in visual appearance. This uncertainty is achieved with techniques based on kernel density estimation.

The experiments on the Scene-15 dataset in figures 5.6 and 5.10 show that of the four considered ambiguity types, codeword plausibility hurts performance. Codeword plausibility (PLA), and the unnormalized kernel-codebook (KCB), are dominated by those few representative image features that are significantly close to a codeword. In essence, PLA and to a lesser extent KCB, ignore the majority of the features, and leads us to conclude that it is better to have an implausible codeword representing an image feature then no codeword at all. When no codeword is selected, all statistical classification techniques developed to deal with noisy data are not used to their full potential. Therefore, codeword uncertainty yields the best results, since it models ambiguity between codewords, without taking codeword plausibility into account.

The results in figure 5.6 indicate that codeword ambiguity is more effective for higher-dimensional features than for lower dimensions. The curse of dimensionality prophesizes that increasing the dimensionality increases the fraction of feature vectors on or near the boundary of codewords. Hence, increasing the dimensionality will increase codeword uncertainty, leading to better results for ambiguity modelling with higher-dimensional features.

Figure 5.6 seems to suggest that a larger vocabulary is always better. Furthermore, the figure suggests that for larger vocabularies the performance of hard assignment and soft-assignment converges. Figure 5.9 illustrates that both these suggestions are not the case. Figure 5.9 shows that a too large vocabulary severely deteriorates the performance of hard assignment, whereas codeword ambiguity degrades only slightly. In the case of the VOC2007/2008 with around 5000 images in the training set with close to 5000 features per image, a vocabulary of 4000 words is rather small. Because of this relatively small vocabulary there is a significant improvement of soft-assignment over hard assignment. Even more performance improvement can be expected by choosing a much larger vocabulary. However, as shown in figures 5.6 and 5.9, the positive effect of a larger vocabulary size on the performance decreases logarithmically. Hence, it takes a vocabulary size of several orders of magnitude higher to obtain a significant improvement. Such larger vocabularies makes it practically infeasible to compute all (color) descriptors, spatial pyramid levels, and machine learning techniques. In contrast, ambiguity modeling provides increased performance at much lower

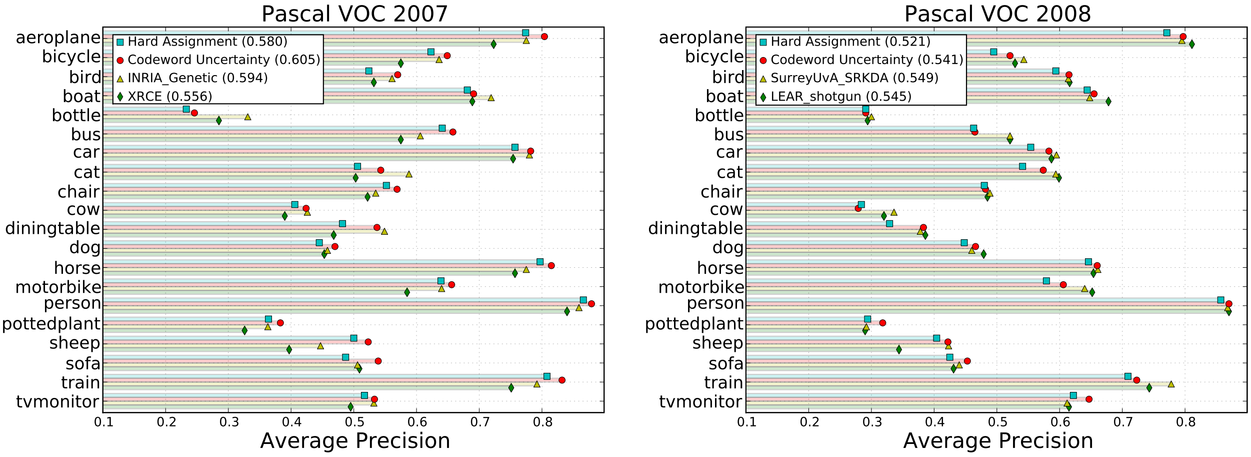


Figure 5.18: Average Precision of the traditional codebook model and codeword uncertainty per category, compared against the best two participants for Pascal VOC 2007 (left) and Pascal VOC 2008 (right). The mean average precision for each method is shown in the legend. Note that codeword uncertainty is used by the SurreyUvA_SRKDA method that participated in Pascal VOC 2008.

Data set	**Train set size**	**Test set size**	**Performance Increase**
Scene-15	1,500	2,985	4.0 ± 1.7 %
Caltech-101	3,030	5,050	6.3 ± 1.9 %
Caltech-256	7,680	6,400	9.3 ± 3.0 %
VOC07-20	5,011	4,952	4.3 %
VOC08-20	4,340	5717	3.8 %

Table 5.2: The relationship between the data set size and the relative performance of codeword uncertainty over hard assignment for 200 codewords in the Scene-15 and Caltech datasets and 4,000 codewords for the VOC07-20 and VOC08-20 sets.

computational costs.

The results over the Scene-15, Caltech-101, Caltech-256, and Pascal datasets are summarized in table 5.2. This table shows the relative improvement of codeword uncertainty over hard assignment. Note that the result for the Pascal datasets is set apart, since it adheres to a different experimental setup. As can be seen in this table, the relative performance gain of ambiguity modeling increases as the number of scene categories grows. A growing number of scene categories requires a higher expressive power of the codebook model. Since the effects of ambiguity modeling increase with a growing number of categories, we conclude that ambiguity modeling is more expressive then the traditional codebook model. The results of all experiments show that codeword uncertainty outperforms the traditional hard assignment over all dimensions, all vocabulary sizes, and over all datasets.

We have demonstrated the viability of our approach by improving results on recent codebook methods. These results are shown on five well-known datasets, where our method consistently outperforms the traditional codebook model. We have shown that ambiguity modeling can obtain the same performance as hard assignment with a considerable smaller vocabulary. What is more, we found that hard assignment suffers more from the curse of dimensionality, whereas our ambiguity modeling approach reaps higher benefits in a high-dimensional feature space. Furthermore, the performance of hard assignment completely deteriorates when using relatively large vocabularies, while the proposed model performs consistently. Similarly, an increasing number of scene categories increases the effectiveness of our method. As future image features and datasets are expected to increase in size, our ambiguity modeling method is unambiguously likely to have more impact.

Chapter 6

Color Invariant Object Recognition using Entropic Graphs[1]

6.1 Introduction

Humans are capable of distinguishing the same object from millions of different images. Machines on the other hand have significant difficulty with this seemingly trivial task. One of the reasons that computational object recognition is such a hard problem is that machines take sensory information very literally, making object recognition vulnerable to accidental scene information. Such accidental variations include scale, illumination color, viewing angle, background, occlusion, shadows, shading, light intensity, highlights, and many more [117].

One approach to dealing with such photometric variations is found in the use of invariant features. Invariant features remain unchanged under certain operations or transformations and are used for various object recognition approaches. For example, the physical laws of image formation can be used to factor out accidental scene effects. The dichromatic reflection model by [111] integrates body and surface reflection properties. This model may be extended upon, to obtain color invariant measurements [36, 37, 44, 47].

In order to compare object images, a similarity measure between image features is required. Often, similarity measures are used that require some parameter tuning in order to be applicable to other datasets or features. An example of such a parameter is the bin-size for histogram matching. A generic alternative is found in the use of unparametric similarity measures. We use entropic graphs [54] to compute an unparametric similarity between image features.

This Chapter utilizes color invariant features for object recognition. We employ an unparametric entropic similarity measure to match object images. Furthermore, the object recognition scheme is evaluated on a large dataset with real-world imaging conditions.

6.1.1 Related Work

A popular method for object recognition is to apply salient point detectors. This method deals with problems, such as partial matching and occluded images. Specifically, [107] use salient point detection for indexing gray images. The detected points are subsequently made robust for scale changes and transformed to be rotationally invariant. In a similar approach, rotational and scale invariant keypoints allows for robust object detection [71]. Scale Invariant Feature Transform (SIFT) features are extracted and matched against a database. A Hough transform gives high probability to multiple features matched in one image. One problem with the interest point approach is the repeatability of the salient point detection. For example, detection may vary depending on pose, illumination, and background changes. Thus, salient points are not guaranteed to be the same over various imaging conditions. Moreover, for images without high curvature the method might not detect any salient points at all.

[1] Published in the *International Journal of Imaging Systems and Technology* [134].

An alternative approach is given by [106] who take multiscale histograms of local gray value structure in an image. Translation invariance is given by the use of histograms. Rotational invariance is achieved by using several rotated versions of a steerable filter in steps of 20°. This technique proves robust for rotated, occluded, and cluttered scenes. Grayscale images, however, lack a significant amount of information compared to color images. In our opinion, using color features in an object recognition approach is favorable, as color is a highly discriminative property of objects.

A biologically inspired object recognition method is presented with SEEMORE [76]. Object matching is achieved with histograms of 102 different filters. Each filter responds to different image features like contour, texture, and color. Experiments are performed over a collection of 100 images. The highest experimental recognition rate of 97% is achieved with color and shape features. By using only color features, 87% recognition is achieved, as opposed to 79% without color. Thus color information may significantly improve object recognition.

[37] propose color invariant histograms for illumination-independent object recognition. Under the assumption of a slowly varying illumination, computed color ratios of neighboring pixels are color invariant. The color ratio is computed by taking derivatives of the logarithm of the color channels. Object recognition experiments were conducted for differing illuminations. Results show that histograms of color ratios outperform color histograms. Histogram bin-size is usually set in an ad-hoc manner, where the best bin size for a specific application is experimentally determined. Kernel density estimation tries to overcome the problem of selecting a suitable bin size for a histogram.

Color invariant histograms may be improved upon by using variable kernel density estimation [47]. Here, an error propagation method is introduced to estimate the uncertainty of a color invariant channel. This associated uncertainty is used to derive the optimal parameterization of the variable kernel used during histogram construction. In this way, a robust estimator of invariant density is constructed. However, noise characteristics of the camera system are often not available.

A solution to image matching without the use of histograms is found in assuming prior knowledge about the probability distributions. A popular approach is mixture of Gaussian estimation [149]. However, not all processes can be described with a fixed parameterized model. Furthermore, assuming one distribution might severely over-simplify the complexity of the data.

Entropic graphs [54] offer an unparameterized alternative to histograms, circumventing choosing and fine tuning parameters such as histogram bin size or density kernel width. Alternatively, classifiers such as support vector machines may be employed for object recognition [98]. A support vector machine [141] finds the best separating hyper plane between two classes. In contrast to support vector machines, entropic graphs allow to estimate information theoretic measures, like entropy, divergence, mutual information and affinities.

In our approach, we extend the work of [106, 37, 47, 54] combining higher order color invariant features with an entropy graph based similarity measure. We extract color invariant features from object images, invariant to viewpoint, shadow and shading. As opposed to using a histograms or kernel density estimations, we employ entropic graphs. The Henze-Penrose similarity measure is then used to compute the similarity of two images. Finally, we evaluate our method on a large collection of object images. The object image collection consists of 1,000 objects recorded under various imaging circumstances.

The Chapter is organized as follows. The next section discusses the color invariant model, section 6.3 introduces entropic graphs and the Henze-Penrose similarity measure. Section 6.4 presents experimental results, after which section 6.5 concludes the Chapter.

6.2 Color Invariant Features

Color is defined in terms of human observation. There is no one-to-one mapping of the spectrum of a light source to the perceived color. The Gaussian color model described in [43] approximates the spectrum with a smoothed Taylor series. In accordance with the human visual system, the Gaussian color model uses second order spectral information. The zeroth order derivative measures the luminance, the first order derivative the 'blue-yellowness', and the second order the 'red-greenness' of a spectrum.

A RGB image is measured in the Red Green and Blue sensitivity components of the light. The RGB sensitivities have to be transformed to the Gaussian spectral derivatives. In [43] an optimal transformation matrix with the Taylor expansion in the point $\lambda_0 = 520$nm and with a Gaussian spectral scale of $\sigma_\lambda = 55$nm is derived under the assumption of standard REC 709 CIE RGB sensitivities:

$$\begin{bmatrix} E \\ \partial_\lambda E \\ \partial_{\lambda\lambda} E \end{bmatrix} = \begin{pmatrix} 0.06 & 0.63 & 0.27 \\ 0.3 & 0.04 & -0.35 \\ 0.34 & -0.6 & 0.17 \end{pmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}. \tag{6.1}$$

When comparing images of the same object, differences in measurement due to the scene environment pose a problem. Taking two pictures of an object yields two different representations of the same scene. Differences in lighting conditions and in camera rotation change the recorded measurements of the scene. Image invariants deal with the problem to measure the information in a scene, independent of properties not inherent to the recorded object. Color invariance aims at keeping the measurements constant under varying intensity, viewpoint and shading. In [44] several of these color invariants for the Gaussian color model are described. A property $\mathcal{C}$ invariant for viewpoint, shadow and shading invariance, is given by

$$\mathcal{C}_{\lambda^m x^n} = \frac{\partial^n}{\partial_{x^n}} \frac{1}{E(\lambda, x)} \frac{\partial^m}{\partial_{\lambda^m}} E(\lambda, x) \qquad m \geq 1, n \geq 0, \tag{6.2}$$

where E is the energy. The $\mathcal{C}$ invariant normalizes the spectral information with the energy E and computes the spatial derivatives independent of the spectral energy. Note that the derivatives on the right-hand side of the equation represent measurements in the Gaussian color model. This makes the local spatial neighborhood invariant for intensity changes like shadow and shading.

Each pixel can be described with a color invariant feature vector. For example a second order spatial representation of a pixel E yields the invariant counterparts of

$$\begin{aligned} &\{C_\lambda, C_{\lambda x}, C_{\lambda y}, C_{\lambda xx}, C_{\lambda xy}, C_{\lambda yy}, \\ &C_{\lambda\lambda}, C_{\lambda\lambda x}, C_{\lambda\lambda y}, C_{\lambda\lambda xx}, C_{\lambda\lambda xy}, C_{\lambda\lambda yy}\}. \end{aligned} \tag{6.3}$$

Note that only color information is used as all luminance information is discarded.

The invariant expressions up to second order are given by,

$$\begin{aligned}
&C_\lambda = \tfrac{E_\lambda}{E}, \quad C_{\lambda x} = \tfrac{E_{\lambda x}E - E_\lambda E_x}{E^2}, \quad C_{\lambda y} = \tfrac{E_{\lambda y}E - E_\lambda E_y}{E^2}, \\
&C_{\lambda\lambda} = \tfrac{E_{\lambda xx}}{E}, \quad C_{\lambda\lambda x} = \tfrac{E_{\lambda\lambda x}E - E_{\lambda\lambda}E_x}{E^2}, \quad C_{\lambda\lambda y} = \tfrac{E_{\lambda\lambda y}E - E_{\lambda\lambda}E_y}{E^2}, \\
&C_{\lambda xx} = \tfrac{E_{\lambda xx}E^2 - E_\lambda E_{xx}E - 2E_{\lambda x}E_x E + 2E_\lambda E_x^2}{E^3}, \\
&C_{\lambda yy} = \tfrac{E_{\lambda yy}E^2 - E_\lambda E_{yy}E - 2E_{\lambda y}E_x E + 2E_\lambda E_y^2}{E^3}, \\
&C_{\lambda xy} = \tfrac{E_{\lambda xy}E^2 + E_{\lambda x}E_y E - E_{\lambda y}E_x E - E_\lambda E_{xy}E - 2E_{\lambda x}E_y E + 2E_\lambda E_x E_y}{E^3}, \\
&C_{\lambda\lambda xx} = \tfrac{E_{\lambda\lambda xx}E^2 - E_{\lambda\lambda}E_{xx}E - 2E_{\lambda\lambda x}E_x E + 2E_{\lambda\lambda}E_x^2}{E^3}, \\
&C_{\lambda\lambda yy} = \tfrac{E_{\lambda\lambda yy}E^2 - E_{\lambda\lambda}E_{yy}E - 2E_{\lambda\lambda y}E_y E + 2E_{\lambda\lambda}E_y^2}{E^3}, \\
&C_{\lambda\lambda xy} = \tfrac{E_{\lambda\lambda xy}E^2 + E_{\lambda\lambda x}E_y E - E_{\lambda\lambda y}E_x E}{E^3} - \tfrac{E_{\lambda\lambda}E_{xy}E - 2E_{\lambda\lambda x}E_y E + 2E_{\lambda\lambda}E_x E_y}{E^3}.
\end{aligned}$$

Indices denote differentiation by Gaussian convolution.

6.3 Entropic Graphs

This section advocates entropic difference measures as an alternative to commonly used difference measures. The entropy measures the information content of a random variable. The information

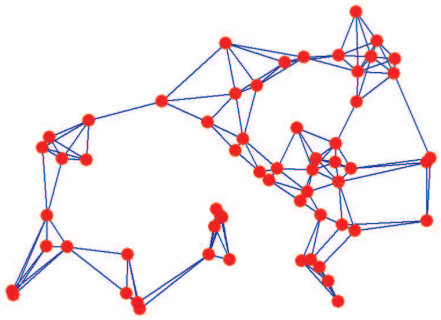


Figure 6.1: An example of a 4-nearest neighbors graph.

in one variable may be used to describe another, by utilizing the mutual information between the two variables. High mutual information implies a high similarity between the two random processes. The difference between two probability distributions is given by the Kullback-Leibler (KL) divergence. The KL divergence between p(x) and q(x) may be seen as the average error by describing distribution p(x) with a distribution q(x). Entropic distance measures are theoretically sound and can capture non-linear relations between probability distributions. Several applications of entropy can be found, for example, in image registration [123], image retrieval [142], video modeling [16], and saliency detection [63].

The entropy of high dimensional features is hard to estimate. Two methods to compare images are: 1) histogram matching and 2) assuming a fixed probability density function. Entropy may be estimated from a histogram. A histogram is a fast and easy to compute method, making no assumptions on the underlying probability distribution. However, the problem of selecting a suitable histogram bin size is more of an art than science. Moreover, for a fixed resolution per dimension, the number of bins increases exponentially in the number of dimensions. Kernel density estimators are a general case of histogram methods [148]. Nevertheless, the problems of selecting the size of the kernel and the curse of dimensionality also apply to kernel density estimation. Another solution to estimating entropy is by assuming prior knowledge about the probability distributions. When the probability distributions can be described with a parameterized model the computation of the entropy becomes feasible. However, not all processes can be described with a fixed parameterized model. Furthermore, assuming one distribution might severely over-simplify the complexity of the data.

Entropic graphs provide an unparameterized, efficient way to estimate the entropy of high dimensional data [54]. An entropic graph is any graph whose normalized total weight (sum of the edge lengths) is a consistent estimator of Rényi's α-entropy. Examples of entropic graphs are the Minimum Spanning Tree and the k-nearest neighbor graph. One advantage of combinatorial methods is that the computation and storage complexity increase linearly in feature dimension. Additionally, graph based estimators have fast asymptotic convergence rates and bypass the complication of choosing and fine tuning parameters such as histogram bin size or density kernel width.

Rényi's α-entropy [104] is a generalization of the Shannon entropy and is defined by

$$H_\alpha(f) = \frac{1}{1-\alpha} \log \int_X f^\alpha(x)dx \quad . \tag{6.4}$$

The α-entropy converges to the Shannon entropy $H(f) = -\int f(x) \log f(x)dx$, as $\alpha \to 1$. For α smaller than 1, the tails in the distribution are heavily weighted in the entropy.

The α-entropy can be estimated by the length of a minimal graph through sample points. Given a set $X_n = \{x_1, x_2, ..., x_n\}$ of n i.i.d vectors in a d-dimensional feature space, the length of a graph is given by

$$L_\gamma(X_n) = \sum_{e \in G(X_n)} |e|^\gamma \quad . \tag{6.5}$$

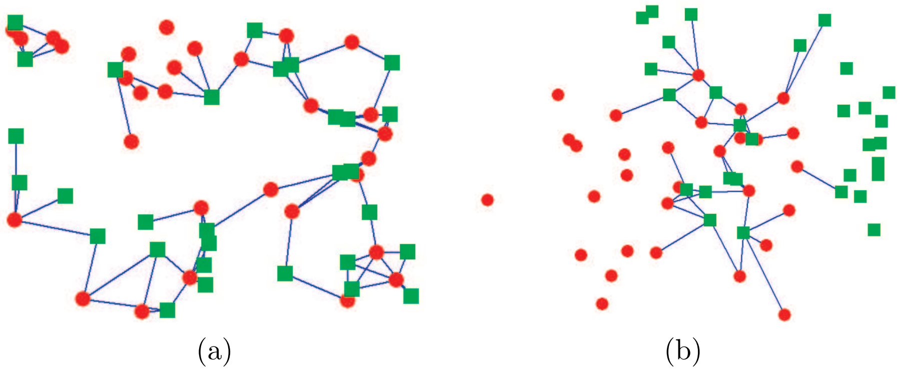


Figure 6.2: Example of the Henze-Penrose affinity in 2 dimensions. (a) The sample points $\{X\}$ (circle) and $\{Y\}$ (square) are drawn from the same uniform distribution. The calculated affinity is 0.85. (b) The sample points $\{X\}$ (circle) and $\{Y\}$ (square) are drawn from slightly different uniform distributions. The calculated affinity is 0.41.

The graph G is over a suitable substructure, e.g. k-nearest neighbor graphs (see figure 6.1). Furthermore, e are edges in a graph connecting pairs of X_i's and $|e|$ denotes the Euclidean distance. The weighting $\gamma \in (0, d)$ relates to the value of α in the α-entropy as $\alpha = (d - \gamma)/d$, where d is the dimensionality of the feature space.

The entropic graph estimator

$$\hat{H}_\alpha(X_n) = \frac{1}{1-\alpha} \log L(X_n)/n^\alpha - \log c \quad , \tag{6.6}$$

is an asymptotically unbiased and consistent estimator of the α-entropy, where c is a constant independent of the data.

Entropic graphs can be used to estimate several similarity measures. These similarity measures include: the α-mutual information, α-Jensen difference divergence, the Henze-Penrose affinity, and the α-geometric-arithmetic mean divergence. For $\alpha \rightarrow 1$, the α-divergence reduces to the Kullback-Leibler divergence, and the α-mutual information to the Shannon mutual information. When α approaches 1, central differences between the two densities become highly pronounced. When α approaches 0, tail differences between two densities f and g become most influential. Therefore, if the feature densities differ in regions where there is a lot of mass one should choose α close to 1 to ensure locally optimum discrimination. Alternatively, if the tails or extreme values of the distribution describe the important events, α should be chosen close to 0.

One measure of similarity between probability distributions f and g is the Henze-Penrose (HP) [53] affinity,

$$D_{\mathcal{HP}} = 2pq \int \frac{f(x)g(x)}{pf(x) + qg(x)} dx \quad , \tag{6.7}$$

where $p \in [0, 1]$ and $q = (1 - p)$.

In [89] an entropic graph algorithm for the Henze-Penrose affinity is introduced for given sample points $\{X_i\}_{i=1}^m$ of f, and $\{Y_i\}_{i=1}^n$ of g. For given samples, the value for p in equation 6.7 is directly related to the number of samples: $p = \frac{m}{m+n}$. The entropic graph algorithm to estimate the Henze-Penrose affinity is given by:
1. Construct the k-nearest neighbor graph on the sample points $\{X\} \cup \{Y\}$;
2. Keep only the edges that connect an X-labeled point to an Y-labeled point ;
3. The HP-test affinity is given by the number of edges retained, divided by $(m + n)k$ for normalization.

This algorithm constructs an entropic graph on the edges that connect classes $\{X\}$ and $\{Y\}$. Counting the connecting edges implies a power weighting with 0. Therefore, the value for α in the estimated α-entropy is 1, emphasizing central differences between the two classes.

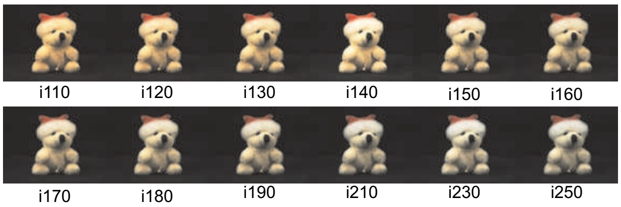


Figure 6.3: Example object from the ALOI collection, viewed under 12 different illumination color temperatures.

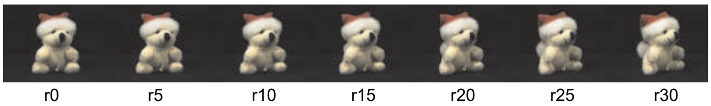


Figure 6.4: Example object from the ALOI collection, viewed from different viewing directions.

In figure 6.2 we show two-dimensional examples of the Henze-Penrose affinity. The examples show sample points $\{X\}$ and $\{Y\}$ drawn from the same uniform distribution, and from a slightly different distribution, respectively. The affinity between the points drawn from the same distribution is significantly higher.

6.4 Experiments

Performance is evaluated with an object recognition task on the ALOI dataset [41]. The ALOI collection consists of 1,000 objects recorded under various imaging circumstances. For each object the viewing angle, illumination angle, and illumination color are varied. See figures 6.3, 6.4, 6.5 and 6.9, for examples of the collection.

The combination of a large image dataset with a considerable variety of appearance offers a formidable challenge for object recognition. Object recognition is the problem of matching one appearance of an object against a standardized version. One object may give rise to millions of different images, as camera conditions may be varied endlessly. In our recognition experiment, one prototypical version of each object in the ALOI dataset is indexed and the diversity of recorded object variations in the collection are used for querying. An object is perfectly recognized when for all different variations the correct indexed object is returned. In this case, one may assume that the object can be recognized under a wide variety of real-life imaging circumstances.

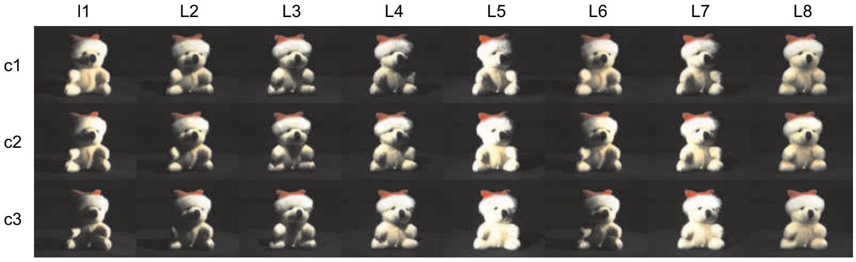


Figure 6.5: Example object from the ALOI collection, viewed under 24 different illumination directions. Each row shows the recorded view by one of the three cameras. The columns represent the different lighting conditions used to illuminate the object

6.4.1 Implementation

Entropic graphs are constructed with k-nearest neighbor search. The nearest neighbor search is implemented using the approximation algorithm by [90]. The nearest neighbor search is simple to implement and efficient in high dimensions. The algorithm proposed in [90] constrains possible nearest neighbors of a point p inside a high-dimensional hypercube around p. For each dimension i, the points outside the limits $i - \epsilon$ and $i + \epsilon$ are discarded where the value of ϵ is typically small. For given distributions, ϵ can be set to an optimal value. For unknown data, however, ϵ may be empirically estimated. An offline sorted data structure makes discarding the points outside the hypercube efficient. In the case of entropic graph construction, this data structure needs to be computed for each query.

We extend the nearest neighbor algorithm specifically for entropic graph construction. Particularly, the approximate nearest neighbor algorithm is transformed to an optimal, exact algorithm. An entropic graph computes the k-nearest neighbors for each query image Q with every database image D. For each point p in the image D, the Euclidean distance to the k-th nearest neighbor, which is furthest away, is stored. These distances are subsequently used as the ϵ values in computing the neighbors to p in Q. Because this ϵ value is the point furthest away in D, all points discarded can never be a k-nearest neighbor of $Q \cup D$. Hence, yielding an optimal value for ϵ, thus an exact, and more efficient entropic graph algorithm.

Before constructing the entropic graphs we pre-process the images to extract features. The values of the color invariant N-jet are sub-sampled, thresholded and whitened. We compute the second order color invariant N-jet by convolution with a Gaussian of $\sigma = 2$. Due to Gaussian smoothing there is a high correlation between neighboring pixel values. Therefore, we keep only 1 pixel in a block of 4 pixels. Sub-sampling will significantly increase the speed of the entropic graph construction. Color invariance is achieved by dividing by the intensity. Hence, the invariants are unstable when the intensity approaches zero. All pixels with intensity lower than 15 gray values are discarded. As the nearest neighbor search uses a hypercube, whitened (or sphered) data is required. Whitening is achieved by dividing all data by a pre-computed standard deviation for each invariant feature. The 1,000 reference images are used for the calculation of the standard deviation. The extracted features are input for the entropic graph matching.

A single match on a standard PC takes 600 milliseconds. Given the size of the dataset, all computations have been performed on the Distributed ASCI Supercomputer 2 (DAS-2), a wide-area distributed computer located at five different universities in The Netherlands [3]. DAS-2 consists of five Beowulf-type clusters, one of which contains 72 nodes, and four of which have 32 nodes (200 nodes in total). All nodes consist of two 1.0 GHz Pentium III CPUs, at least 1.0 GByte of RAM, and are connected by a Myrinet-2000 network.

We used the parallel Horus framework introduced in [109]. The Parallel-Horus framework is a software architecture that allows non-expert parallel programmers to develop fully sequential multimedia applications for efficient execution on homogeneous Beowulf-type commodity clusters. The core of the architecture consists of an extensive software library of data types and associated operations commonly applied in multimedia processing. To allow for fully sequential implementations, the library's application programming interface is made identical to that of Horus, an existing sequential library.

6.4.2 Results

We utilized the ALOI collection [41] for evaluation of object recognition performance. For each object, 49 different appearance variations are evaluated. The 49 variations consist of: 12 illumination color variations, 13 rotated views of the object and 24 different illumination directions. Object recognition requires reference images and query images. The reference images are the ones recorded with white illumination and frontal camera with all lights turned on. The 49 query images per object are all matched against the 1,000 reference images, making a total of 49,000 queries.

We compare our method with a standard work in object recognition [45]. This method uses

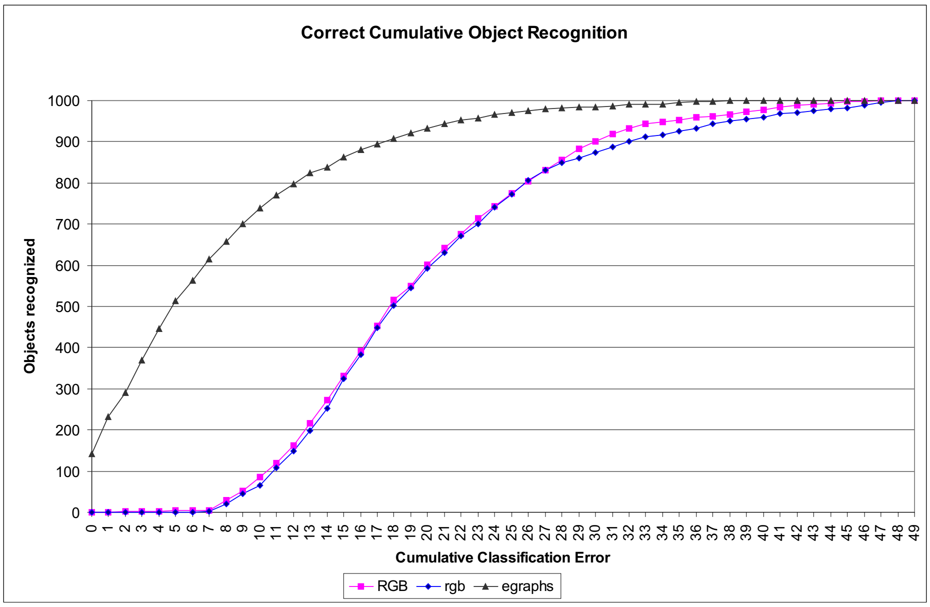


Figure 6.6: Correct Cumulative object recognition. The number of object correctly recognized for an increasing error tolerance. The legend indicates different experiments with *RGB* for histogram intersection on RGB values, *rgb* for histogram intersection on normalized RGB values and *egraphs* for our entropic graph algorithm.

histogram intersection on color invariant pixel values. The number of bins that is used for the histograms is 32 per dimension, which is identical to value used in the original article. Object recognition results on the ALOI collection are computed for *RGB* histograms and for normalized *rgb* histograms.

Figure 6.6 shows the number of objects correctly recognized for an increasing error tolerance. Each of the 49 viewing condition gives rise to a possible mistake. Therefore, the graph displays the number of objects perfectly recognized if we allow 0 errors, to 1000 objects recognized if we allow all 49 mistakes. A desirable graph starts high and has a steep ascend. Our method starts at 141 objects and for a 5% error (2 errors) 291 objects are recognized. For histogram intersection no objects are recognized perfectly. Furthermore, it doesn't matter much if RGB or normalized rgb is used. However, the object recognition results based on entropic graphs significantly outperforms color histograms.

To acquire some insight in the results for both object recognition methods, we analyzed the recognition rate for each of the 49 viewing conditions. Figure 6.7 shows the object recognition performance of both methods grouped by color temperature and rotation direction. See figure 6.3 and figure 6.4 for examples of these conditions. Note the considerable increase in recognition error for both methods under changes in illumination color ($i250, ..., i110$). Hence, both methods are not color constant, where the normalized color histograms suffer the most. Under different viewing angles ($r30, ..., r330$) our proposed method shows a high degree of robustness. The error for histogram intersection under different angles does not favor normalized or raw RGB values. Figure 6.8 shows the object recognition performance of both methods for each camera and illumination direction. See figure 6.5 for examples of these conditions. For the lighting directions l1 and l5 performance degrades for both methods. This result is to be expected as the light shines only on a small part of the object. Performance further decreases as the position of the camera ($c1$ vs $c3$) is farther away from the frontal position, where camera 3 is particulary difficult for the histogram based method. The raw RGB histograms suffer most from changes in lighting directions, which is to be expected as no steps are taken to account for intensity changes. The results are summarized in table 6.1. For all experiments, our method significantly outperforms the histogram based methods.

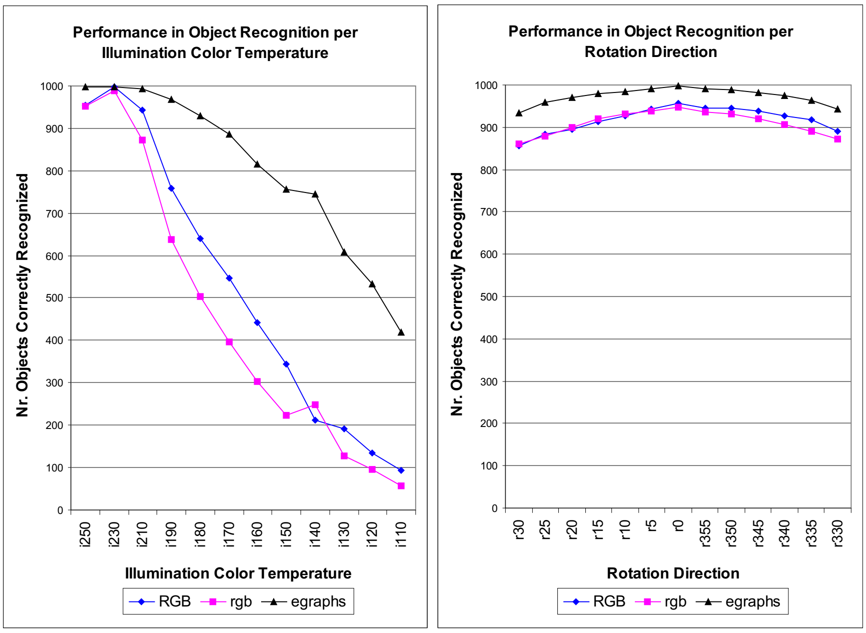


Figure 6.7: Number of objects recognized, grouped by color temperature and rotation direction. The conditions are abbreviated with letters. The prefix i indicates illumination color and r represents degrees of rotation. The legend indicates different experiments with RGB for histogram intersection on RGB values, rgb for histogram intersection on normalized RGB values and *egraphs* for our entropic graph algorithm.

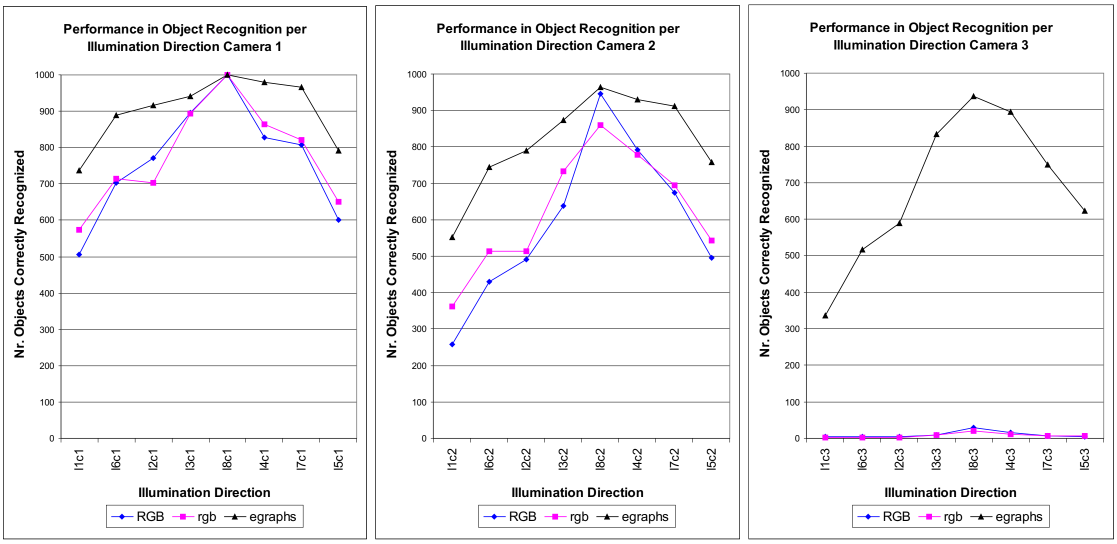


Figure 6.8: Number of objects recognized, grouped by camera and illumination direction. The conditions are abbreviated with letters. The prefix c conforms to camera position and l denotes the light source. The legend indicates different experiments with RGB for histogram intersection on RGB values, rgb for histogram intersection on normalized RGB values and *egraphs* for our entropic graph algorithm.

	RGB	rgb	egraphs
Color Temperature	52.14%	45.06%	80.43%
Rotation	91.85%	91.02%	97.35%
Illumination Direction (c1)	76.36%	77.74%	90.25%
Illumination Direction (c2)	59.04%	62.51%	81.51%
Illumination Direction (c3)	0.99%	0.79%	68.53%

Table 6.1: Percentage correct recognition for each method per condition. RGB is histogram intersection for RGB values, rgb is histogram intersection for normalized RGB, and egraphs is the entropic graph algorithm.

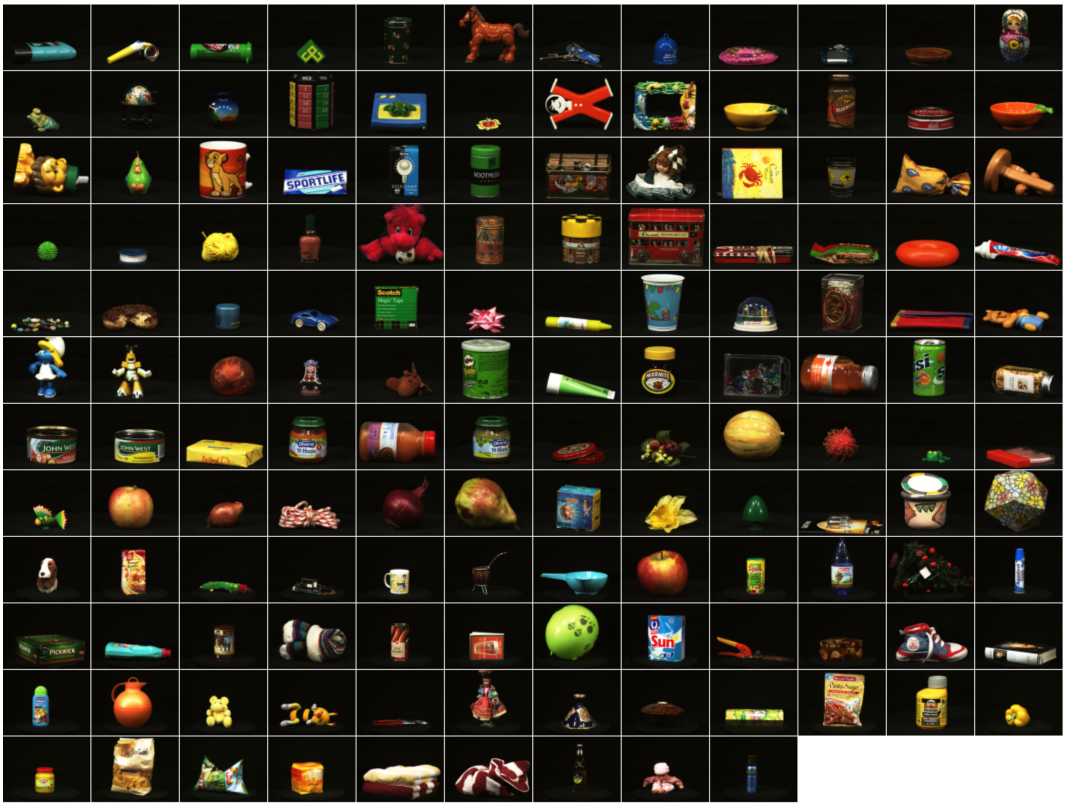


Figure 6.9: 141 ALOI objects perfectly recognized by our method.

For 1,000 objects with 49 viewing conditions per object, we recognize 141 objects perfectly. That is, the number of objects that correctly match all different recordings. Given the diversity in recording circumstances, we may safely assume the objects will be recognized under a high variety of real-life imaging conditions. Figure 6.9 displays the perfectly recognized objects. These objects have no apparent visual similarity, indicating that our approach is not biased towards specific type of objects.

6.5 Discussion and Conclusions

In this Chapter, an unparameterized entropy estimator in combination with color invariant features are used for object recognition. We use color invariant features that keep image measurements constant under varying intensity, viewpoint and shading. For similarity matching we employ a measure based on entropic spanning graphs. Entropic graphs provide an alternative to traditional approaches of image matching such as assuming a fixed probability distribution or histogram binning. The parameters required are the number of nearest neighbors and the value for α in the

α-entropy. The number k of the k-nearest neighbors is not critical, however a higher k adds more robustness. The value of α is set through the power weighting γ, it determines the importance of the tails in a probability distribution. Therefore, α is an additional degree of freedom of the entropy, where $\alpha = 1$ is equivalent to the Shannon entropy. We introduce a new, efficient and exact entropic graph matching algorithm, based on an approximate nearest neighbor algorithm. Despite an efficient algorithm, one drawback to entropic distance measures is that they are computationally more expensive than traditional approaches. Object recognition performance reported on a large dataset show that color invariant entropic graph matching significantly outperforms histogram based methods.

Chapter 7

Summary and Conclusions

7.1 Summary

In this thesis we explore robust and practical methods for visual scene categorization. To this end, we scrutinize and improve the bag-of-visual-words, a.k.a. codebook model. In the codebook model, image features are represented by discrete prototypes, describing an image as a histogram of prototype counts. Prototype-histograms are subsequently used by a classifier to separate images of visual scene categories. In this thesis we focus on the codebook model, and identify four core parts where robust methods may improve the practical application of the model:

1. Prototype vocabulary size and eloquence: the more compact, i.e. smaller, the vocabulary, the larger the image collections that can be indexed. Moreover, a prototype vocabulary may be tuned to the image domain at hand.

2. Image feature sampling: the contextual surroundings of an object may be more informative than the object itself.

3. Prototype to feature assignment: representing an image feature by multiple prototype candidates over merely the best prototype.

4. Classification parameter tuning: careful classification performance estimation may allow more accurate parameter tuning.

In the following paragraphs we summarize the contributions of this thesis per chapter:

Chapter 2, Episode-Constrained Cross-Validation in Video Concept Retrieval. In this Chapter we propose an episode-constrained cross-validation method for estimating scene classification performance in video. The traditional method of cross-validation is based on shots, whereas we propose a method based on episodes. Our episode-constrained method prevents the leaking of nearly identical shots to the rotating hold-out set. Consequently, episode-constrained cross-validation produces sets with an unbalanced number of relevant items. Such unbalances sets apriori have a better Average Precision (AP) score, since AP is not normalized for the number of relevant items. To remedy this bias, we introduce a new performance measure: Balanced Average Precision (BAP). We experimentally compare BAP with AP, and episode-constrained cross-validation with shot-based cross-validation for two classifiers on a large video collection. The results show that the bias of AP for unbalanced data does occur. However, in our dataset, BAP performs equal to AP because the effect does not occur frequently enough in this set. Further experimental evaluation shows that the episode-constrained method yields a more accurate estimate of the classifier performance than the shot-based method. Moreover, when cross-validation is used for parameter optimization, the episode-constrained method is better able to estimate the optimal classifier parameters, resulting in higher performance on validation data compared to the traditional shot based cross-validation.

Chapter 3, Visual Scene Categorization by Learning Image Statistics in Context. We present a scene category classification method by learning the contextual occurrence of proto-concepts like sky, water, vegetation, etc., in images. We compactly represent these proto-concepts by using color invariance and natural image statistics properties. We exploit similarity responses as opposed to strict selection of a codebook vocabulary, and we have been able to generalize these proto-concepts to be applicable in general image collections. We demonstrated the applicability of our approach in a) learning 50 scene categories from a large collection of news video data; b) a collection of 101 categories of images; c) two instances of the Pascal VOC object recognition challenge and d) two large collections of photo-stock images, comprising 89 categories, where categories are learned from one and categorized from the other. An important contribution is scalability, showing that the proposed scheme is effective in capturing visual characteristics for a large class of concepts, over a wide variety of image sets. Where specific methods may have better performance for specific datasets, we have shown a method which is neither tuned nor optimized in parameters for each collection. Hence, the method has proven to robustly categorize scenes from learned context.

Chapter 4, Comparing Compact Codebooks for Visual Categorization. In this Chapter we focus on compact, and thus efficient, models for visual concept categorization. We use the codebook scene classification algorithm where model complexity is determined by the size of the vocabulary. We structurally compared four approaches that lead to compact and expressive codebooks. Specifically, we compared three methods to create a compact vocabulary: 1) global clustering, 2) concept-specific clustering and 3) a semantic vocabulary. The fourth approach increases expressive power by soft-assignment of codewords to image features. We experimentally compared these four methods on a large and standard video collection. The results show that soft-assignment improves the expressive power of the vocabulary, leading to increased categorization performance without sacrificing vocabulary compactness. Further experiments showed that a semantic vocabulary leads to compact vocabularies, while retaining reasonable categorization performance. A concept-specific vocabulary leads to reasonable compact vocabularies, while providing fair visual categorization performance. Given these results, the best method depends at the application at hand. In this Chapter we presented a guideline for selecting a method given the size of the video dataset, the desirability of manual annotation, the amount of available computing power and the desired categorization performance.

Chapter 5, Visual Word Ambiguity. With *visual word ambiguity* we refer to modeling soft-assignment in the codebook model. One inherent component of the codebook model is the assignment of discrete visual words to continuous image features. Despite the clear mismatch of this hard assignment with the nature of continuous features, the approach has been applied successfully for some years. In this Chapter we investigate four types of soft-assignment of visual words to image features. We demonstrate that explicitly modeling visual word assignment ambiguity improves classification performance compared to the hard-assignment of the traditional codebook model. The traditional codebook model is compared against our method for five well-known datasets: 15 natural scenes, Caltech-101, Caltech-256, and Pascal VOC 2007/2008. The results of all experiments show that soft-assignment outperforms the traditional hard assignment over all dimensions, all vocabulary sizes, and over all datasets. We demonstrate that large codebook vocabulary sizes completely deteriorate the performance of the traditional model, whereas the proposed model performs consistently. Moreover, we show that our method profits in high-dimensional feature spaces and reaps higher benefits when increasing the number of image categories.

Chapter 6, Color Invariant Object Recognition using Entropic Graphs. In this Chapter we combine an unparameterized entropy estimator with color invariant features for object recognition. We use color invariant features that keep image measurements constant under varying intensity, viewpoint and shading. For similarity matching we employ a measure based on entropic spanning graphs. Entropic graphs provide an alternative to traditional approaches of image matching such as assuming a fixed probability distribution or histogram binning. The parameters required are the number of nearest neighbors and the value for α in the α-entropy. The number k of the k-nearest neighbors is not critical, however a higher k adds more robustness. The value of α

determines the importance of the tails in a probability distribution. Therefore, α is an additional degree of freedom of the entropy, where $\alpha = 1$ is equivalent to the Shannon entropy. We introduce a new, efficient and exact entropic graph matching algorithm, based on an approximate nearest neighbor algorithm. Object recognition performance reported on a large dataset show that color invariant entropic graph matching significantly outperforms histogram based methods.

7.2 Conclusions and Discussion

This thesis contributes to practical automatic scene classification by endowing the bag-of-visual-words model with more robust properties. The proposed properties increase the classification performance or allow indexing of large, real-world image and video collections. This work allows us to draw the following conclusions.

From chapter 2 we conclude that respecting the contextual narrative structure in video data leads to accurate estimation of classification performance. More accurate estimation, in turn, leads to better parameter selection yielding improved performance. We retain narrative structure by treating a video episode as an atomic element during cross-validation. We imagine that more advanced techniques such as automatic story-segmentation allow more fine-grained atomic story elements. Smaller story units may improve performance estimation by achieving a more diverse spread of stories over the rotating hold-out sets.

Chapters 3 and 4 allow us to conclude that scene context is capable of capturing the global essence of an image. Moreover, a vocabulary of semantic prototypes like sky, water, vegetation, etc., is suitable for many datasets. Such a semantic vocabulary is related to a recently proposed method [31] that describes an image by its attributes such as *has wheel*, *has head*, *is furry*, *is shiny*, etc. Such attributes allow a compositional approach to scene classification, where the meaning of an image is made up of the meaning of its parts. In some sense this is a Homunculus argument, where the problem of image classification is simply postponed to another level. Nevertheless, classification of semantic prototypes or image attributes is intended to be simpler and limited in options for atomic compositional elements. These elements fully determine what the model can 'see'. What compositional elements to choose, remains an open question [82].

Our third conclusion, drawn from chapters 3, 4 and in particular chapter 5, states that in the codebook model the soft-assignment of image features to vocabulary elements is always beneficial when compared to hard-assignment. We have observed this benefit for all vocabulary types: semantic, concept-specific, and generic, for all vocabulary sizes: ranging from extremely small to extremely large, for many datasets: Scene-15, Caltech-101, Caltech-256, the Mediamill challenge and Trecvid collections, and for several image features: SIFT, Wiccest and Gabor features. Soft-assignment reflects ambiguity in the visual word vocabulary. We model this ambiguity with a global similarity function fitting to the image features at hand. However, more advanced methods may readily be applied. We can imagine a classifier's posterior probability, or a learned distance metric that may change depending on its position in feature space.

On the matter of visual vocabulary compactness and large scale image indexing as studied in chapter 4, we conclude that generally there is a compactness *vs.* performance tradeoff. This balance may be tipped somewhat towards better performance by using soft-assignment and a visual word vocabulary that is tuned to the problem at hand. Nevertheless, higher performance comes at a price of less compact models. The choice of how much performance is 'good enough' depends on the application at hand, or alternatively, determined by the size of the dataset and the choice of available hardware.

Our final conclusion from chapter 6 states that entropy-based similarity measures can outperform histogram-based methods. Since a histogram is also used in the codebook model, an obvious extension would be to use an entropic similarity measure instead of a visual word histogram. A somewhat similar approach based on flexible image-to-image matching has shown excellent performance [155]. Removing the histogram would eliminate the need for a visual word vocabulary altogether; and with it the need for ambiguity modeling.

Bibliography

[1] A. Agarwal and B. Triggs. Multilevel image coding with hyperfeatures. *Int. J. Comput. Vision*, 78(1), 2008.

[2] J.A. Aslam, E. Yilmaz, and V. Pavlu. A geometric interpretation of r-precision and its correlation with average precision. In *SIGIR*, 2005.

[3] H.E. Bal et al. The distributed ASCI supercomputer project. *Operating Systems Review*, 34(4):76–96, 2000.

[4] M. Bar. Visual objects in context. *Nature Reviews: Neuroscience*, 5:617–629, August 2004.

[5] D. Batra, R. Sukthankar, and T. Chen. Learning class-specific affinities for image labelling. In *CVPR*, 2008.

[6] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool. Speeded-up robust features (surf). *Comput. Vis. Image Underst.*, 110(3):346–359, 2008.

[7] S. Belongie, J. Malik, and J. Puzicha. Shape matching and object recognition using shape contexts. *IEEE Pattern Analysis and Machine Intelligence*, 24(4):509–522, 2002.

[8] A.C. Berg, T.L. Berg, and J. Malik. Shape matching and object recognition using low distortion correspondences. In *CVPR*, pages 26–33, 2005.

[9] C.M. Bishop. *Pattern Recognition and Machine Learning.* Springer, August 2006.

[10] D.M. Blei, A.Y. Ng, and M.I. Jordan. Latent dirichlet allocation. *J. Mach. Learn. Res.*, 3:993–1022, 2003.

[11] O. Boiman, E. Shechtman, and M. Irani. In defense of nearest-neighbor based image classification. In *CVPR*, 2008.

[12] A. Bosch, A. Zisserman, and X. Munoz. Image classification using random forests and ferns. In *Proceedings of the 11th International Conference on Computer Vision, Rio de Janeiro, Brazil*, 2007.

[13] A. Bosch, A. Zisserman, and X. Munoz. Scene classification using a hybrid generative/discriminative approach. *TPAMI*, 30(4):712–727, 2008.

[14] M. Boutell, J. Luo, and C. Brown. Factor-graphs for region-based whole-scene classification. In *CVPR SLAM Workshop*, 2006.

[15] A.C. Bovik, M. Clark, and W. S. Geisler. Multichannel texture analysis using localized spatial filters. *IEEE Trans. Pattern Anal. Mach. Intell.*, 12(1):55–73, 1990.

[16] M. Brand and V. Kettnaker. Discovery and segmentation of activities in video. *IEEE Pattern Analysis and Machine Intelligence*, 22(8):844–851, 2000.

[17] C.J.C. Burges. A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, 2:121–167, 1998.

[18] G.J. Burghouts and J.M. Geusebroek. Performance evaluation of local colour invariants. *Computer Vision and Image Understanding*, 113:48–62, 2009.

[19] G.J. Burghouts, A.W.M. Smeulders, and J.M. Geusebroek. The distribution family of similarity distances. In *Advances in Neural Information Processing Systems*, 2007.

[20] O. Chum and A. Zisserman. An exemplar model for learning object classes. In *CVPR*, 2007.

[21] C. Cortes and M. Mohri. Auc optimization vs. error rate minimization. In *NIPS*, 2004.

[22] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *In CVPR*, pages 886–893, 2005.

[23] J. Davis and M. Goadrich. The relationship between precision-recall and roc curves. In *ICML*, 2006.

[24] L-Y. Duan, M. Xu, X-D. Yu, and Q. Tian. A unified framework for semantic shot classification in sports video. *Trans. on Multimedia*, 7(6), 2005.

[25] R.O. Duda, P.E. Hart, and D.G. Stork. *Pattern Classification.* Wiley-Interscience Publication, 2000.

[26] F. Ennesser and G. Medioni. Finding Waldo, or focus of attention using local color information. *PAMI*, 17:805–809, 1995.

[27] C.G.M. Snoek et al. The MediaMill TRECVID 2008 semantic video search engine. In *Proceedings of the 6th TRECVID Workshop*, Gaithersburg, USA, November 2008.

[28] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2007 (VOC2007) Results. http://www.pascal-network.org/challenges/VOC/voc2007/workshop/index.html.

[29] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2008 (VOC2008) Results. http://www.pascal-network.org/challenges/VOC/voc2008/workshop/index.html.

[30] M. Everingham, A. Zisserman, C. K. I. Williams, and L. Van Gool. The PASCAL Visual Object Classes Challenge 2006 (VOC2006) Results. http://www.pascal-network.org/challenges/VOC/voc2006/results.pdf.

[31] A. Farhadi, I. Endres, D. Hoiem, and D.A. Forsyth. Describing objects by their attributes. In *CVPR*, pages 1778–1785, 2009.

[32] L. Fei-Fei, R. Fergus, and P. Perona. Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. In *WGMBV*, 2004.

[33] L. Fei-Fei and P. Perona. A bayesian hierarchical model for learning natural scene categories. In *CVPR*, 2005.

[34] R. Fergus, P. Perona, and A. Zisserman. Object class recognition by unsupervised scale-invariant learning. In *CVPR*, 2003.

[35] R. Fisher. The use of multiple measurements in taxonomic problems. *Annals of Eugenics*, 7:179–188, 1936.

[36] D.A. Forsyth. A novel algorithm for color constancy. *International Journal of Computer Vision*, 5(1):5–36, 1990.

[37] B.V. Funt and G.D. Finlayson. Color constant color indexing. *IEEE Pattern Analysis and Machine Intelligence*, 17(5):522–529, 1995.

[38] S. Gao and Q. Sun. Improving semantic concept detection through optimizing ranking function. *IEEE Trans. on Multimedia*, 9(7), 2007.

[39] J.M. Geusebroek. Compact object descriptors from local colour invariant histograms. In *British Machine Vision Conference*, volume 3, pages 1029–1038, 2006.

[40] J.M. Geusebroek. Compact object descriptors from local colour invariant histograms. In *British Machine Vision Conference*, 2006.

[41] J.M. Geusebroek, G.J. Burghouts, and A.W.M. Smeulders. The Amsterdam library of object images. *International Journal of Computer Vision*, 61(1):103–112, January 2005.

[42] J.M. Geusebroek and A.W.M. Smeulders. A six-stimulus theory for stochastic texture. *IJCV*, 62(1/2):7–16, 2005.

[43] J.M. Geusebroek, R. van den Boomgaard, A.W.M. Smeulders, and A. Dev. Color and scale: The spatial structure of color images. In D. Vernon, editor, *Sixth Europian Conference on Computer Vision (ECCV)*, volume 1, pages 331–341. Springer Verlag (LNCS 1842) Berlin, 2000.

[44] J.M. Geusebroek, R. van den Boomgaard, A.W.M. Smeulders, and H. Geerts. Color invariance. *IEEE Pattern Analysis and Machine Intelligence*, 23(12):1338–1350, 2001.

[45] T. Gevers and A. W. M. Smeulders. Color-based object recognition. *Pattern Recognition*, 32(3):453–464, 1999.

[46] T. Gevers and A.W.M. Smeulders. Pictoseek: Combining color and shape invariant features for image retrieval. *IEEE transactions on Image Processing*, 9(1):102–119, 2000.

[47] T. Gevers and H. Stokman. Robust histogram construction from color invariants for object recognition. *IEEE Pattern Analysis and Machine Intelligence*, 26(1):113–117, 2004.

[48] K. Grauman and T. Darrell. Pyramid match hashing: Sub-linear time indexing over partial correspondences. In *Proc. CVPR 2007*, June 2007.

[49] G. Griffin, A. Holub, and P. Perona. Caltech-256 object category dataset. Technical Report UCB/CSD-04-1366, California Institute of Technology, 2007.

[50] A. Hanjalic, R.L. Lagendijk, and J. Biemond. Automated high-level movie segmentation for advanced video-retrieval systems. *Trans. on Circuits and Systems for Video Technology*, 9(4), 1999.

[51] C. Harris and M. Stephens. A combined corner and edge detection. In *Proceedings of The Fourth Alvey Vision Conference*, pages 147–151, 1988.

[52] A.G. Hauptmann, M.-Y. Chen, M. Christel, W.-H. Lin, and J. Yang. A hybrid approach to improving semantic extraction of news video. *icsc*, 2007.

[53] N. Henze and M. Penrose. On the multivariate runs test. *Annals of Statistics*, 27:290–298, 1999.

[54] A. Hero, B. Ma, O. Michel, and J. Gorman. Applications of entropic spanning graphs. *IEEE Signal Processing Magazine*, 19(5):85–95, 2002.

[55] M.A. Hoang, J.M. Geusebroek, and A.W.M. Smeulders. Color texture measurement and segmentation. *Signal Processing*, 85(2):265–275, 2005.

[56] T. Hofmann. Unsupervised learning by probabilistic latent semantic analysis. *Machine Learning*, 42(1/2):177–196, 2001.

[57] D. Hoiem, A. Efros, and M. Hebert. Putting objects in perspective. In *CVPR*, 2006.

[58] A.D. Holub, M. Welling, and P. Perona. Combining generative models and fisher kernels for object class recognition. In *ICCV*, 2005.

[59] B. Huurnink and M. de Rijke. Exploiting redundancy in cross-channel video retrieval. In *ACM Multimedia-MIR*, 2007.

[60] H. Jégou, M. Douze, and C. Schmid. On the burstiness of visual elements. In *CVPR*, 2009.

[61] Y-G. Jiang, C-W. Ngo, and J. Yang. Towards optimal bag-of-features for object categorization and semantic video retrieval. In *CIVR*, 2007.

[62] F. Jurie and B. Triggs. Creating efficient codebooks for visual recognition. In *ICCV*, pages 604–610, 2005.

[63] T. Kadir and M. Brady. Saliency, scale and image description. *International Journal of Computer Vision*, 45(2):83–105, 2001.

[64] L. Kennedy and S-F. Chang. A Reranking Approach for Context-based Concept Fusion in Video Indexing and Retrieval. In *ACM International Conference on Image and Video Retrieval*, Amsterdam, Netherlands, July 2007.

[65] K. Kise, K. Noguchi, and M. Iwamura. Simple representation and approximate search of feature vectors for large-scale object recognition. In *BMVC07*, 2007.

[66] S. Kumar and M. Hebert. A hierarchical field framework for unified context-based classification. In *International Conference on Computer Vision*, 2005.

[67] D. Larlus and F. Jurie. Category level object segmentation. In *International Conference on Computer Vision Theory and Applications*, 2007.

[68] S. Lazebnik, C. Schmid, and J. Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, volume 2, pages 2169–2178, 2006.

[69] T. Lindeberg. Feature detection with automatic scale selection. *Int. J. Comput. Vision*, 30(2):79–116, 1998.

[70] J. Liu and M. Shah. Scene modeling using co-clustering. In *ICCV*, 2007.

[71] D.G. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 60:91–110, 2004.

[72] L. Lu, H. Jiang, and H.J. Zhang. A robust audio classification and segmentation method. In *ACM Multimedia*, 2001.

[73] J. Luo and M.R. Boutell. Natural scene classification using overcomplete ica. *Pat. Rec.*, 38(10):1507–1519, 2005.

[74] M. Marszałek, C. Schmid, H. Harzallah, and J. van de Weijer. Learning object representations for visual object class recognition. In *Visual Recognition Challenge workshop, in conjunction with ICCV*, 2007.

[75] J. Matas, O. Chum, U. Martin, and T. Pajdla. Robust wide baseline stereo from maximally stable extremal regions. In *Proceedings of British Machine Vision Conference*, volume 1, pages 384–393, London, 2002.

[76] B.W. Mel. SEEMORE: Combining color, shape and texture histogramming in a neurally-inspired approach to visual object recognition. *Neural Computation*, 9(4):777–804, 1997.

[77] K. Mikolajczyk, B. Leibe, and B. Schiele. Multiple object class detection with a generative model. In *CVPR*, 2006.

[78] K. Mikolajczyk and J. Matas. Improving descriptors for fast tree matching by optimal linear projection. In *Proceedings of IEEE International Conference on Computer Vision*, 2007.

[79] K. Mikolajczyk and C. Schmid. Indexing based on scale invariant interest points. In *ICCV*, 2001.

[80] K. Mikolajczyk and C. Schmid. A performance evaluation of local descriptors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(10):1615–1630, 2005.

[81] K. Mikolajczyk, T. Tuytelaars, C. Schmid, A. Zisserman, J. Matas, F. Schaffalitzky, T. Kadir, and L. Van Gool. A comparison of affine region detectors. *International Journal of Computer Vision*, 65(1/2):43–72, 2005.

[82] A. Mojsilović, J. Gomes, and B. Rogowitz. Semantic-friendly indexing and quering of images based on the extraction of the objective semantic cues. *Int. J. Comput. Vision*, 56(1-2), 2004.

[83] F. Moosmann, E. Nowak, and F. Jurie. Randomized clustering forests for image classification. *TPAMI*, 2008. to appear.

[84] F. Moosmann, B. Triggs, and F. Jurie. Fast discriminative visual codebooks using randomized clustering forests. In B. Schölkopf, J. Platt, and T. Hoffman, editors, *NIPS*, pages 985–992. MIT Press, Cambridge, MA, 2006.

[85] H. Müller, S. Marchand-Maillet, and T. Pun. The truth about corel - evaluation in image retrieval. In *CIVR*, pages 38–49, 2002.

[86] H. Müller, W. Müller, D.M. Squire, S. Marchand-Maillet, and T. Pun. Performance evaluation in content-based image retrieval: overview and proposals. *Pattern Recogn. Lett.*, 22(5), 2001.

[87] K. Murphy, A. Torralba, and W.T. Freeman. Using the forest to see the trees: A graphical model for recognizing scenes and objects. In *Advances in Neural Information Processing Systems 16*, 2004.

[88] M.R. Naphade and T.S. Huang. A probabilistic framework for semantic video indexing, filtering, and retrieval. *Trans. on Multimedia*, 3(1), 2001.

[89] H. Neemuchwala and A. O. Hero. Image registration in high dimensional feature space. In *Proc. of SPIE Conference on Electronic Imaging*, San Jose, 2005.

[90] S. A. Nene and S. K. Nayar. A simple algorithm for nearest neighbor search in high dimensions. *IEEE Pattern Analysis and Machine Intelligence*, 19(9):989–1003, 1997.

[91] NIST. TRECVID Video Retrieval Evaluation, 2001–2007.

[92] D. Nistér and H. Stewénius. Scalable recognition with a vocabulary tree. In *CVPR '06: Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 2161–2168, Washington, DC, USA, 2006. IEEE Computer Society.

[93] E. Nowak, F. Jurie, and B. Triggs. Sampling strategies for bag-of-features image classification. In *European Conference on Computer Vision*. Springer, 2006.

[94] A. Oliva and A. Torralba. Modeling the shape of the scene: A holistic representation of the spatial envelope. *IJCV*, 42(3):145–175, 2001.

[95] F. Perronnin. Universal and adapted vocabularies for generic visual categorization. *IEEE Trans. Pattern Anal. Mach. Intell.*, 30(7):1243–1256, 2008.

[96] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman. Lost in quantization: Improving particular object retrieval in large scale image databases. In *CVPR*, 2008.

[97] J. Philbin, M. Isard, J. Sivic, and A. Zisserman. Object retrieval with large vocabularies and fast spatial matching. In *In Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 2007.

[98] M. Pontil and A. Verri. Support vector machines for 3d object recognition. *IEEE Pattern Analysis and Machine Intelligence*, 20(6):637–646, 1998.

[99] Y. Qi, A. Hauptmann, and T. Liu. Supervised classification for video shot segmentation. In *ICME*, July 2003.

[100] P. Quelhas, F. Monay, J.M Odobez, D. Gatica-Perez, and T. Tuytelaars. A thousand words in a scene. *IEEE Trans. Pattern Anal. Mach. Intell.*, 29(9):1575–1589, 2007.

[101] P. Quelhas, F. Monay, J.M. Odobez, D. Gatica-Perez, T. Tuytelaars, and L. Van Gool. Modeling scenes with local descriptors and latent aspects. In *ICCV*, 2005.

[102] V. Raghavan, P. Bollmann, and G.S. Jung. A critical investigation of recall and precision as measures of retrieval system performance. *Trans. Inf. Syst.*, 7(3):205–229, 1989.

[103] M. Rautiainen, T. Seppänen, and T. Ojala. On the significance of cluster-temporal browsing for generic video retrieval: a statistical analysis. In *ACM Multimedia*, 2006.

[104] A. Rényi. On measures of entropy and information. In *Proceedings of the 4th Berkeley Symposium on Mathematical Statistics and Probability*, volume I, pages 547–561, University of California Press, Berkeley, 1961.

[105] G. Salton and M. McGill. *Introduction to Modern Information Retrieval.* McGraw-Hill, 1983.

[106] B. Schiele and J. L. Crowley. Recognition without correspondence using multidimensional receptive field histograms. *International Journal of Computer Vision*, 36(1):31–50, 2000.

[107] C. Schmid and R. Mohr. Local grayvalue invariants for image retrieval. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(5):530–534, May 1997.

[108] F.J. Seinstra, J.M. Geusebroek, D. Koelma, C.G.M. Snoek, Marcel Worring, and A.W.M. Smeulders. High-performance distributed image and video content analysis with parallel-horus. *IEEE Multimedia*, 14(4), 2007.

[109] F.J. Seinstra and D. Koelma. User transparency: A fully sequential programming model for efficient data parallel image processing. *Concurrency and Computation: Practice & Experience*, 16(6):611–644, 2004.

[110] T. Serre, L. Wolf, and T. Poggio. Object recognition with features inspired by visual cortex. In *CVPR*, pages 994–1000, 2005.

[111] S.A. Shafer. Using color to separate reflection components. *Color Research and Applications.*, 10(4):210–218, 1985.

[112] E. Shechtman and M. Irani. Matching local self-similarities across images and videos. In *IEEE Conference on Computer Vision and Pattern Recognition 2007 (CVPR'07)*, June 2007.

[113] B. W. Silverman and P. J. Green. *Density Estimation for Statistics and Data Analysis.* Chapman and Hall, London, 1986.

[114] J. Sivic and A. Zisserman. Video Google: A text retrieval approach to object matching in videos. In *ICCV*, volume 2, pages 1470–1477, October 2003.

[115] A.F. Smeaton, P. Over, and W. Kraaij. Evaluation campaigns and trecvid. In *Multimedia Information Retrieval*, pages 321–330, 2006.

[116] A.F. Smeaton, P. Over, and W. Kraaij. High-Level Feature Detection from Video in TRECVid: a 5-Year Retrospective of Achievements. In Ajay Divakaran, editor, *Multimedia Content Analysis, Theory and Applications*, pages 151–174. Springer Verlag, Berlin, 2009.

[117] A. W. M. Smeulders, J. M. Geusebroek, and T. Gevers. Invariant representation in image processing. In *IEEE International Conference on Image Processing*, volume III, pages 18–21. IEEE Computer Society, 2001.

[118] A. W. M. Smeulders, J. C. van Gemert, J. M. Geusebroek, C. G. M. Snoek, and M. Worring. Browsing for the national dutch archive. In *International Symposium on Communications, Control and Signal Processing (ISCCSP)*, 2006.

[119] A.W.M. Smeulders, M. Worring, S. Santini, A. Gupta, and R. Jain. Content based image retrieval at the end of the early years. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(12):1349–1380, 2000.

[120] C.G.M. Snoek, M. Worring, D.C. Koelma, and A.W.M. Smeulders. A learned lexicon-driven paradigm for interactive video retrieval. *IEEE Transactions on Multimedia*, 9(2):280–292, February 2007.

[121] C.G.M. Snoek, M. Worring, J.C. van Gemert, J.M. Geusebroek, and A.W.M. Smeulders. The challenge problem for automated detection of 101 semantic concepts in multimedia. In *ACM Multimedia*, 2006.

[122] C.G.M. Snoek, Marcel Worring, J.M. Geusebroek, D.C. Koelma, F.J. Seinstra, and A.W.M. Smeulders. The semantic pathfinder: Using an authoring metaphor for generic multimedia indexing. *TPAMI*, 28(10), 2006.

[123] C. Studholme, D. L. G. Hill, and D. J. Hawkes. An overlap invariant entropy measure of 3d medical image alignment. *Pattern Recognition*, 32(1):71–86, january 1999.

[124] E. Sudderth, A. Torralba, W. Freeman, and A. Willsky. Describing visual scenes using transformed dirichlet processes. In *NIPS*, 2005.

[125] E.B. Sudderth, A. Torralba, W.T. Freeman, and A.S. Willsky. Describing visual scenes using transformed objects and parts. *IJCV*, 77(1-3):291–330, May 2008.

[126] M. Tahir, K. van de Sande, J. Uijlings, F. Yan, X. Li, K. Mikolajczyk, J. Kittler, T. Gevers, and A. Smeulders. Surreyuva_srkda method, pascal voc 2008. http://pascallin.ecs.soton.ac.uk/ challenges/VOC/voc2008/workshop/tahir.pdf.

[127] A. Torralba. Contextual priming for object detection. *Int. J. Comput. Vision*, 53(2):169–191, 2003.

[128] T. Tuytelaars and C. Schmid. Vector quantizing feature space with a regular lattice. In *International Conference on Computer Vision*, 2007.

[129] J.R.R. Uijlings, A.W.M. Smeulders, and R.J.H. Scha. What is the spatial extent of an object? In *CVPR*, 2009.

[130] A. Vailaya, M. Figueiredo, A. Jain, and H. Zhang. Image classification for content-based indexing. *IEEE Transactions on Image Processing*, 10(1):117–130, 2001.

[131] A. Vailaya, A.K. Jain, and H.J. Zhang. On image classification: city images vs. landscapes. *Pat. Rec.*, 31(12), 1998.

[132] K.E.A. van de Sande, T. Gevers, and C.G.M. Snoek. Evaluating color descriptors for object and scene recognition. *IEEE Trans. Pattern Analysis and Machine Intelligence*, in press, 2010.

[133] J. van de Weijer, T. Gevers, and J. M. Geusebroek. Edge and corner detection by photometric quasi-invariants. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(4):625–630, 2005.

[134] J.C. van Gemert, G.J. Burghouts, F.J. Seinstra, and J.M. Geusebroek. Color invariant object recognition using entropic graphs. *International Journal of Imaging Systems and Technology*, 16(5):146–153, 2006.

[135] J.C. van Gemert, J.M. Geusebroek, C.J. Veenman, and A.W.M. Smeulders. Kernel codebooks for scene categorization. In *ECCV*, 2008.

[136] J.C. van Gemert, J.M. Geusebroek, C.J. Veenman, C.G.M. Snoek, and A.W.M. Smeulders. Robust scene categorization by learning image statistics in context. In *CVPR-SLAM*, 2006.

[137] J.C. van Gemert, C.G.M. Snoek, C.J. Veenman, and A.W.M. Smeulders. The influence of cross-validation on video classification performance. In *ACM Multimedia*, 2006.

[138] J.C. van Gemert, C.G.M. Snoek, C.J. Veenman, A.W.M. Smeulders, and J.M. Geusebroek. Comparing compact codebooks for visual categorization. *Computer Vision and Image Understanding*, (in press), 2010.

[139] J.C. van Gemert, C.J. Veenman, and J.M. Geusebroek. Episode-constrained cross-validation in video concept retrieval. *IEEE Trans. Multimedia*, 11(4):780– 785, 2009.

[140] J.C. van Gemert, C.J. Veenman, A.W.M. Smeulders, and J.M. Geusebroek. Visual word ambiguity. *IEEE Trans. Pattern Analysis and Machine Intelligence*, in press, 2009.

[141] V.N. Vapnik. *The nature of statistical learning theory.* Springer-Verlag New York, Inc., 1995.

[142] N. Vasconcelos. On the efficient evaluation of probabilistic similarity functions for image retrieval. *IEEE Transactions on Information Theory*, 50(7):1482–1496, 2004.

[143] N. Vasconcelos and A. Lippman. A unifying view of image similarity. In *ICPR*, pages 1038–1041, 2000.

[144] J. Vendrig and M. Worring. Systematic evaluation of logical story unit segmentation. *IEEE Trans. on Multimedia*, 4(4), 2002.

[145] J. Vogel and B. Schiele. Natural scene retrieval based on a semantic modeling step. In *ICVR*, Dublin, Ireland, July 2004.

[146] J. Vogel and B. Schiele. Semantic modeling of natural scenes for content-based image retrieval. *Int. J. Comput. Vision*, 72(2):133–157, 2007.

[147] H. Wactlar, T. Kanade, M. Smith, and S. Stevens. Intelligent access to digital video: The informedia project. *IEEE Computer*, 29(5):46–52, May 1996. Digital Library Intiaive special issue.

[148] M.P. Wand and M.C. Jones. *Kernel Smoothing.* Chapman and Hall, London, 1995.

[149] T. Westerveld and A.P. de Vries. Multimedia retrieval using multiple examples. In *Proceedings of The International Conference on Image and Video Retrieval (CIVR2004)*, Dublin, Ireland, 2004.

[150] J. Winn, A. Criminisi, and T. Minka. Object categorization by learned universal visual dictionary. In *ICCV*, pages 1800–1807, 2005.

[151] J. Yang, M.Y. Chen, and A.G. Hauptmann. Finding person x: Correlating names with visual appearances. In *CIVR*, 2004.

[152] J. Yang and A.G. Hauptmann. Exploring temporal consistency for video analysis and retrieval. In *ACM Multimedia-MIR*, 2006.

[153] L. Yang, R. Jin, C. Pantofaru, and R. Sukthankar. Discriminative cluster refinement: Improving object category recognition given limited training data. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, June 2007.

[154] E. Yilmaz and J.A. Aslam. Estimating average precision with incomplete and imperfect judgments. In *CIKM*, 2006.

[155] J. Zhang, M. Marszałek, S. Lazebnik, and C. Schmid. Local features and kernels for classification of texture and object categories: A comprehensive study. *Int. J. Comput. Vision*, 73(2):213–238, 2007.

Samenvatting[1]

Dit proefschrift is gericht op praktische en robuuste methodes voor geautomatiseerde visuele scene herkenning. Hiervoor wordt de gevestigde algoritmiek verbeterd en uitgebreid. De gangbare methode voor scene herkenning vandaag de dag is het zogenaamde *visuele woord algoritme*. Dit algoritme neemt een vooraf gedefinieerd prototype alfabet en vervangt elk beeld-kenmerk door het best passende prototype. Een histogram van prototype frequenties in een beeld wordt vervolgens, met behulp van automatisch lerend algoritmes, gebruikt om verschillende scenes te herkennen. Dit proefschrift richt zich op de volgende vier onderdelen van het visuele woord algoritme:

1. De grootte en eloquentie van het prototype alfabet: een kleiner alfabet maakt het mogelijk om grotere beeld-collecties te beschrijven. Zo'n verkleining kan worden gerealiseerd door het afstemmen van het alfabet op het beeld domein.

2. Beeld-kenmerk monstering: de contextuele omgeving van een object kan informatiever zijn dan het object zelf.

3. Prototype toekenning aan kenmerken: een beeld-kenmerk kan gerepresenteerd worden door meerdere prototypes, in plaats van alleen het beste prototype.

4. Leer-algoritme afstelling: een zorgvuldige schatting van de uitkomst van een zelf lerend algoritme maakt een betere afstelling mogelijk.

In de volgende paragrafen staan de bijdrages gegroepeerd per hoofdstuk.

Hoofdstuk 2 biedt een verbetering voor de parameter afstelling van classificatie algoritmes in de context van visuele scene herkenning in een grote collectie video's. Deze verbetering stelt een betere meting van de classificatie algoritme nauwkeurigheid voor. Een nauwkeurigheidsmeting laat het classificatie algoritme meerdere malen trainen op een willekeurige verzameling van trainingsbeelden waarna vervolgens de resultaten worden geëvalueerd op een onafhankelijke test verzameling. Normaliter worden beelden willekeurig verspreid over de train en test verzamelingen. Echter, omdat video een verhaalstructuur heeft, komen binnen een video dikwijls nagenoeg identieke beelden voor. Dit heeft tot gevolg dat bij een willekeurige opdeling deze identieke beelden worden verdeeld over de train en test verzamelingen. Hierdoor ontstaat een afhankelijkheid tussen de train en test verzamelingen. Dit hoofdstuk laat ziet dat deze afhankelijkheid de uiteindelijke classificatie nauwkeurigheid nadelig beinvloed, en biedt een oplossing door video's te behandelen als atomair eenheden. Als een consequentie daarvan worden volledige video's willekeurig verdeeld over de train en testverzamelingen in plaats van individuele beelden.

In hoofdstuk 3 wordt een scene herkenningsmethode gepresenteerd die de contextuele voorkomens leert van proto-concepten zoals water, lucht, vegetatie, etc. in beelden. Deze proto-concepten worden compact gerepresenteerd met behulp van kleur-invariantie en beeldstatistiek. Deze proto-concepten worden als een alfabet voor het visuele woord algoritme gebruikt, waardoor er een zekere mate van semantiek aan het algoritme wordt toegevoegd. Verder, gebruiken we een maat van gelijkenis tussen een beeld-kenmerk en alle prototypes, in tegenstelling tot alleen het beste prototype te gebruiken. Het semantische alfabet leent zich om gebruikt te worden als een universeel alfabet, toepasbaar op gevarieerde beeld-collecties. Dit wordt gedemonstreerd door het toe te passen op 50

[1] Summary in Dutch

categorieën nieuws video, 101 beeld categorieën, twee uitgaves van een jaarlijkse beeld-herkennings competitie en twee grote commerciele beeldcollecties waarbij de de ene set op de andere wordt geëvalueerd. Een belangrijke bijdrage van dit hoofdstuk is schaalbaarheid. Er wordt aangetoond dat een semantisch alfabet het mogelijk maakt om grote en gevarieerde beeld-collecties efficiënt kan indexeren en verwerken.

Hoofdstuk 4 bouwt verder op hoofdstuk 3 en richt zich op compacte, en dus efficiënte, modellen voor visuele concept herkenning. In het visuele woord algoritme, wordt de complexiteit van een model bepaald door de grootte van het alfabet. In dit hoofdstuk worden vier methodes vergeleken die ieder een compact en expressief alfabet oplevert: 1) een globaal groeperingsalgoritme voor locale beeld-kenmerken 2) een groepering per concept 3) een semantisch alfabet, en de vierde methode is het evalueren van het toekennen van meerdere alfabet elementen aan een beeld-kenmerk. Deze methodes worden geëvalueerd op een aanzienlijke collectie van video's. De resultaten tonen aan dat meerdere alfabet elementen toekennen uitstekend werkt. Verder, is het kleinste alfabet de semantische methode, en geeft een groepering per concept betere resultaten, maar is iets minder compact. Dit hoofdstuk biedt een leidraad die af hangt van de hoeveelheid data, de soort data, de applicatie, de gewenste hoeveelheid handmatig werk en de beschikbare reken capaciteit.

Hoofdstuk 5 gaat volledig over het toekennen van meerdere alfabet elementen aan een beeld-kenmerk. In dit hoofdstuk worden drie methodes onderzocht om deze zogenaamde visuele woord ambiguïteit te modelleren. Deze methodes worden vergeleken met de traditionele aanpak waarbij alleen het beste element wordt gekozen. Er worden vijf welbekende beeld-collecties geëvalueerd, en de resultaten van alle experimenten laten zien dat het expliciet modelleren van ambiguïteit altijd beter werkt dan alleen de beste kiezen. Dit geldt voor alle visuele woord alfabet groottes, alle groottes van beeld kenmerk beschrijvingen, en voor alle vijf de beeld-collecties. Verder, toont het hoofdstuk aan dat een te groot alfabet is funest voor de resultaten van de traditionele methode maar de voorgestelde methode van ambiguïteit modelleren is hier robuust tegen bestand. Ambiguïteit modelleren werkt het best in hoog-dimensionale beeld-kenmerk ruimtes, en met talrijke categorieën.

Hoofdstuk 6 wijkt af van het visuele woord model, en beschrijft een ongeparametriseerde entropie schatter met kleur-invariantie beeld-kenmerken voor object herkenning. Kleur-invariantie wordt gebruikt om metingen constant te houden onder intensiteit, schaduw en gezichtspunt variaties. Om gelijkenis te bepalen tussen beelden, stellen we methode voor gebaseerd op entropisch omspannende bomen. Dit soort boom-structuren bieden een alternatief voor traditionele methodes waarin vaak een statistisch verdeling wordt aangenomen, of waar een histogram gebruikt wordt. De benodigde parameters zijn het aantal buren k in de omspannende boom, en een waarde α in de α-entropie. De waarde k is niet kritisch, alhoewel een grotere k langzamer is, maar robuustheid oplevert. De α-waarde hangt af van de applicatie, en bepaald het gewicht voor de staarten van een kans verdeling, waar een waarde van $\alpha = 1$ gelijk is aan de standaard Shannon entropie. Dit hoofdstuk presenteert een nieuw, en efficiënt algoritme voor het vinden van buren in een hoog-dimensionale ruimte. De resultaten tonen aan dat de methode gebaseerd op entropie beter werkt dan een histogram gebaseerde methode.

Dankwoord[2]

Mijns inziens gaan wetenschap en creativiteit hand in hand. Velen hebben de inspiratie vlam aangewakkerd, en hiervoor wil ik jullie hartelijk bedanken.

Jan-Mark, je inspireerde me door je indrukwekkende theoretische kennis, praktische insteek, je gedrevenheid en ambitie. Van jou heb ik geleerd dat een vlam niet uit zichzelf brand. Daar moet voor gewerkt worden. Jouw rol als co-promotor vervulde je met een eigen vorm van strenge relaxedheid, die goed bij mij aansluit. Ik kreeg de tijd om te zweven, maar eindigde bij jou altijd met de voeten ferm op de grond als er een deadline gehaald moest worden. (Zo zit ik nu midden in de nacht, voor de deadline van morgen dit dankwoord af te schrijven.).

Arnold, geestverruimende professor uit Amsterdam. Tijdens de afspraken met jou zag ik dingen die ik nog nooit eerder had gezien. Onvermoeibaar stimuleer je mensen om je heen, zonder zelf opgebruikt te worden. Je bent een Daedaleske creativiteit katalysator, waardoor ik me soms in de rol van Icarus waande. Ondanks je miljoenen-projecten en andere organisatorische bezigheden, was de deur naar je kamer maar zelden gesloten.

Cor, je nam de honneurs een jaar waar toen Jan-Mark in Oxford zat, en we klikte meteen. Ware het niet dat je later te druk werd om nog tafeltennis te spelen ;). Van jou heb ik geleerd om wetenschappelijke vragen te stellen. Leuke kleine ideetjes tijdens het samen richting huis fietsen worden zomaar journal papers.

Verder wil ik vier mensen nog specifiek bedanken. Dat ik niet nog langer over mijn proefschrift gedaan heb is te danken aan Arjan, Dennis, Koen en Virginie.

Ook wil ik iedereen bedanken voor alle bezielende activiteiten. Onder andere, bedankt voor East of Eden, The Mill, vrijdagmiddagprojectjes, lees clubje, (ASCI) conferenties, Captein en Co, Tokyo, Ping-pong, Latei, FIS, Cuba, Grieks, Handelse bossen, Parijs, Spiderman, Film festival, pannenkoeken, soepkip, koffiepauzes, waterhole, spelletjes in de kroeg.

Het laatste woord is gereserveerd voor diegenen zonder wie ik het niet had gekund. Marieke, voor het geloof zodat ik het aandurfde. Amber, voor het meedragen tijdens de lange middenweg. Gosia, zonder wie ik het nooit had kunnen afmaken.

[2] Acknowledgements in Dutch