Topic models for image retrieval on large-scale databases

Eva Hörster

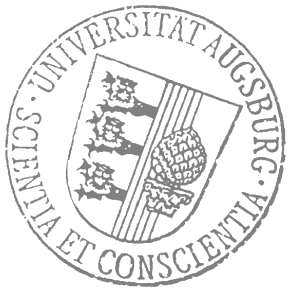
Angaben zur Veröffentlichung / Publication details:

Hörster, Eva. 2009. "Topic models for image retrieval on large-scale databases." Augsburg: Universität Augsburg.

Dissertation

TOPIC MODELS FOR IMAGE RETRIEVAL ON LARGE-SCALE DATABASES

Eva Hörster



Department of Computer Science
University of Augsburg

Adviser:	Prof. Dr. Rainer Lienhart
Readers:	Prof. Dr. Rainer Lienhart
	Prof. Dr. Bernhard Möller
	Prof. Dr. Wolfgang Effelsberg

Thesis Defense: July 14, 2009

Abstract

With the explosion of the number of images in personal and on-line collections, efficient techniques for navigating, indexing, labeling and searching images become more and more important. In this work we will rely on the image content as the main source of information to retrieve images. We study the representation of images by topic models in its various aspects and extend the current models. Starting from a bag-of-visual-words image description based on local image features, images representations are learned in an unsupervised fashion and each image is modeled as a mixture of topics/object parts depicted in the image. Thus topic models allow us to automatically extract high-level image content descriptions which in turn can be used to find similar images. Further, the typically low-dimensional topic-model-based representation enables efficient and fast search, especially in very large databases.

In this thesis we present a complete image retrieval system based on topic models and evaluate the suitability of different types of topic models for the task of large-scale retrieval on real-world databases. Different similarity measure are evaluated in a retrieval-by-example task.

Next, we focus on the incorporation of different types of local image features in the topic models. For this, we first evaluate which types of feature detectors and descriptors are appropriate to model the images, then we propose and explore models that fuse multiple types of local features. All basic topic models require the quantization of the otherwise high-dimensional continuous local feature vectors into a finite, discrete vocabulary to enable the bag-of-words image representation the topic models are built on. As it is not clear how to optimally quantize the high-dimensional features, we introduce different extensions to a basic topic model which model the visual vocabulary continuously, making the quantization step obsolete.

On-line image repositories of the Web 2.0 often store additional information about the images besides their pixel values, called metadata, such as associated tags, date of creation, ownership and camera parameters. In this work we also investigate how to include such cues in our retrieval system. We present work in progress on (hierarchical) models which fuse features from multiple modalities.

Finally, we present an approach to find the most relevant images, i.e., very representative images, in a large web-scale collection given a query term. Our unsupervised approach ranks highest the image whose image content and its various metadata types gives us the highest probability according to a the model we automatically build for this tag.

Throughout this thesis, the suitability of all proposed models and approaches is demonstrated by user studies on a real-world, large-scale database in the context of image retrieval tasks. We use databases consisting of more than 240,000 images which have been downloaded from the public Flickr repository.

Contents

1. Introduction

1.1. Motivation

With the emergence and spread of digital cameras in everyday use, the number of images in personal and on-line collections grows daily. For example the Flickr™ [1] photo repository now consists of more than three billion images [2]. Such huge image databases require efficient techniques for navigating, labeling and searching. At the same time, those Web 2.0 repositories open new possibilities for the statistical analysis and automatic model learning of images for classification and indexing.

Currently, indexing and search of images is mainly based on surrounding text, manually entered tags and/or individual and group usage patterns. However, manually entered tags have the disadvantage of being very subjective and noisy as they usually reflect the author's personal view with respect to the image content. A good example, for instance, is the tag *christmas* in Flickr. Only a fraction of the images depicts the religious event as one might expect. Instead, the tag often denotes the time and date of creation. Thus thousands of vacation and party photos pop up with no real common theme. Moreover there are cases where no associated text is available for the images, as for instance many users do not label their pictures in their personal photo collection. We conclude that image retrieval and indexing solely based on tags/text is difficult.

In this work we put our main focus on a different source of information to retrieve images: the image content. Our analysis will focus on image search and the Flickr repository. Compared to standard image databases, this collection provides a huge amount of annotated training data. On the other hand the annotations are noisy and, compared with standard image databases available for image classification/object recognition tasks, they show very diverse content and objects in all sorts of environments, situations and backgrounds including very cluttered scenes and artistic pictures.

It should be noted however that the majority of the models and concepts presented here could not only be used in the Flickr environment. Our aim is to develop methods that explore such huge databases for learning the models which could as well be used in smaller (e.g.) personal databases.

Thus the main objective of this thesis is to develop models appropriate for representing the image content in the context of retrieval on large scale databases. Besides enabling efficient and

fast retrieval, such models need to be learned automatically, i.e., without supervision. In this work we will study the representation of images by topic models in its various aspects. We will analyze the current models with respect to their suitability in an image retrieval task and extend them.

Probabilistic models with hidden/latent topic variables such as probabilistic Latent Semantic Analysis (pLSA) [40] and Latent Dirichlet Allocation (LDA) [14] and their extensions are popular in the document and language modeling community. Recently they have been introduced and re-purposed for image content analysis tasks such as scene classification [54, 15, 76], object recognition [30, 87, 101], image segmentation [98, 16, 82] and image annotation [12, 68, 7].

In the context of text, hidden topic models model each document in a collection as a distribution over a fixed number of topics. Each topic aims at modeling the co-occurrence of words inside and across the documents and is in turn characterized by a distribution over a fixed size and discrete vocabulary. Applied to visual tasks, the distribution of hidden topics in an image refers to the degree to which an abstract object such as grass, water, sky, street, etc. is contained in the image. This gives rise to a low-dimensional description of the coarse image content and allows us to put images into subspaces for higher-level reasoning which can be used to enable efficient retrieval of images in large databases.

Given unlabeled training images, the parameters, i.e., the probability distributions of the topic models are estimated in a completely unsupervised fashion, which is a huge advantage in large and noisy annotated databases.

1.2. Related Work

The area of content-based image retrieval deals mainly with techniques that enable searching and finding one or more images out of a possibly very large database. We can identify the following sub-areas of images retrieval with respect to their search goal [90]:

- Associative search: The user has no specific result image in mind when searching, only a vague idea of his/her search goal. During searching and browsing (result) images he/she interactively defines what constitutes an appropriate result image. Some examples for interactive image retrieval systems are [25, 80, 95, 51, 108].
- Category search: The user searches images of a specific category. These could be scene images such as a beach during sunset, or specific object classes, for instance cats or flowers, as well as landmark images (e.g. Eiffel tower, Golden Gate bridge).
- Targeted search: The user searches for one special image. He/she has a very precise idea of how the result image has to look, e.g., he/she has already seen it before.

In this thesis we concentrate on category search. Most previous works in this area have only been designed and applied to relatively small and unrealistic image databases ranging from a few

thousand to ten-thousands of images [84, 57, 76, 102]. For example the widely used COREL-database consists of very clean and homogeneous images with almost perfect annotations.

There has been a significant amount of research in image retrieval systems over the last years. Detailed overviews can be found in [81, 90, 33, 21]. Most image search or indexing systems consist of two steps:

1. image content analysis or recognition of all images in the database;
2. subsequent search for similar images based on the extracted content features.

Thus first one or multiple features/models, are extracted from the images for the purpose of content analysis. The aim of such a model or feature is to possibly best represent the image content. In cases where we have a query image, we can then, based on the models/features, retrieve those images whose content is most similar to the query image. Thus an appropriate similarity function based on the model or features used needs to be defined.

Besides the query-by-example scenario, the query could also be given by drawing an example image or by a word, term or tag. We will focus in this thesis on the query-by-example task, although in Chapter 8 we will show an approach to find relevant images given a query term.

1.2.1. Image Features

As stated above, the first necessary step in a retrieval system is to describe the image content by one or multiple features in order to perform retrieval. There are several possible features such as:

- color features (e.g., color histograms [93], color moments [91]),
- texture/structure features (e.g., Gabor-filters [45, 107]), wavelets [104, 24], local edge distributions [62, 3]),
- shape features (e.g., [46]) or
- object detectors and recognizers ([56, 60, 58]).

Some of the features are being computed based on the entire image, i.e., globally, others are computed based on a local image neighborhood. To compute local features, we first need to detect appropriate image regions. This can be done for instance by segmenting the image into objects, detecting interest points [62, 64, 65], detecting known objects [99, 60] or even ignoring the pixel data by e.g. dividing the image into square regions or describing the image by a dense grid over pixels and their respective neighborhood [100, 54]. In the context of segmenting objects there are some works that aim to annotate image regions [86, 17, 27]. This automatic region annotation requires annotated training data and enables image search based on terms similar to text search. Examples of systems that use automatic annotation are [12, 7, 55]. However, object segmentation is a very challenging task due to high object and background variability and thus existing approaches for object segmentation work reliably only on very limited image

databases.

Computing features at local interest points [84, 62] has nowadays become popular in many computer vision applications such as scene and object recognition. In this work we will focus on this type of feature to build our models for representation and thus to characterize the images' content. Interest points mark (distinctive) image regions and do not require any form of segmentation. When using those local features we describe an image solely by a list of features that have been computed from the local neighborhood of the detected interest points. Previously, local features have been used for object (class) recognition [4, 70, 62, 53, 87, 74], object categorization [20], and object search in movies [89]. Often those high-dimensional local features are vector quantized to derive a so-called bag-of-words description, i.e., a normalized histogram over local features.

1.2.2. Probabilistic Models

More and more computer vision works combine local features with complex probabilistic models. Mainly they have been developed and examined in the context of object recognition, such as the constellation model [31] that uses semi-supervised learning for object class recognition or the pictorial structures model [32, 29] which describes objects as a collection of local features or parts which are arranged in a deformable geometric configuration.

Recently probabilistic models, originally developed for large text collections, have been adapted to the image domain. Those models use mixtures of so-called hidden topics (as they are not directly observable) that are common among different documents in the collection to describe the coarse document content. Such topics are learned completely unsupervised and enable indexing the documents in the collection efficiently. They are frequently referred to as topic models. The first probabilistic topic model, called probabilistic Latent Semantic Analysis (pLSA), was introduced by Hofmann [40]. Here each document is described by a mixture of topics and in turn each topic is characterized as a distribution over the words in a finite vocabulary. A fixed number of topics is used to model the documents in the database. The pLSA was then extended by Blei et. al. [14] to a fully generative model, called Latent Dirichlet Allocation (LDA). Both models will be reviewed in depth in Chapter 2 of this thesis.

A number of extensions to those basic topic models have been proposed [94, 18, 13, 37, 79]. They incorporate hierarchical structures [67, 11], model the correlation between otherwise independent topics [13], or account for the authors of the respective documents in the database [79].

Topic models have been applied in the image domain mainly in combination with local image features. Applied to visual tasks, the mixture of hidden topics refers to the degree to which a certain object/scene type is contained in the image. However, to be able to apply topic models to images, the usually high-dimensional, continuous local image features need to be quantized

into a fixed size 'visual' vocabulary. Topic models have been applied and extended in various image content analysis tasks: scene recognition in combination with the pLSA [76, 15] and LDA [54] model, object recognition [30, 61, 87], automatic annotation [12, 68, 7], and image segmentation [98, 16, 82]. Extensions of these focus on modeling the special properties of images and objects, especially the object's geometry, i.e. relative and absolute positions of the interest points in images, have been integrated into the models [30, 61, 92, 101, 75]. Modeling of multiple hierarchical topics levels has also been addressed [92, 88].

In contrast to those previous works we concentrate in this thesis on the application and extension of topic models in the context of image retrieval. Here the challenge consists of mapping the low-level local features of an image to semantic concepts, i.e., the shown object classes, scene or object parts. Images can then be modeled by a number of topics and the topic distributions in images can be used for indexing and searching images. Thus we aim to realize this mapping by applying topic models to images.

1.2.3. Databases

In most of the above-mentioned works topic models have only been applied to carefully selected and labeled image databases of relatively small size. The databases consist of very clean and homogeneous images with almost perfect annotations. Examples for such databases are the OT-dataset [71], the Caltech-101 [28] and Caltech-256 [35] databases, the VOC dataset [26] and the COREL database.

In this thesis, our aim is to design and evaluate models that work for real-world noisy databases. In this thesis we will apply topic models to and examine them on real-word databases. We will therefore download a large number of images from Flickr. The resulting databases are large-scale as they consists of hundreds of thousands of images. In contrast to the above-mentioned databases we do not clean them which results in a very diverse set of images that reflects the properties of a real on-line image repository. Besides their size and diversity those databases also differ in their availability when it comes to labeled data. Many of the Flickr images are tagged by their authors. However, those annotations are only reliable in few cases and in general very noisy. This makes evaluations especially difficult as no ground truth is available. Thus we will mainly rely on user studies to judge the suitability of our proposed approaches.

1.3. Contributions

The main contributions of this dissertation can be summarized in nine points:

- **Topic models applied to image retrieval**: We explore the application of different topic models in a content based image retrieval system. Topic models are used to automatically

extract a high-level image content description based on local features appropriate for retrieval. As this image representation is of low dimensionality it is suitable for large-scale retrieval due to its small memory requirements and fast search. Further we evaluate various parameter settings and different distance measures for similarity judgment and perform a competitive comparison between the different topic models examined.

- **Active Learning and topic models**: We examine topic-model-based image representations in an interactive search scenario, more specifically in combination with an active learning algorithm. Retrieval results are further improved by means of a novel preprocessing scheme for data selection that prunes the set of candidate images used during active learning.
- **Local feature evaluation**: We compare different local image features in combination with topic models to determine their suitability in scene recognition and image retrieval tasks.
- **Feature fusion**: We propose three topic models for fusing different types of local features and explore them in a content based image retrieval task. We also examine different local descriptors and their combination with respect to their suitability to model certain image categories.
- **Continuous vocabulary topic models**: We present three extensions to the pLSA which model the visual vocabulary continuously, thus making the quantization step to derive a finite, discrete visual vocabulary superfluous. We present the algorithms for parameter estimation and inference for each proposed model. Further we perform a competitive evaluation of the models in scene recognition and image retrieval tasks. The original pLSA model serves as the baseline and shows that these models improve performance.
- **Deep networks applied to image retrieval**: We exploit deep network models for deriving a low-dimensional description of the coarse image content. Once their parameters have been learned, those networks are fast to apply due to their feed-forward structure. Additionally they offer a multi-level hierarchical image content description. We compare their performance to topic models in a retrieval-by-example task.
- **Modality fusion**: We present work in progress on fusing multiple modalities for image retrieval. We investigate three models. Two of those are hierarchical models, one based on the pLSA, the other based on deep networks. In our experiments we fuse visual features and semantic features based on tags and evaluate the proposed models.
- **Finding relevant images**: We present an approach to find the most relevant images, i.e., very representative images, in a large web-scale collection given a query term. Our approach ranks that image highest whose image content and various metadata types give us the highest probability according to the model we build for this tag. We learn such a model to predict the most relevant images in an unsupervised fashion.

- **Evaluation**: We judge the suitability of all proposed approaches by user studies on a real-world, large-scale database in the context of image retrieval. Our databases consist of more than 240,000 images which have been downloaded from the public Flickr repository.

1.4. Thesis Overview

Based on a review of different topic models in Chapter 2, we will present our retrieval system using a topic-model-based image representation in Chapter 3. We evaluate the system for different topic models and different distance measures for similarity judgment in a retrieval-by-example task. Active learning is applied to the image representations derived to further improve the results, and a novel pre-processing scheme is proposed. Next, in Chapter 4, different local image descriptors are compared in the context of topic models and approaches for fusing multiple types of local features are discussed. We propose and evaluate extensions to the pLSA for modeling the visual vocabulary continuously in Chapter 5. Chapter 6 examines deep networks in the context of image retrieval and compares their performance to topic model based image representations. Approaches for modality fusion are examined in Chapter 7 and in Chapter 8 we present a system to find highly relevant images to a given query term. Finally we summarize the thesis in Chapter 9 and discuss directions for future research work.

2. Topic Models

In this chapter we will review different types of basic topic models such as probabilistic Latent Semantic Analysis (pLSA) and Latent Dirichlet Allocation (LDA). These topic models are a fundamental part of our image retrieval system. Due to their origin from text collection and document modeling, we will review the topic models in this context before we explain how they can be adapted and applied to image databases in the next chapter. Extensive evaluations and certain extensions of the here described models which are relevant for the image case are presented in the subsequent chapters.

All topic models discussed here start from a term-document co-occurrence table for the documents in the database. Assuming a number of M documents $d_1, ..., d_M$ in our database and a finite set of N words $w^1, ..., w^N$, the so-called vocabulary, the entries of this co-occurrence matrix **D** specify the number of times $n(w^j, d_i)$ the j-th word w^j from the vocabulary occurs in the document d_i (see Figure 2.1). Note that a document could be a section, a paragraph or even an entire book. This basic representation ignores the order of words in the documents completely, which is often known as the bag-of-words assumption. Note that this assumption might not hold in many cases.

Such a, usually sparse, co-occurrence matrix can be used directly to compare documents and perform retrieval. However there are several problems with such a document representation in the context of retrieval. A word might have multiple meanings (polysemy), for instance a jaguar is a type of car as well as an animal, and at the same time we use different words for the same object/entity or concept (synonymy), e.g., sometimes we might use the word 'automobile' instead of 'car'. This makes the word count vector, a column of the term document matrix representing the words in a document, noisy and accurate retrieval difficult.

The aim of topic models is to overcome these problems by automatically finding a latent, i.e., hidden, semantic space that is more accurate to model documents in the context of retrieval or similarity tasks. The semantic structure of a document here refers to some underlying hidden concepts, topics or themes that are responsible for the (co-)occurrence of words in documents. Each document is assumed to consist of multiple hidden topics and is represented by their weights. This kind of representation has several advantages. First of all the semantic space is usually of lower dimension than the simple space that arises from using word count vectors. This enables fast search and needs less storage. Moreover the conversion to a semantic space helps reducing noise in the word vectors and addresses the above-mentioned problems of

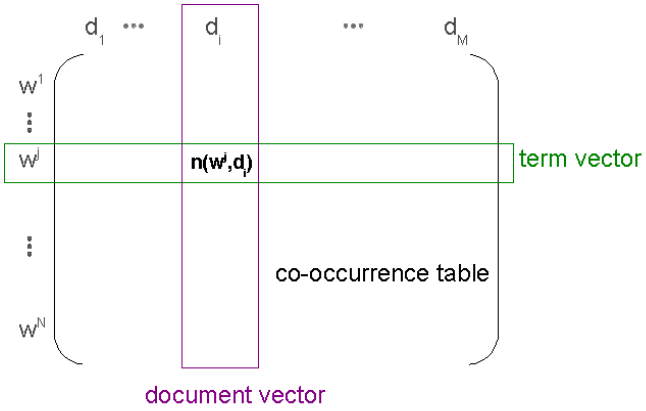


Figure 2.1: Illustration of the $N \times M$ co-occurrence table $\mathbf{D}$. The entries $n(w^j, d_i)$ specify the counts of the word w^j in document d_i.

synonymy and polysemy thus improving e.g. retrieval performance.

We will now first review the Latent Semantic Analysis (LSA) [22] model. It has been the first statistical method which aimed to uncover a latent semantic structure in document collections. Next, in the subsequent sections, we discuss its probabilistic extensions. In the following chapters of this thesis we will then solely focus on the various probabilistic topic models.

2.1. Latent Semantic Analysis (LSA)

To automatically compute the latent semantic structure in a database of documents, Latent Semantic Analysis (LSA) [22] applies singular value decomposition (SVD). SVD transforms the term document representation, i.e., the co-occurrence table, into a relation between some concepts (or topics) and words as well as one between the concepts and the documents. These automatically derived concepts thus represent the latent semantic structure in the data and are used to model the documents in some topic space. Note that the topic space is typically lower dimensional than the word space, and thus LSA performs a dimensionality reduction.

As stated above, the method starts from a term document co-occurrence matrix $\mathbf{D}$ of dimension $N \times M$ where N denotes the number of words that occur across the documents and M denotes the number of documents in the collection. Then we perform SVD:

$$\mathbf{D} = \mathbf{U_0} \Sigma_0 \mathbf{V_0^T} \tag{2.1}$$

where $\mathbf{U_0}$ is a $N \times F$ matrix that describes which words form a specific topic, $\mathbf{V_0}$ has dimension $M \times F$ and identifies which topics a document is built of, and Σ_0 is a diagonal matrix of size $F \times F$. F denotes the rank of $\mathbf{D}$ and the entries of Σ_0 are called singular values σ_f; they characterize the importance of the respective topic. $\mathbf{U_0}$ and $\mathbf{V_0}$ have orthonormal columns. The SVD is

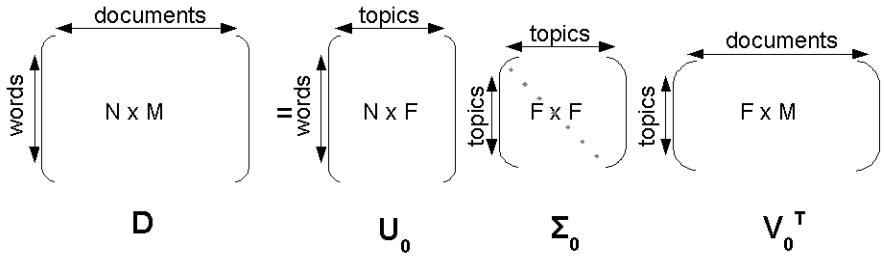


Figure 2.2: Singular value decomposition of the co-occurrence table **D**.

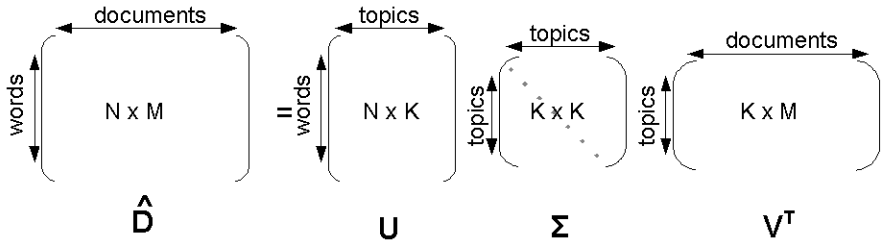


Figure 2.3: Approximation of the co-occurrence table by the LSA model.

depicted in Figure 2.2.

Let's now assume that the singular values σ_f in the matrix Σ_0 are ordered by size. If we only keep the K largest values σ_f, i.e., we set the remaining ones to zero, and then delete the zero rows and columns of Σ_0, we can also eliminate the corresponding columns of $\mathbf{U_0}$ and rows of $\mathbf{V_0}$. Thus we obtain new matrices $\mathbf{U}$ of size $N \times K$, $\mathbf{V}$ of dimension $M \times K$ and the $K \times K$ matrix Σ and for the term document matrix it holds:

$$\hat{\mathbf{D}} = \mathbf{U}\Sigma\mathbf{V}^{\mathbf{T}} \tag{2.2}$$

where $\hat{\mathbf{D}}$ approximates the original co-occurrence matrix $\mathbf{D}$ and is used in the following to represent our data. $\hat{\mathbf{D}}$ is the matrix of rank K that is closest to $\mathbf{D}$ in the least-square sense [22]. Note that the choice of K is important as we want to keep the important structure of the data and at the same time we aim to eliminate noise in our original co-occurrence matrix. The SVD of the matrix $\hat{\mathbf{D}}$ is shown in Figure 2.3.

If we now want to compare the documents in the database with each other in order to find documents that are close together thematically, i.e., in the semantic space, we only need the matrices Σ and $\mathbf{V}$ if we use the dot product to compare documents:

$$\hat{\mathbf{D}}^{\mathbf{T}}\hat{\mathbf{D}} = \mathbf{V}\Sigma^2\mathbf{V}^{\mathbf{T}} \tag{2.3}$$

Thus we can consider the rows of the matrix $\mathbf{V}\Sigma$ as the lower dimensional representations for our documents in the semantic space.

One remaining issue is how to project new documents into this subspace, i.e., how do we find the low-dimensional representations for novel documents which have not been used for training our model. This task is essential in cases where for instance the database is too large to learn the model from all documents and it is instead only trained on a subset. Another case occurs if someone wants to add a novel document to the database.

Thus starting with a term vector, denoted by $\mathbf{d_i}$ we want to find its representation $\mathbf{v_q}$, i.e. its representation in the K-dimensional semantic space. $\mathbf{v_q}$ can be calculated by assuming that $\mathbf{d_i} = \hat{\mathbf{d}}_\mathbf{i}$:

$$\mathbf{v_q} = \mathbf{d_i^T U \Sigma^{-1}} \tag{2.4}$$

Once we have computed $\mathbf{v_q}$, we can treat it simply as an additional row in the $\mathbf{V}$ matrix and use Equation 2.3 to compute its similarity to the other documents in the database in the semantic space.

2.2. probabilistic Latent Semantic Analysis (pLSA)

The probabilistic Latent Semantic Analysis (pLSA) [40] is the probabilistic variant of the LSA. Instead of performing the mapping into the semantic space by SVD, the pLSA assumes an underlying probabilistic model where each document is represented by a mixture of topics. Each topic denotes in turn a distribution over the discrete words. Furthermore, the topics in the model are hidden, i.e., we do not know to which extent which topic is contained in the document or how the topics are defined in terms of probabilities over the words. Nevertheless, the model associates one of the hidden topics with each word observation in a document. Note that while the LSA aims to reconstruct the co-occurrence table in the least square sense, the pLSA aims to optimize the predictive performance of the model.

Assuming we have M documents d_i in our database and a finite vocabulary of size N, we suppose that each document d_i in the collection is represented as a set of N_i words, i.e., we write $\mathbf{w_i} = \{w_1, w_2, ..., w_{N_i}\}$, where w_n denotes the value of the n-th word in the set. Note that this representation can be derived directly from the co-occurrence table by collecting all non-zero entries. The pLSA model then assumes the following generative process for a document d_i [40]:

- Pick a document d_i with prior probability $P(d_i)$
- For each of the N_i words in document d_i:
  - Select a hidden topic label z_n with probability $P(z_n|d_i)$
  - Generate a word w_n with probability $P(w_n|z_n)$

Note that the number of topics K and words N in the model are predefined and usually $K < N$ which means a bottleneck.

The above-described generative process results in the following probability of a word w_j in a

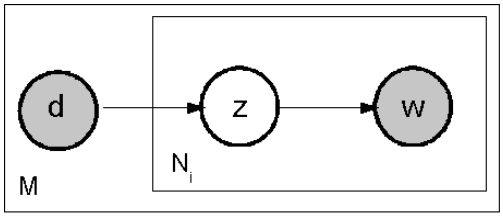


Figure 2.4: Graphical representation of the pLSA model. M denotes the number of documents in the database and N_i the number of words in document d_i. Shaded nodes highlight the observable random variables w for the occurrence of a word and d for the respective document. z denotes the hidden topic variable.

document d_i:

$$P(w_j, d_i) = P(d_i) \sum_{k=1}^{K} P(w_j|z_j = k) P(z_j = k|d_i) \tag{2.5}$$

The likelihood L of the database can then be written as:

$$L = \prod_{i=1}^{M} \prod_{j=1}^{N_i} P(w_j, d_i) \tag{2.6}$$

We can also express the likelihood in terms of the co-occurrence matrix:

$$L = \prod_{i=1}^{M} \prod_{j=1}^{N} P(w^j, d_i)^{n(w^j, d_i)} \tag{2.7}$$

where w^j denotes the j-th word in the vocabulary consisting of N words. Figure 2.4 shows the graphical representation of the pLSA model.

The parameters of the pLSA models, i.e., the probability distributions $P(w|z)$ of the visual words w given a topic z, which thus define the hidden topics, as well as the probability distributions $P(z|d)$ of hidden topics z given an image d, are learned completely unsupervised by means of the Expectation Maximization (EM) algorithm [23]. The EM algorithm iterates between the E- and M-step to find the maximum likelihood estimate of the latent variables. Denoting with z^k the k-th topic, i.e. $z = k$, the E- and M-step for the pLSA are given by:

E-step:

$$P(z^k|d_i, w^j) = \frac{P(z^k|d_i) P(w^j|z^k)}{\sum_k P(z^k|d_i) P(w^j|z^k)} \tag{2.8}$$

M-step:

$$P(w^j|z^k) = \frac{\sum_i n(d_i, w^j) P(z^k|d_i, w^j)}{\sum_j \sum_i n(d_i, w^j) P(z^k|d_i, w^j)} \tag{2.9}$$

$$P(z^k|d_i) = \frac{\sum_j n(d_i, w^j) P(z^k|d_i, w^j)}{N_i} \tag{2.10}$$

Note that the EM algorithm does not ensure the globally optimal solution, the algorithm con-

verges to a local optimum.

Note that the pLSA learns the topic distributions $P(z|d)$ only for the training images. There is no intuitive way to include new documents into the model, this is the reason why the pLSA model is not a fully generative model for documents. However we can estimate the topic distributions for new images that are not part of the original training corpus by a heuristic, the so called fold-in technique [40]. Here the EM algorithm as described above is applied to the unseen images. However, this time the word distributions conditioned on the topic $P(w|z)$ are fixed (i.e., Equation 2.9 is not updated) and only the remaining steps of the EM algorithm are performed in order to compute the topic distribution $P(z|d_l)$ for each novel image d_l.

A further problem of the pLSA model is that it is prone to overfitting as the number of variables increases with the number of documents in the training set. One attempt to solve this problem is to use the tempered EM algorithm [41, 40].

2.3. Latent Dirichlet Allocation (LDA)

The Latent Dirichlet Allocation (LDA) [14] is a generative probabilistic model which is similar to the pLSA model but fully generative. It also represents documents by a finite mixture over latent topics. Similarly each topic is characterized by a distribution over words, and each occurrence of a word in a specific document is associated with one unobservable topic. The main difference to the pLSA model is that topic probabilities can be easily assigned to new documents and at the same time the overfitting problem of the pLSA is avoided. This is realized by treating the topic mixtures as a hidden random variable and placing a Dirichlet prior[1] on the multinomial mixing weights.

We suppose as before that we represent each document i in our collection as a set of N_i words, written as $\mathbf{w_i} = \{w_1, w_2, ..., w_{N_i}\}$, where w_n denotes the value of the n-th word in the set. The LDA model describes the process of generating such a document as follows [14]:

- Choose a K-dimensional Dirichlet random variable $\theta_i \sim Dir(\alpha)$, where K denotes the finite number of topics in the corpus.
- For each of the N_i words in document i:
  - Choose a topic label $z_n \sim Multinomial(\theta_i)$.
  - Sample the value w_n of the n-th word from $P(w_n|z_n, \beta)$, a multinomial probability conditioned on the topic z_n.

[1] A K-dimensional Dirichlet random variable θ has the following probability density function:

$$P(\theta|\alpha) = \frac{\Gamma\left(\sum_{k=1}^{K}\alpha_k\right)}{\prod_{k=1}^{K}\Gamma(\alpha_k)}\prod_{k=1}^{K}\theta_k^{\alpha_k-1}$$

where $\Gamma(x)$ is the Gamma function, $\alpha_k > 0$ are the parameters of the distribution and $0 \leq \theta_k \leq 1$ and $\sum_{k=1}^{K}\theta_k = 1$ holds.

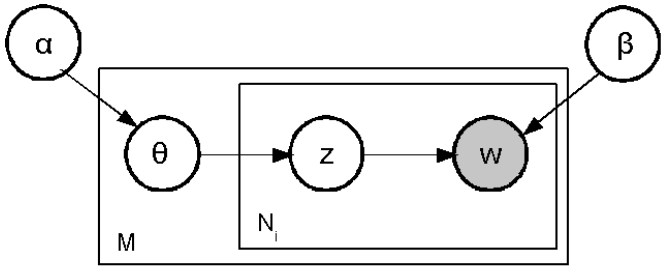


Figure 2.5: Graphical representation of the LDA model. M denotes the number of documents in the database and N_i the number of words in document i. The shaded node denotes the observable random variable w for the occurrence of a word, z denotes the topic variable and θ the topic mixture variable.

The graphical representation of the LDA model is shown in Figure 2.5. Again, M indicates the number of images in the entire database and N_i denotes the number of visual words in image d_i.

The probability of a document $\mathbf{w_i}$ according to the LDA model is given by:

$$P(\mathbf{w_i}|\alpha,\beta) = \int P(\theta_i|\alpha)\prod_{j=1}^{N_i}(\sum_{k=1}^{K} P(z_j = k|\theta)P(w_j|z_j = k,\beta))d\theta \tag{2.11}$$

The likelihood of the complete document collection is then the product of the probabilities of each single document.

Probability distributions of words given a hidden topic as well as probability distributions of hidden topics given the documents are learned in a completely unsupervised manner. We learn an LDA model by finding the corpus parameters α and β such that the log marginal likelihood of a database is maximized. Since this maximization cannot be solved directly, approximate algorithms such as variational inference [14] or Gibbs Sampling [36] are used. In this thesis we will apply variational inference when using the LDA model. Given the learned parameters α and β, we can then perform approximate inference, i.e, for estimating the document level parameters such as the topic mixtures, using also these approaches. Thus we may learn the LDA corpus level parameters on a subset of the database (in order to reduce total training time) and then assign probability distributions to all documents.

2.4. Correlated Topic Model (CTM)

Much like in the LDA model, the Correlated Topic Model (CTM) [13] assumes that each document is composed of words that all arise from a mixture of topics, i.e., documents are represented by finite mixtures over hidden topics. Unlike the LDA, where the topic proportions of a specific document are drawn from a Dirichlet and therefore the correlation between different topics is disregarded, the CTM draws these topic proportions from a logistic normal distribu-

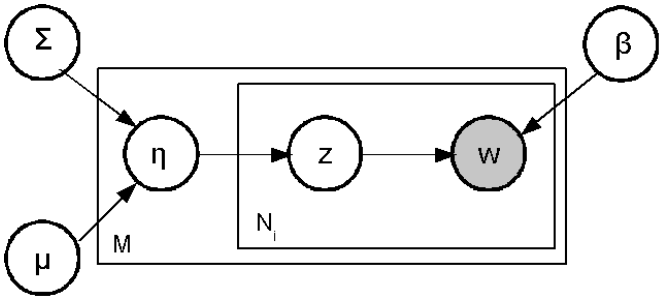


Figure 2.6: Graphical representation of the CTM model. M denotes the number of documents in the database and N_i the number of words in document i. The shaded node denotes the only observable random variable w for the occurrence of a word, z denotes the topic variable and η the topic mixture variable.

tion. That means in detail, to generate the topic proportions for a document, a random vector is drawn from a multivariate Gaussian and then mapped to the simplex to obtain a multinomial parameter. Thus, the covariance of the Gaussian entails dependencies between the elements of the vector.

Assuming that each document i of a corpus/database of M documents is composed of a set of N_i words, we represent the document i by a N_i-dimensional vector $\mathbf{w_i}$ that contains the values w_n of the words the document consists of. Each word in a document is associated with one of the K topics in the model. According to [13], the generative process an N_i-word document i arises from can formally be summarized as follows:

1. Draw $\eta_i|\{\mu,\Sigma\} \sim N(\mu,\Sigma)$, where μ denotes a K-dimensional mean vector and Σ a covariance matrix of size $K \times K$.
2. For each of the N_i words in document i:
   a) Draw topic assignment z_n from $multinomial(f(\eta_i))$.
   b) Choose a word w_n from $P(w_n|z_n,\beta)$ a multinomial probability conditioned on the topic z_n.

and where $f(\eta)$ denotes a mapping of the natural parameterization of the topic proportions to the mean parameterization.

$$\theta = f(\eta) = \frac{\exp\{\eta\}}{\sum_l \exp\{\eta^l\}} \tag{2.12}$$

The graphical representation of the CTM is shown in Figure 2.6.

The only observable variables in the CTM are the words each document consists of. Learning the parameters of such a model given a set of training documents is accomplished by a variational Expectation-Maximization (EM) procedure. Given the learned model we can estimate the topic proportions of a new document by a variational inference algorithm. Details regarding the learning and inference algorithms in the CTM model can be found in [13].

2.5. Summary

In this chapter we reviewed different types of topic models in the context of document modeling. These topic models are a fundamental part of our image retrieval system and we will explain in the next chapter how they can be adapted and applied to image databases. In the subsequent chapters we will extensively evaluate our topic-model-based retrieval system and propose several extensions of the here described models especially suited for the application of topic models in the image domain.

3. Topic-Model-Based Image Retrieval

In the previous chapter we have introduced different types of probabilistic topic models for text collections. We use these topic models throughout this thesis in a different context; for modeling image databases and representing images in image retrieval and scene recognition tasks.

It has been shown in [59] that a pLSA-based image description in a large-scale image retrieval task outperforms conventional methods such as using directly a bag-of-words model on local image features or Color Coherence Vectors (CCVs) [73] as image representations. In [103] the authors show that by using LDA models they are able to improve information retrieval. Inspired by those two works we will in this chapter examine several topic models in the context of image retrieval. In the first section we present our complete retrieval system. We describe the necessary steps to adapt topic models to image collections. Various similarity measures appropriate for topic-model-based image retrieval are then discussed in the second section. The proposed system is evaluated experimentally for different kinds of topic models in Section 3.3.

Furthermore we will show in the fourth section of this chapter how the system can be modified to allow interactive image retrieval by using active learning techniques.

3.1. Retrieval System

The core component of our image retrieval system is a probabilistic topic model. This topic model is used to represent each image by its topic distributions, thus the topic model enables a high-level low-dimensional representation for each image in our database. Once we have found an appropriate representation for each image in our database we can perform query-by-example image retrieval and scene recognition tasks as described below.

As described in the previous section, topic models have been originally developed in the context of text modeling where words are the elementary parts of documents. Documents are modeled as mixtures of intermediate hidden topics, representing the semantic structure of the documents, and topics are characterized by a distribution over words. Applied to image modeling, the images are our documents. The mixture of hidden topics then refers to the degree to which certain objects or certain object parts are contained in an image. It is important to note that topic models allow us explicitly to represent an image as a mixture of topics, i.e., as a mixture of one or more objects/object parts. Since for all currently practical applications the number of

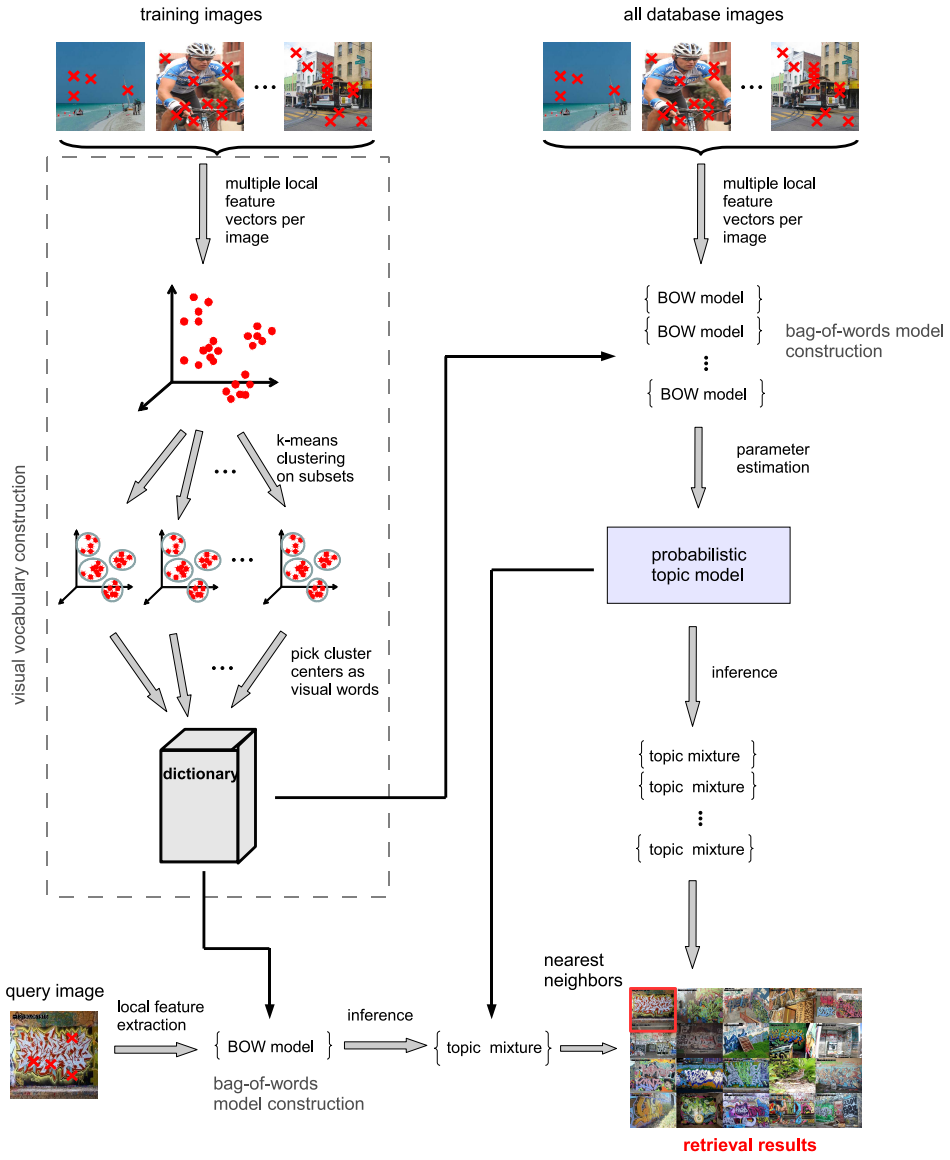


Figure 3.1: Image retrieval system based on a discrete probabilistic topic model.

topics in the model is much smaller than the number of words, the topic distribution gives rise to a compact, low-dimensional description of the coarse image content.

In order to apply the various topic models to image databases an equivalent to words as elementary parts in documents has to be found for images. These elementary parts are commonly known as visual words. To learn visual words we start by automatically extracting local image descriptors at previously detected interest points (also called keypoints) from a set of training images. There exist many types of interest point detectors and local image descriptor combinations, each capturing a different property of a local image region and being more or less invariant to illumination, changes in viewpoint and other image transformations. However the existing descriptors are usually high dimensional (between 80 and 300 dimensions) and their

entries are continuous. Thus to derive discrete visual words those extracted local features are vector quantized.

Several approaches for quantization of local image features exist. Often the k-means algorithm is applied in various forms and the means of the derived clusters are kept as visual words. However, direct application of the k-means algorithm is not efficient, especially if the number of visual words to be computed and thus the number of clusters k in the algorithm is large. Then a large number of training images and extracted features has to be used to reliably compute the clusters. To overcome this efficiency problem we derive visual words by first building small subsets of features out of the huge amount of training features. Then we perform k-means clustering on each subset separately. Assuming we have l subsets and we derive k cluster centers for each subset, our final vocabulary is then built by merging the cluster centers of all subsets resulting in a vocabulary size of $l \cdot k$. Another approach to build a large vocabulary is hierarchical k-means clustering which builds a vocabulary tree [69].

Given the determined discrete and finite vocabulary, local features are extracted from each image in the database. Each image d_i in our image database is then represented as consisting of N_i visual words by replacing each detected feature vector by its most similar visual word. The most similar visual word to a specific feature vector is defined as the closest word in the high-dimensional feature space. A term frequency vector for each image can then be computed by counting the visual word occurrences in the images, and thus a co-occurrence table or equivalently a bag-of-words model may be derived.

This bag-of-words image model can then be used as input to the previously described probabilistic topic models in order to derive a high-level image representation. Note that the spatial relationships between the occurrences of different visual words in images is completely disregarded in the bag-of-words image description and therefore also in the subsequently computed probabilistic topic models. It should be noted that instead of using a bag-of-words image representation based on local features, one could as well use any type of count vector as the input to our probabilistic topic models, e.g., one derived from global image features. However, the computed topics will then have to be interpreted differently.

Based on the bag-of-visual-words image descriptions a topic model is trained completely unsupervised as described in the previous chapter. Note that such a model is often learned on a subset of the images in the database only, as otherwise training is computationally too expensive. After the model has been learned it is applied to all images in the database and thus a topic distribution for each image is derived. This topic distribution is in the following used to represent the image, providing a high-level description of the image content. In most cases the number of topics is chosen to be smaller than the number of visual words in the model, thus by describing our images using the discrete topic distributions we obtain a dimensionality reduction as well. This dimensionality reduction is especially advantageous if we go to very large databases, as storage space for the image descriptions as well as retrieval time can be significantly reduced.

Once we have trained a probabilistic topic model and computed a topic representation for each image in the database, we need to define an image similarity measure in order to perform retrieval. In this work, we focus on the task of query-by-example retrieval, thus searching in the database for the most similar items to a given query image. Various similarity measures are presented in the next section. Having found the most similar images to the query image we show the retrieval results to the user.

Figure 3.1 gives a complete system overview of our proposed topic model based image retrieval system.

Our query-by-example retrieval system only needs a small modification to be able to perform scene recognition. Here we can perform a simple k-Nearest Neighbor (kNN) search for the unlabeled test images over labeled training images. This approach is similar to the scene classification system proposed by Bosch et. al. [15], who use a pLSA model to describe images. However, for solving such a recognition task labeled training images are necessary in order to perform kNN classification or to train any other classifier, such as Support Vector Machines (SVMs), Random Forrest (RF), or Adaboost, on the computed topic distributions as image representation.

3.2. Similarity Measures

An essential part of our retrieval system is the distance measure used to determine the similarity of two images represented by their topic mixtures. The topic mixture for each image indicates to which extent a certain topic is contained in the respective image. Based on the topic mixtures, we look at four different ways to measure similarity. These measures are experimentally evaluated in the next section.

First the similarity between two images d_a and d_b can be measured by calculating the cosine similarity between the their topic distributions. The cosine $\cos(\mathbf{a},\mathbf{b})$ between two vectors $\mathbf{a}$ and $\mathbf{b}$, here representing the topic mixtures of d_a and d_b, is popular in text retrieval [6] and is defined by:

$$cos(\mathbf{a},\mathbf{b}) = \frac{\mathbf{a}\cdot\mathbf{b}}{\|\mathbf{a}\|\cdot\|\mathbf{b}\|} \tag{3.1}$$

Thus, similarity is defined as the cosine of the angle between the two vectors in the topic space.

A second possibility to measure image similarity is the use of the $L1$ distance between two topic distributions. The $L1$ distance between to K dimensional vectors $\mathbf{a}$ and $\mathbf{b}$ is given by:

$$L1(\mathbf{a},\mathbf{b}) = \sum_{k=1}^{K} |a_k - b_k| \tag{3.2}$$

The third similarity measure that we study is the Jensen-Shannon divergence $JS()$ between the

discrete topic distributions of two images. The JS measure is based on the discrete Kullback Leibler divergence $KL()$:

$$JS(\mathbf{a},\mathbf{b}) = \frac{1}{2}(KL(\mathbf{a},\frac{\mathbf{a}+\mathbf{b}}{2}) + KL(\mathbf{b},\frac{\mathbf{a}+\mathbf{b}}{2})) \tag{3.3}$$

where

$$KL(\mathbf{a},\mathbf{b}) = \sum_{i=k}^{K} a_i \log \frac{a_k}{b_k} \tag{3.4}$$

We do not use the Kullback Leibler divergence directly to measure similarity as it is not symmetric. In contrast, the Jensen-Shannon divergence is a metric defined by the average KL distance to the average of the two probability distributions. It is a symmetric and smoothened version of the KL distance.

The fourth measure we examine in this work is adopted from language-based information retrieval. Here, each document is indexed by the likelihood of its model generating the query document, i.e., the most relevant documents are the ones whose models maximize the conditional probability on the query terms. In the visual domain, we represent a query image as a set of visual words $\mathbf{w_a}$ and the above-mentioned likelihood turns into:

$$P(\mathbf{w_a}|M_b) = \prod_{j=1}^{N_a} P(w_j^a|M_b) \tag{3.5}$$

where M_b is the model of an image d_b, N_a the total number of detected visual words in image d_a and w_j^a the value of the word j in image d_a.

In previous work, Wei and Croft [103] linearly combined the LDA model and the unigram (or bag-of-words) model with Dirichlet smoothing to represent an image. Similarly we estimate the terms $P(w_j^a|M_b)$ by combining the respective topic model for image b, M_b^t, with the unigram model of that image M_b^u:

$$P(w_j^a|M_b) = \lambda \cdot P_U(w_j^a|M_b^u) + (1-\lambda) \cdot P_T(w_j^a|M_b^t) \tag{3.6}$$

where $0 \leq \lambda \leq 1$ is a weighting factor.

$P_U(w_j^a|M_b^u)$ in Equ. 3.6 denotes the probability of w_j^a in image d_a specified by the unigram document model with Dirichlet smoothing M_b^u of image d_b. According to [106] it is given by:

$$P_U(w_j^a|M_b^u) = \frac{N_b}{N_b+\mu} P_{ML}(w_j^a|M_b^u) + (1-\frac{N_b}{N_b+\mu}) P_{ML}(w_j^a|D) \tag{3.7}$$

where μ denotes the Dirichlet prior and N_b the number of visual words detected in image d_b. $P_{ML}(w_j^a|M_b^u)$ is the Maximum Likelihood (ML) estimate for the probability of w_j^a under the unigram model M_b^u of image d_b. The ML estimate of the visual word w_j under the unigram model of d_b is simply derived by the relative word count of the value of the visual word j in that

image:

$$P_{ML}(w_j|M_b^u) = \frac{n(w_j,d_b)}{\sum_{j=1}^{N_b} n(w_j,d_b)} \tag{3.8}$$

$n(w_j,d_i)$ is here the number of times the value w_j of the word j occurs in image d_i. This probability is smoothened by using the maximum likelihood probability $P_{ML}(w_j^a|D)$ of w_j^a given the entire collection of images D, i.e., the probability of a visual word w_j given the image collection is given by:

$$P_{ML}(w_j|D) = \frac{\sum_{i=1}^{M} n(w_j,d_i)}{\sum_{i=1}^{M}\sum_{j=1}^{N_i} n(w_j,d_i)} \tag{3.9}$$

where M denotes the number of image in the database and d_i a specific image in D.

The term $P_T(w_j^a|M_b^t)$ in Equation 3.6 refers to the probability of the value w_j^a of the visual word j in image d_a given the topic model M_b^t of image d_b. For the different topic models the respective probability is given by:

pLSA:

$$P_T(w_j^a|M_b^{plsa}) = \sum_{k=1}^{K} P(w_j^a|z_j=k)\cdot P(z_j=k|d_b) \tag{3.10}$$

LDA:

$$P_T(w_j^a|M_b^{lda}) = \sum_{k=1}^{K} P(w_j^a|z_j=k,\beta)\cdot P(z_j=k|\theta^b,\alpha) \tag{3.11}$$

CTM:

$$P_T(w_j^a|M_b^{ctm}) = \sum_{k=1}^{K} P(w_j^a|z_j=k,\beta)\cdot P(z_j=k|\eta^b,\mu,\Sigma) \tag{3.12}$$

3.3. Experimental Evaluation

We experimentally evaluate the proposed system in a query-by-example retrieval task. The objective of example-based image retrieval is to obtain images with content similar to a given sample image, also called the query. We evaluate retrieval results based on the judgments of several ordinary test users about the perceived visual similarity of the retrieved images with respect to the query image.

We compare the performance of different probabilistic topic models in our system and examine the parameter settings. The following three topic models are evaluated: pLSA, LDA and CTM. Furthermore the proposed similarity measures are evaluated as well. As stated above, it has been shown in previous work [59] that using a pLSA model to represent images in an query-by-example task outperforms conventional methods such as using directly a bag-of-words model on local image features or a Color Coherence Vectors (CCVs) [73] as image representation. Thus we will compare different topic models with each other but not to previous approaches.

Category	OR list of tags	# of images
1	wildlife animal animals cat cats	28509
2	dog dogs	24660
3	bird birds	20908
4	flower flowers	25457
5	graffiti	21888
6	sign signs	14333
7	surf surfing	29552
8	night	33142
9	food	18602
10	building buildings	16826
11	goldengate goldengatebridge	23803
12	baseball	12372
	Total # of images (Note images may belong to multiple tag categories)	246,348

Table 3.1: Image database and its categories used for experiments.

3.3.1. Database

All experiments are performed on a database consisting of 246,348 images. The images were selected from all public Flickr images uploaded prior to Sep. 2006 and labeled as *g*eotagged together with one of the following tags: *s*anfrancisco, beach, and *t*okyo. Of these images only images having at least one of the following tags were kept: *w*ildlife, animal, animals, cat, cats, dog, dogs, bird, birds, flower, flowers, graffiti, sign, signs, surf, surfing, night, food, building, buildings, goldengate, goldengatebridge, baseball. The resulting image database was not cleaned nor preprocessed in any way to increase consistency. We can group images into twelve categories as shown in Table 3.1. Example images from all twelve categories are shown in Figure 3.2.

The preselection of a subset of images from the entire Flickr database based on tags is needed as Flickr is a repository with hundreds of millions of images. However, it should be noted that indexing purely based on tags is not sufficient as the tags are a very noisy indication of the content shown in the images. This can be observed in Figure 3.3. Note that this database has also been used for the experimental evaluation in [59].

3.3.2. Local Feature Descriptors

As mentioned in the system description above, various local feature detectors and descriptors can be applied to extract the basic image features that are used to build the discrete vocabulary, which in turn is used to derive the bag-of-words model and thus the basic image description a

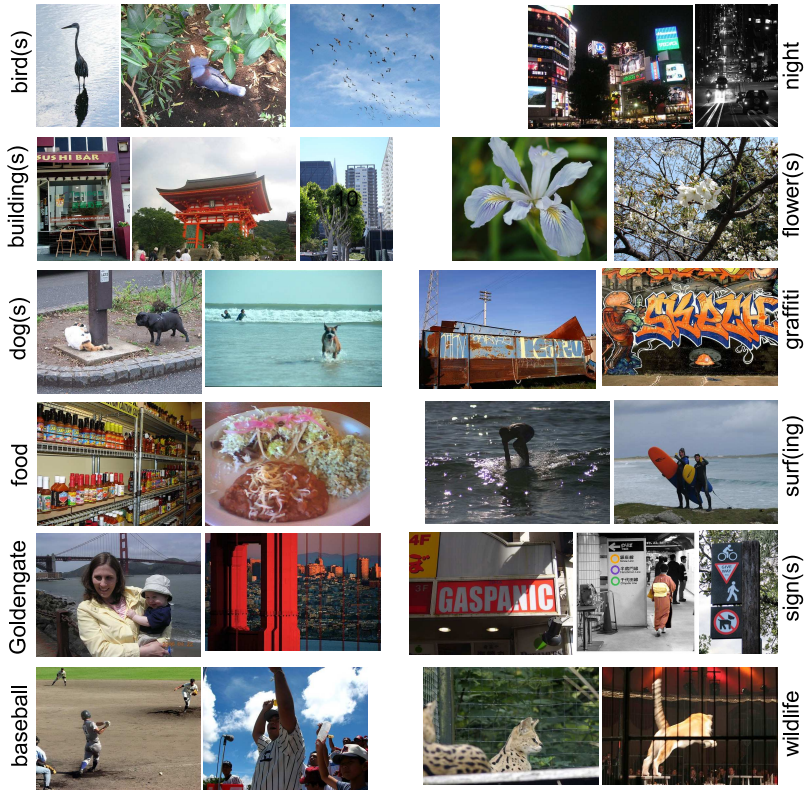


Figure 3.2: Example images from the twelve categories of the Flickr dataset.

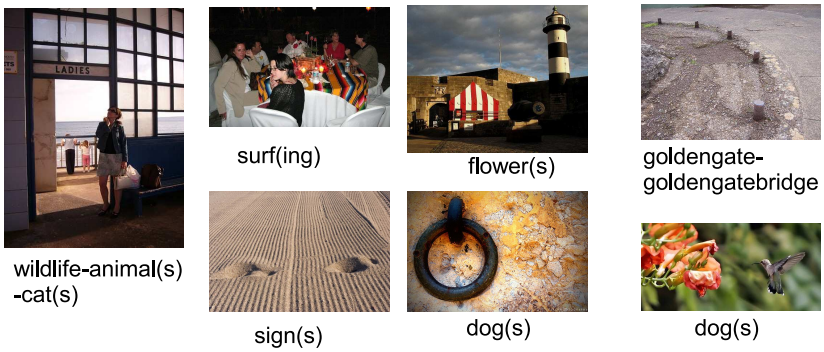


Figure 3.3: Example images showing that categories/tags do not always refer to the content shown.

topic model is computed from. In Chapter 4 we will further evaluate the influence of various feature detectors and descriptors in the context of our retrieval system and describe some of them in more detail. For now we use the well-known SIFT features proposed by Lowe [62] as local image descriptors. They are computed in two steps: A sparse set of interest points is detected at extrema of the difference of Gaussian pyramid, and a scale and orientation are assigned to each interest point besides its position. Then we compute a 128-dimensional gradient-based feature vector from the local gray scale neighborhood of each interest point in a scale and orientation

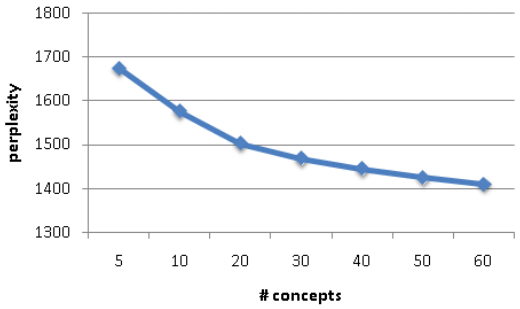


Figure 3.4: Perplexity vs. number of topics K for the pLSA model.

invariant manner.

Note that each image usually leads to a different number of features even if two images have the same size. The number of feature computed depends on the structure and texture of the image.

For our experiments we construct a visual vocabulary out of twelve randomly selected, non-overlap-ping subsets of all visual words of the image database. Each subset contains 500,000 visual features and was clustered to produce 200 distinct visual words. The clusters were then merged, resulting in an overall vocabulary size of 2,400.

3.3.3. Parameter Settings

Since it is not obvious how to choose the parameters in our probabilistic topic models, the first step in evaluating our retrieval system is to determine suitable parameter settings, such as the number of training images as well as the number of topics K. Thus a measure to assess the performance with respect to different parameter settings is needed. The *perplexity* is frequently used to assess the performance of language models and to evaluate LDA models in the context of document modeling [14]. The perplexity indicates how well the model is able to generalize on a held out dataset D_{test}, and is defined by:

$$per(D_{test}) = exp\left\{-\frac{\sum_{i=1}^{M} \log P(\mathbf{w_i})}{\sum_{i=1}^{M} N_i}\right\} \tag{3.13}$$

This measure decreases monotonically in the likelihood of the test data, thus lower values indicate better modeling performance.

In order to evaluate the influence of the choice of the number of hidden topics, we train our models on a subset of 50,000 database images using a different number of topics each time. The perplexity is then calculated on a previously unseen test set consisting of 25,000 images also from the database. The results for the different models are depitced in Figure 3.4 to 3.6.

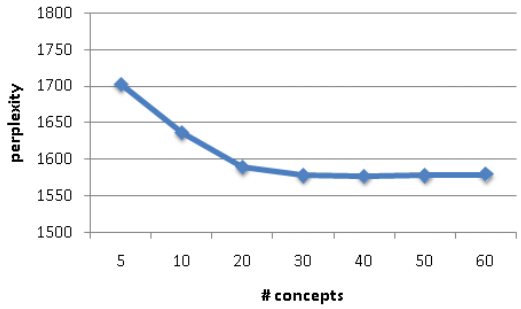


Figure 3.5: Perplexity vs. number of topics K for the LDA model.

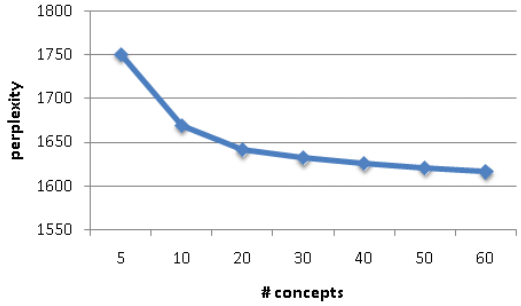


Figure 3.6: Perplexity vs. number of topics K for the CTM model.

One can see that for all models the perplexity decreases with an increasing number of topics. If the number of topics is small, i.e., $K < 30$, the perplexity grows rapidly indicating that the model fails to fit the unseen test data. For $K \geq 30$ the perplexity is almost constant.

Our aim is to obtain a rich image description for our retrieval task, but at the same time we need to consider the dimensionality of our final model as a smaller number of topics is preferred in large-scale databases to represent the images due to memory constraints and computational efficiency. Observing that the difference in perplexity values is rather small above 50 topics, we set $K = 50$ in all our subsequent experiments.

Next we evaluate the influence of the size of the training set on the perplexity. We fix the number of topics $K = 50$ and vary the number of images in the training set in order to evaluate the change in perplexity. Perplexity is again calculated for each setting based on a previously unseen test set consisting of 25,000 images. The resulting perplexities for the different models can be observed in Figure 3.7 to 3.9.

In the pLSA and LDA case, the perplexity decreases with an increasing number of training samples and is approximately constant for training set sizes above $20,000$ images. For the CTM model the perplexity does not follow a clear pattern. This unexpected behavior may be explained by the more complex model and training procedure as well as other parameter settings for training in this model. In general, the dependence of the perplexity on the number of training

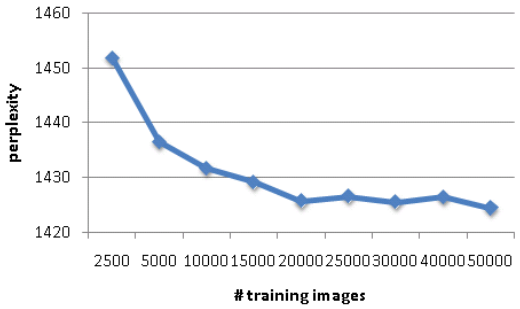


Figure 3.7: Perplexity vs. number of training samples for the pLSA model.

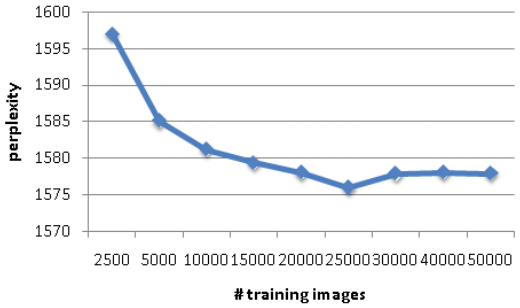


Figure 3.8: Perplexity vs. number of training samples for the LDA model.

samples does not seem to be as pronounced as it is for the number of topics. The appropriate number of images used to train the topic models may also depend on other parameters such as the choice of the maximum number of iteration in the (variational) inference part as well as the number of topics and the size of the vocabulary, respectively. However it is important to notice that in our tests it does not seem to be necessary to learn the parameters of the different topic models on the entire database, which is a huge advantage in large-scale databases. This also enables adding novel images without re-learning the corpus level parameters as long as they show already learned topics.

3.3.4. Different Similarity Measures

We described four different similarity measures for image representations based on topic models in the previous section. Here we evaluate their effects on an image retrieval task, with the number of topics set to 50 and by training each model on 50,000 images. Once we compute the model, we derive the topic mixtures for each image in the database as described in the previous chapter. The parameters μ and λ of the information retrieval based distance measure are set to 50 and 0.2, respectively.

We judge the effect of the similarity measures on the retrieval results in a query-by-example task, i.e., given a query image the goal is to find images of similar content in the database.

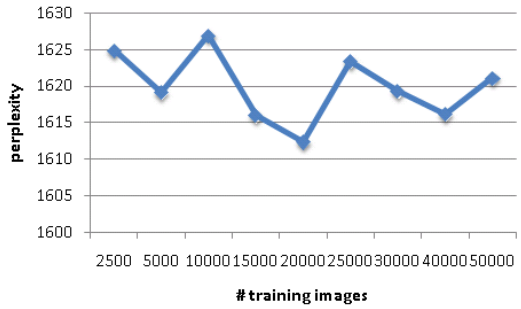


Figure 3.9: Perplexity vs. number of training samples for the CTM model.

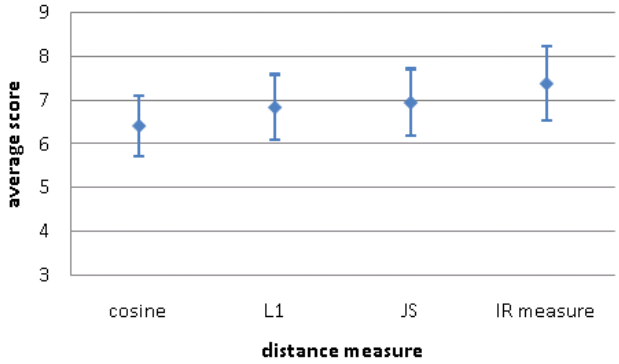


Figure 3.10: Average number of correctly retrieved images using a pLSA-based image representation and different similarity measures.

Thus, we select five query images per category at random resulting in a total of 60 query images for the experiments. These 60 test images are used throughout the experiments in this thesis, and they are depicted in Appendix A.

Having computed for each of the four different measures the $L = 19$ most similar images to each query image, we rate the performance of our models by means of user studies: Users are presented the retrieved images and asked to count the number of correctly retrieved images including the query image, i.e., the minimum count for a query is 1, the maximum 20. The final score for each distance measure is then computed as the average score over all images and users. Note that the judgment of the users is subjective, as each user may perceive the content of an image slightly differently. Thus we also compute the standard deviation from the average score.

The average number of correctly retrieved images for each similarity measure and the three different topic models according to the users' judgment are depicted in Figure 3.10 to 3.12, the vertical bars mark the standard deviation. Eight users have participated in each experiment.

Clearly, the language-based probability measure adopted from information retrieval [103] outperforms all other similarity measures for all three topic models. This indicates that retrieval

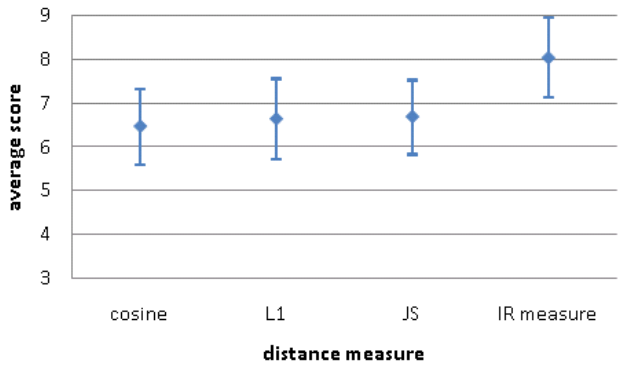


Figure 3.11: Average number of correctly retrieved images using a LDA-based image representation and different similarity measures.

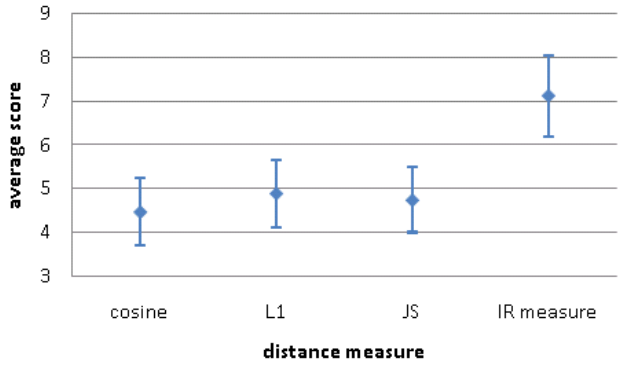


Figure 3.12: Average number of correctly retrieved images using a CTM-based image representation and different similarity measures.

based on the topic distribution is enhanced by also taking word distributions into account. Note that the word probability calculated based on the unigram model is assigned only a small weight of 0.2 whereas the word probability based on the LDA model is assigned a large weight (0.8).

Out of the three similarity measures based on only the topic distributions, the Jensen-Shannon divergence and the L1 distance performed almost equally well. If image retrieval is performed on large-scale databases the probability measure from information retrieval may be too time consuming, and dimensionality reduction in image representation is important. In this case one should also consider one of these approaches, the Jensen-Shannon divergence or the very fast-to-compute L1 norm. As the high-dimensional bag-of-visual-words models are solely needed to learn the topic-model-based representations, only the low-dimensional topic distributions need to be stored and processed for the retrieval task. This allows us to search even very large databases in a reasonable time.

The cosine distance does not seem to be appropriate for large-scale image retrieval in the context of topic models, as it shows the worst performance throughout the experiments.

Further it should be noted that in comparison with LDA and CTM, the resulting scores of the pLSA model are more consistent, they do not show as large differences between the distance measures as we obtain e.g. using an LDA-based image representation.

3.3.5. Different Types of Probabilistic Topic Models

In order to determine the most appropriate topic model for our image retrieval system, the results obtained by using the different topic models are compared to each other. Since the previous subsection shows that retrieval performance depends on the distance measure used, we perform the comparison for two different similarity measures, the best performing IR distance measure as well as the very fast-to-compute L1 measure which is only based on the topic distribution. Using the same setup as before, we perform a user study on 60 test images and present to the users the retrieved images of three models: the pLSA, LDA and CTM. Eight respectively ten users judge the retrieval results for the IR measure and the L1 distance as described above, and the results are depicted in Figure 3.13 and 3.14, showing the means and standard deviations of the number of correctly retrieved images.

It can be seen that when applying the IR distance measure, the LDA model outperforms the other two topic models. In combination with the L1 norm, the pLSA performs best, closely followed by the LDA. As in both experiments the scores for the pLSA and LDA model are close, we perform paired t-tests with $\alpha = 0.01$ to verify the statistical significance of our results. For the IR distance measure the test shows that the hypothesis that the LDA model performs equally well or better than the pLSA model is valid. Similarly we derive that the hypothesis that the scores of the pLSA model are equally good as or better than the ones for the LDA model when using the L1 similarity measure is valid as well. This can also be seen by comparing the scores given by one single person for the LDA and pLSA model directly. Here we see that, while different users derive different mean scores for the models, still in the case of the IR measure all users gave higher mean scores to the LDA model than to the pLSA model. When using the L1 similarity measure, nine out of ten test users assigned higher mean scores to the pLSA model than to the LDA model. Thus in both cases the variance shown in the figures is due to the different interpretations of the relevance of the result images and does not show that the users disagree in the ranking of the models.

It is obvious that the average number of correctly retrieved images of the CTM-based representation is lower compared to the score of the LDA and pLSA-based descriptions. This result is surprising as, when applied to text documents, the CTM has been shown to produce decent results [13]. The inferior performance of the CTM model in our database might be due to the number of topics in the model. As the database is quite noisy, the number of topics might have been chosen to small to allow for dependencies between the topics. However, a large number of topics contradicts the aim of finding a suitable low-dimensional representation that allows fast retrieval in large databases. This issue needs to be investigated and addressed in further

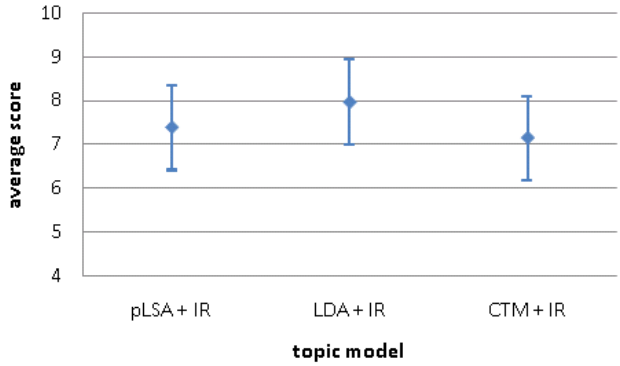


Figure 3.13: Average number of correctly retrieved images of the different topic-model-based image representations when using the IR distance measure.

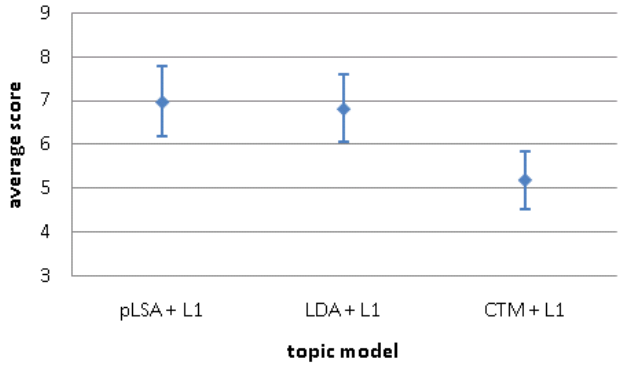


Figure 3.14: Average number of correctly retrieved images of the different topic-model-based image representations when using the L1 distance measure.

research.

Summarizing we can conclude that the pLSA and LDA model perform equally well, whereas in our experiments the CTM model did not seem to be appropriate for such a large-scale image retrieval task.

3.3.6. Results

Finally we show some retrieval results obtained by the proposed system with different topic models in Figure 3.15 to 3.19. As one can see, the system performed very well for the queries shown in Figure 3.15 to 3.17. It should be noted that some queries, such as the one shown in Figure 3.17, perform very well for all topic model types while other queries are more difficult. Figure 3.18 shows a query where the results were suboptimal, and for the query depicted in Figure 3.19 the system failed completely. Displayed results are obtained using the IR similarity measure.

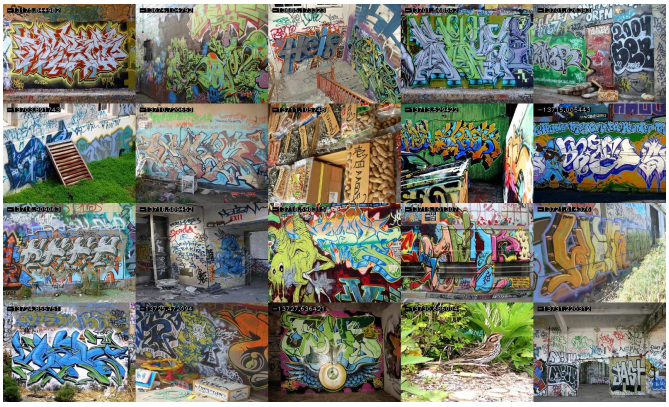


Figure 3.15: Result obtained by our retrieval system using the pLSA model and the IR similarity measure. The top left image shows the query image and the remaining images are the 19 most relevant images retrieved.

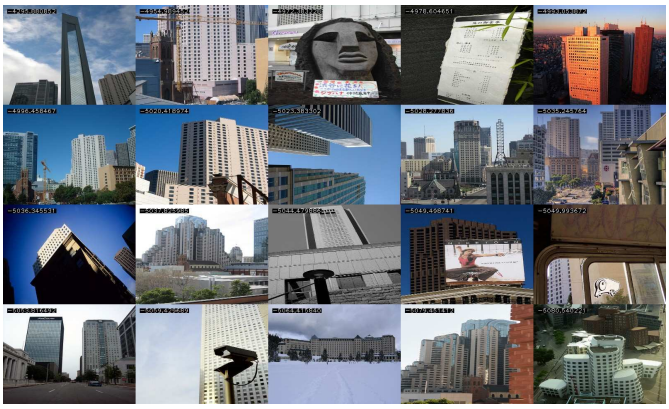


Figure 3.16: Result obtained by our retrieval system using the LDA model and the IR similarity measure. The top left image shows the query image and the remaining images are the 19 most relevant images retrieved.

3.4. SVM-based Active Learning

Interactive retrieval is another image search task that we consider in our work. Here the goal is to deduce the user's intent with respect to his/her desired retrieval results. This is accomplished by taking user-provided feedback through interaction with the system into account, additional to the current query image. This enables the system to refine its search results.

Active learning is an approach to interactive image retrieval which performs several query rounds. After each round the system presents its current search results and asks the user to judge a certain number of appropriately chosen images as relevant or not. This feedback is then used in the next round to improve the classifier that is learned to separate relevant images, i.e., the ones the user is interested in, from irrelevant images.

Tong and Chang [95] proposed active learning with a binary support vector machine (SVM classifier), i.e., a hyper plane in some high dimensional space. The presented active learning

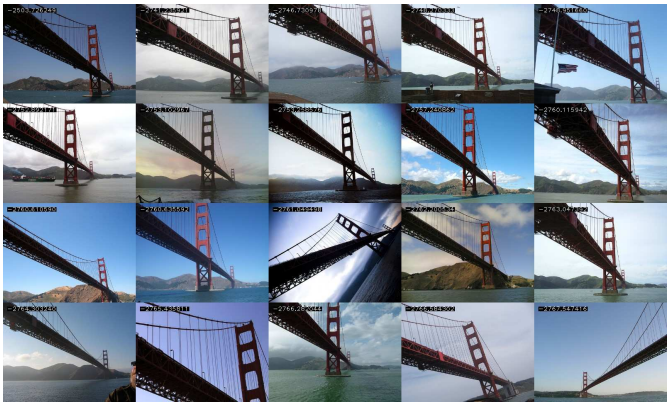


Figure 3.17: Result obtained by our retrieval system using the CTM model and the IR similarity measure. The top left image shows the query image and the remaining images are the 19 most relevant images retrieved.

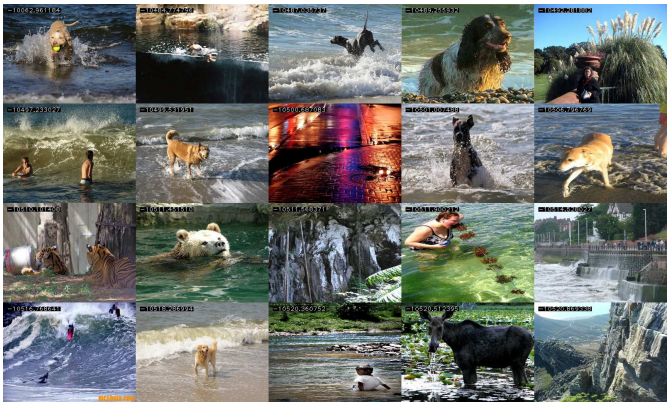


Figure 3.18: Result obtained by our retrieval system using the LDA model and the IR similarity measure. The top left image shows the query image and the remaining images are the 19 most relevant images retrieved.

method works as follows: An SVM classifier is trained in each query round based on all labeled images so far. In the first query round the algorithm is initialized with one relevant and one irrelevant image, and the user labels a randomly selected set of T images. In each following round the T most informative images are presented to the user for labeling. The most informative images are defined according to the so called 'simple method' [95] as the closest images to the current hyper plane. Note that it is important to select appropriate images for labeling in each query round, as the system should converge quickly to the user's desired query concept, i.e., in a minimum number of query rounds. After a number of feedback rounds, the most relevant images are presented to the user as the query result. The binary SVM classifier subdivides the space by the hyper plane in two sets, relevant and irrelevant images, and thus the most relevant images are those that are farthest from the current SVM boundary in the kernel space and at the same time on the correct side of the hyper plane.

In order to apply this algorithm to images, each image needs to be presented as a vector. We

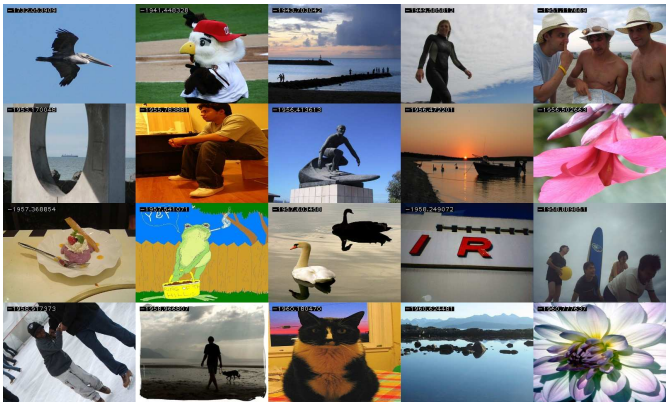


Figure 3.19: Result obtained by our retrieval system using the pLSA model and the IR similarity measure. The top left image shows the query image and the remaining images are the 19 most relevant images retrieved.

propose to represent the images in the database by their topic distributions, thus combining topic-model-based image representation and SVM active learning.

The active learning algorithm works well for small databases with carefully selected images. Problems arise when applying the algorithm to large-scale databases. First, the user needs to find at least one positive query image to initialize the algorithm. Fortunately in this work the query-by-example task is considered and thus the example image can be used to initialize the algorithm. A second problem arises due to the number of images showing the desired content with respect to the total number of images in the database. If this fraction is very small (as it usually is in large-scale databases), active search is aggravated.

In order to solve this problem, we propose a novel preprocessing step before starting the actual active learning algorithm. This preprocessing step aims to reduce the total amount of images in the database while at the same time keeping images that likely contain the desired concept, i.e., the active learning algorithm will not work on the entire database of images but only on a preselected subset of images. As a convenient side effect of preprocessing, computation time of each query round is reduced as the algorithm is running on a smaller dataset making active search faster.

The proposed data selection approach takes advantage of the learned topic-model-based image representation: We choose a subset of R images for active learning based on the prior detected relevance to the query image. Relevance is defined by similarity based on the topic mixture and one of the distance measures discussed in a previous section. This makes sense as the topic mixture models the image content by topic assignment and thus images having completely different topic distributions are unlikely to match the desired user concept.

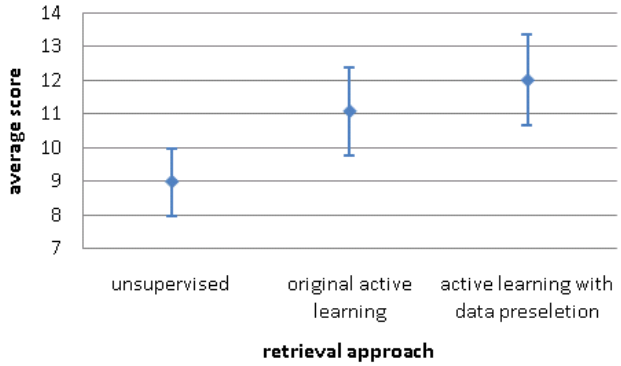


Figure 3.20: Average number of correctly retrieved images of the two active learning approaches and the unsupervised approach.

3.4.1. Experimental Results

In order to evaluate the proposed combination of a topic-based image representation and active learning we perform experiments on the database used in the previous section. We chose the LDA as the topic model in our active learning scenario.

The parameters are set as follows: images are represented by their topic distribution, which is derived from a 50 topics LDA model. We use a radial basis function (RBF) kernel with $\alpha = 0.01$ in the SVM and we set the number of query images T presented to the user in each query round to 20. We choose the parameter R, the size of the preselected subset, to be 20,000. This ensures a sufficient downsizing from the original amount of images in the database while at the same time keeping an adequate number of images likely containing the desired content. The subset of R images is determined by applying the $L1$ distance on the topic distributions.

The results of the active learning algorithm with pre-filtering are compared to the results obtained by the active learning algorithm without pre-filtering [95] and the results from unsupervised retrieval using the IR similarity measure. Evaluation is again performed through user studies as described above. 25 sample query images are chosen from the pool of 60 test images used for the evaluation in the previous section. As a common user will most likely perform no more than three to four query rounds we presented the 19 most relevant images to the given query concept after three rounds of active learning to the test users. The mean over all 25 images is then calculated and the results over all eight test users are depicted in Figure 3.20.

The results show that active learning clearly improves the results compared to completely unsupervised retrieval. Moreover, an additional improvement over the original algorithm [95] can be achieved by using pre-filtering, i.e., data pre-selection.

In Figure 3.21 to 3.23 some sample results showing the effectiveness of the presented active learning approach are depicted. Three pairs of 10 images are displayed, each pair showing the query image and the nine most relevant images found using the unsupervised algorithm evalu-

Figure 3.21: Example results obtained by the unsupervised algorithm (top) and after active learning with pre-filtering (bottom).

Figure 3.22: Example results obtained by the unsupervised algorithm (top) and after active learning with pre-filtering (bottom).

ated in the previous section (top) and after three rounds of active learning with data pre-selection (bottom). Green dots mark images showing the correct content, red dots mark incorrectly retrieved images. Clearly an improvement of the results by active learning can be noticed.

3.5. Summary

In this chapter we have presented our query-by-example image retrieval system. The system's core component is a topic-model-based image representation which solely relies on visual features. We described the necessary steps to adapt topic models to image collections and proposed four different similarity measures appropriate for topic-based image description.

In our experimental evaluation we compared three different topic models, the pLSA, the LDA and the CTM in a retrieval scenario. It was shown that the pLSA and LDA model perform equally well whereas the CTM does not seem to be an appropriate model for query-by-example image retrieval. Experimental comparison of the similarity measures showed that a probabilistic measure combining a topic model and a unigram representation outperformed the other mea-

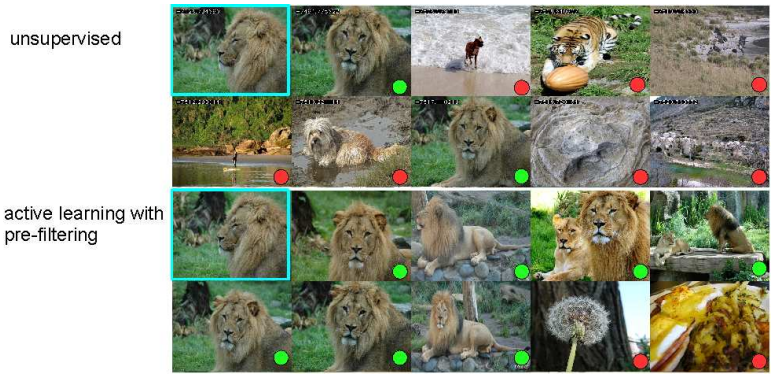


Figure 3.23: Example results obtained by the unsupervised algorithm (top) and after active learning with pre-filtering (bottom).

sures.

Furthermore we have demonstrated how our system can be modified to allow interactive image retrieval. We have applied an active learning algorithm to our topic-model-based image description and we demonstrated that retrieval results were further improved by means of a novel preprocessing scheme that prunes the set of candidate images used during active learning.

Most of the experimental evaluations in this chapter have been performed through user studies. These were necessary as no ground truth is available for our large-scale image database. It should be noted that all our user studies reported in this thesis were based on a small number of test users, typically only five to twelve users were available. However, we performed paired t-tests that showed the significance of our results. Nevertheless it would be desirable to validate the results with a larger group of test users in future experiments.

4. Visual Features and their Fusion

Our retrieval system presented in the last section uses a topic-model-based image description as its core component. To learn and use such a topic model in the context of images we first need to represent each image as a bag-of-visual-words. This is done as described by extracting local features from the images and quantizing them into a finite set of visual words.

In this chapter we will focus on the local image features used as the basic building block in our model. In the last chapter we have first identified interest points as extrema in the difference of Gaussian pyramid and then extracted SIFT features at those interest points. However many other local feature detectors and descriptors can be used. Thus, in the first part of this chapter we will evaluate the influence of the type of detector and descriptor on the performance of our retrieval system as well as in a closely related scene recognition task.

As just mentioned there are various types of descriptors that may be used to build the bag of word model. In some cases it may improve performance to use more than one feature type, i.e., to combine two or more descriptors in order to take advantage of complementary image descriptions. Therefore we propose and examine different models for fusion of more than one local feature type in the second part of this chapter.

4.1. Feature Comparison

When applying topic models in the image domain, the first step is to find an appropriate visual equivalent for words in documents. Thus the retrieval system we described in the previous chapter starts building the topic-based image representation by describing each image with a number of local image descriptor vectors of one kind. By quantizing those features computed for each image we define our visual vocabulary. Subsequently a bag-of-words representation for each image can be easily derived by quantizing the extracted features for each image.

Local image features are often used in this context as they have the advantage of being more flexible than global image characterizations, while at the same time capturing more meaningful patterns than individual pixel values. Local descriptors have become very popular in many computer vision and pattern recognition tasks, and a wide variety of types of local descriptors has been proposed [62, 8, 9, 85], each capturing a different property of a local image region and being more or less invariant to illumination, changes in viewpoint and other image transformations. Given a predefined interest point at a specified scale (i.e., size of local neighborhood),

they describe the local image region surrounding the interest point compactly by a feature vector. In the following we will use the term *f*eature and *d*escriptor interchangeably.

A thorough comparison of local descriptors in the context of matching and recognizing the same object or scene is presented in [66]. However, in a matching task, the aim is to find precisely corresponding points of an object or scene in two images under different viewing conditions such as lightning or pose changes. This requires a very distinct region description. In contrast, in a topic-model-based scene classification or image retrieval task we would like to pool features describing visually similar regions in order to produce meaningful visual words. Most previous works using topic models in the image domain apply and compare the popular SIFT [62] descriptor or simple color/gray scale patches [15, 54, 76]. Bosch et al.'s work [15] proposes a variation of SIFT, taking color channels into account, in the context of scene recognition with a pLSA-based image representation.

In this section we compare two recently proposed local features descriptors, the geometric blur descriptor [9] and the self-similarity descriptor [85] in a scene classification task and a query-by-example retrieval scenario using a pLSA-based image representation. Both features have shown promising performance in various image analysis tasks. They have not been considered in the previous comparison [66]. As the SIFT descriptors have shown to outperform other features in [66], we take results obtained with the SIFT descriptor as a baseline. Moreover we also evaluate three different local interest region detectors with respect to their suitability for these tasks.

4.1.1. Local Region Detectors

We compute local features as described above at predefined interest points with an associated scale factor defining the size of the supporting image region around the interest point. Such interest points and their associated regions can be detected using various approaches. Thus, besides comparing the performance of different local descriptors, we will also analyze the behavior of the features for three different interest point detectors. In the following we describe the considered region detectors.

Difference of Gaussian (DoG) detector

The DoG detector [62] is a scale-invariant region detector which first detects a set of interest points. Then it filters this set to preserve only those points that are stable under a certain amount of additive noise.

As a first step, potential interest points, also called keypoints, in an image are identified by scanning the image over location and scale. The localization and the scale of the keypoints are

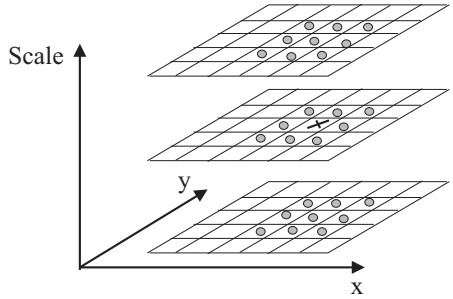


Figure 4.1: Detection of extrema in scale-space by comparing a pixel (x) to its neighbors (○) in the current and adjacent scales (based on [62])

detected as scale-space [1] extrema of the function $D(x,y,\sigma)$, which is the difference-of-Gaussian function convolved with the input image $I(x,y)$:

$$D(x,y,\sigma) = (G(x,y,k\sigma) - G(x,y,\sigma)) * I(x,y) \tag{4.1}$$

where k indicates a constant multiplicative factor and

$$G(x,y,\sigma_i) = \frac{1}{2\pi\sigma_i^2} e^{-(x^2+y^2)/2\sigma_i^2} \tag{4.2}$$

is a Gaussian kernel. Local 3D extrema of $D(x,y,\sigma)$ are detected by comparing each pixel to its eight neighbors in the current scale space level and the nine neighbors in the scale above and below (see Figure 4.1). A point is selected only if it is larger or smaller than any of these neighbors.

Then to accurately determine location and scale, a detailed model is fitted to each candidate location. The function value $D(\hat{\mathbf{x}})$ at the extremum $\hat{\mathbf{x}}$, is used for rejecting unstable extrema with low contrast. All extrema with a value of $|D(\hat{\mathbf{x}})|$ less than a certain threshold (determined empirically through experiments) are discarded. However, to ensure stability it is not sufficient to reject interest point candidates with low contrast, but also points with unstable localization along edges must be eliminated. That is done by discarding interest points that have a ratio of principal curvatures greater than a certain threshold, as interest points on edges will have a large principal curvature across the edge but a small one perpendicular to it [62].

Summarizing, interest points are defined as scale space extrema in the DoG pyramid and are associated with their respective scale. Thus the DoG detector facilitates scale-invariant computation of the subsequent local feature descriptor if the supporting region size takes the scale factor into account.

[1] Scale-space $L(x,y,\sigma)$ is a local 3D representation of an image where σ indicates the scale. Different levels of the scale-space representation are created by convolving the input image $I(x,y)$ with a variable-scale Gaussian kernel $G(x,y,\sigma)$: $L(x,y,\sigma) = G(x,y,\sigma) * I(x,y)$

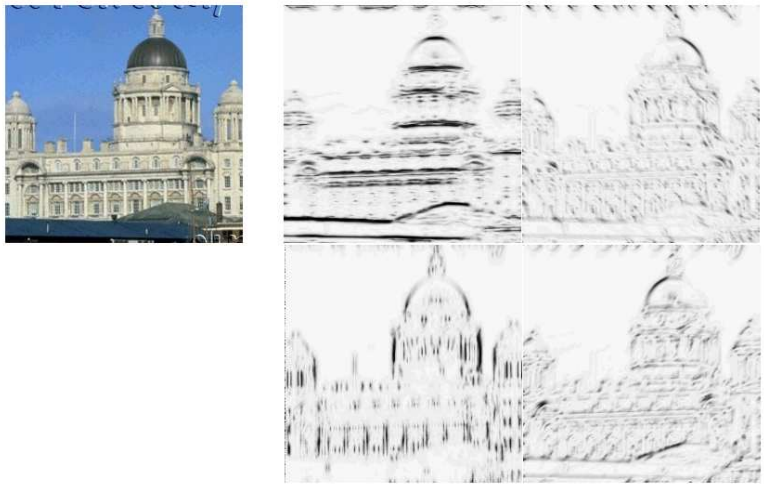


Figure 4.2: Example image (a) and its four computed edge channels (b).

Note that in this approach the number of interest points detected in an image varies as it depends on the structure and texture in each image.

Dense grid over several scales

The second detector we consider is a dense grid over several scales. Therefore we create an image pyramid with a predefined scale factor. Then we apply dense sampling with a vertical and horizontal step size of d pixels across the images in the pyramid. The supporting region size of the subsequent descriptor computation is held fixed for all pyramid levels, and thus the scale of an interest point is defined by the current pyramid level.

Note that this approach does not ensure scale invariance of the feature descriptors computed subsequently. However by computing features at different scales, i.e., at different pyramid levels, this dense sampling at least ensures a (very) limited degree of scale invariance in the representation. One advantage of this approach is that if images have the same size, the same number of interest points is computed for each image. Moreover all regions of the image are considered in the feature vectors, whereas in case of the DoG detector mostly distinct regions which are textured/structured are represented.

Edge sampling

Extracting interest points by edge sampling [9] requires them to be located at positions of high edge energy. In a first detection step we compute a number of oriented edge channels, in our case four, by using a boundary detector [63]. Four edge channels of an example image are depicted in Figure 4.2. Then all edge channels are thresholded keeping only locations of high

edge energy. Interest points are computed by randomly sampling those locations. We randomly sample all edge channels, nevertheless every position is selected at most once.

Note that in this approach we predefine the number of features per image. Features are computed at one scale and they only represent image regions close to positions with high edge energy in at least one of the edge channels.

4.1.2. Local Feature Descriptors

We investigate the performance of the following three local feature descriptors in the context of topic models:

Scale Invariant Feature Transform (SIFT)

The Scale Invariant Feature Transform (SIFT) [62] feature computation for a detected interest point starts by assigning an orientation, scale, and location to the interest point. The subsequent descriptor computation is then performed on image data that has been transformed relative to the assigned orientation, scale and location, thereby providing invariance to these transformations.

The scale and location of the interest point are determined by the feature detector used (see above) and define the size and position of the local neighborhood around the detected interest point the descriptor is based upon.

One or more orientations are assigned to an interest point based on the dominant gradient orientations of the local image patch surrounding the interest point. Dominant gradient directions are identified by selecting peaks within an orientation histogram. This histogram is formed from the gradients' angles of sample points within a region around the keypoint, weighted by each gradients' magnitudes. For each dominant orientation an interest point is created with that orientation, i.e., multiple interest points might be created for the same location and scale, but with different orientations.

Having determined the size and location of the interest point neighborhood, we construct a representation of the local image patch around the interest point based upon the local image gradients. The feature entries are thereby computed relative to the interest point's assigned orientation, i.e., the local image region is transformed such that the associated orientation always points in the same direction.

Feature computation starts by dividing the local neighborhood into subregions. Subsequently the gradient magnitudes of each image sample point in a patch around the interest point location are accumulated into a local orientation histogram. While aggregating the gradients into the histograms, they are weighted with a Gaussian window centered at the interest point location.

The computation of a 2×2 descriptor array computed from an 8×8 set of samples is illustrated

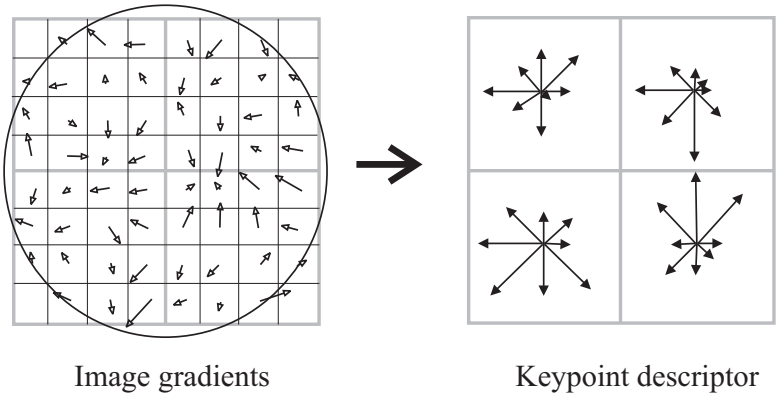


Figure 4.3: Creation of an SIFT descriptor from the image gradients in the neighborhood (based on [62])

in Figure 4.3. On the left side the gradient's magnitudes and orientations are indicated by the lengths and directions of the arrows respectively; the overlaid circle indicates the Gaussian window.

In most applications and also in this work, 4×4 arrays of histograms with 8 orientation bins in each histogram are computed from 16×16 sample arrays. This results in 128-dimensional feature vectors where the values of the local orientation histogram bins (= length of arrows) form the entries of the feature vector.

Finally, the vector is normalized to ensure invariance to illumination conditions. SIFT features are also invariant to small geometric distortions and translations due to location quantization. They are widely used in several computer vision and pattern recognition tasks. Thus the results obtained with SIFT features serve as a baseline in this work.

Geometric blur

The geometric blur feature vector computation [9] starts by computing a number of oriented edge channels for the currently considered image. In our work we computed the edge channel images by the boundary edge detector proposed by Martin et al. [63]. These edge channels provide the required sparse signal for computing the geometric blur descriptor. An example image together with its four edge channels is shown in Figure 4.2.

Having derived the edge channels for an image, we determine a sub-descriptor for each edge channel and each interest point. The concatenation of all sub-descriptors associated with one interest point forms its geometric blur descriptor.

In order to build a sub-descriptor we collect the values of sample points in the neighborhood of the interest point. Sample points lie on concentric circles around the interest point as shown in Figure 4.4. In our implementation the outermost circle has a radius of 20 pixels. The distance between the six concentric circles decreases in a quadratic manner towards the center. As twelve

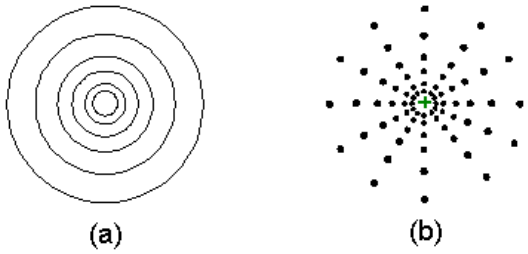


Figure 4.4: Concentric circles around the interest point where the distance between the circles decreases in a quadratic manner (b) and twelve equally distributed samples are taken from each concentric circle around the interest point to form the descriptor (a).

equally distributed sample values are taken from each circle the size of each sub-descriptor is 72 and thus the dimensionality of the entire feature vector is 288 when using four oriented edge channels.

The value of each sample point is taken from a blurred version of the respective edge channel image. Blurring is performed by convolving with a Gaussian kernel whose standard deviation is defined by the distance of the sample point from the interest point. A blurred version $\mathbf{E_d}$ of the edge channel $\mathbf{E}$ is thus derived by:

$$\mathbf{E_d} = \mathbf{E} * \mathbf{G_d} \tag{4.3}$$

where $\mathbf{G_d}$ is a Gaussian kernel with standard deviation d. Thus to compute the geometric blur descriptor, assuming we take a sparse set of sampling points s_i, we first need to compute the blurred versions $\mathbf{E_d}$ of the edge channels for the values d:

$$d = \alpha |s_i| + \beta \tag{4.4}$$

where α and β are constants that determine the amount of blur, s_i is given relative to the interest point and thus $|s_i|$ denotes the distance of the respective sampling point to the interest point. Then each sample point value is drawn from the appropriate blurred version of the edge channel image and they together form the final descriptor.

By taking the sample point value from versions of the edge channel image that are smoothed proportional to the distance of the sample point to the interest point, we derive a partly affine invariant region descriptor. This is due to the assumption that under an affine transformation of the region around the interest point, a piece of signal further away from the interest point moves more than a closer piece.

Note that features could also be computed in an orientation invariant manner by determining the edge channel with the strongest response and computing the feature relative to this edge channel. This option has not been explored in our implementation.

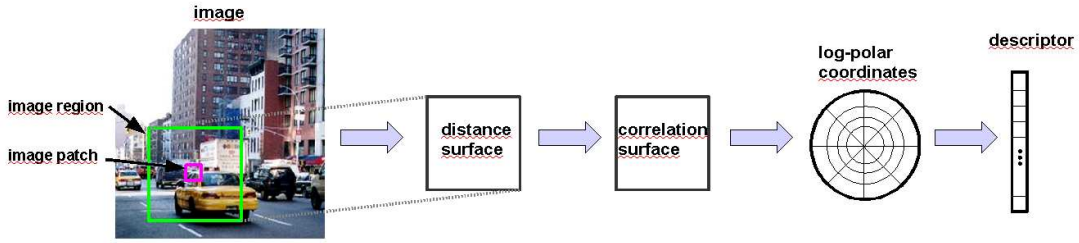


Figure 4.5: Self-similarity feature extraction.

Self similarity

An overview if the self-similarity feature extraction is shown in Figure 4.5.

The extraction of a self-similarity feature [85] for an interest point starts by computing a so called correlation surface for the neighborhood around the interest point q. The size of the neighborhood is determined by the scale associated with the interest point and the resulting local image region is then resized to a patch R_q of size $p_2 \times p_2$ which is centered at q (denoted by the large green rectangle in Figure 4.5).

In order to derive a correlation surface, we compare a small image patch i_q of size $p_1 \times p_1$ around the interest point q (denoted by the small red rectangle in Figure 4.5) with sub-patches i_q^l of the same size out of the resized local image region R_q. Comparing the patches based on the sum of squared differences between the pixels' gray values, we derive a discrete distance surface $SSD_q(x,y)$ as a function of the position of the center (x,y) of i_q^l. In this work we choose $p_1 = 5$ and $p_2 = 41$.

This distance surface $SSD_q(x,y)$ is then normalized and transformed into a correlation surface $S_q(x,y)$ according to:

$$S_q(x,y) = \exp\left(-\frac{SSD_q(x,y)}{\max(var_{noise}, var_{auto}(q))}\right) \tag{4.5}$$

$var_{auto}(q)$ is a variable that takes into account the variations of gray values in the patch, i.e., its contrast and its pattern structure, and is used for normalization. We choose $var_{auto}(q)$ as the largest value in the distance surface directly around the interest point, i.e., in the 3×3 region around q. var_{noise} denotes a constant corresponding to acceptable photometric variations. var_{noise} is used in cases where $var_{auto}(q)$ is small, i.e., the pixel values in the region around the interest point are very similar, in order not to increase the pixel noise.

Then this correlation surface is transformed into a log-polar coordinate system and partitioned into 80 bins (20 angles, 4 radial intervals). The maximum values in each bin constitute the local self-similarity descriptor which thus has 80 dimensions.

Finally the derived descriptor vector is normalized which ensures some invariance to color

and illumination changes. Invariance against small local affine and non-rigid deformations is achieved by the log-polar representation. By choosing the maximal correlation value in each bin, the descriptor becomes insensitive to small translations.

All investigated feature descriptors are purely based on gray-scale images. In some tasks such as scene classification, the performance is likely to improve by taking color into account (e.g., color SIFT [15]). As this may not be true for other content analysis tasks using probabilistic topic models such as object recognition or image retrieval (because here categories might be defined by shape rather than color), we do not consider color in this section. Color features are investigated in more detail in the second part of this chapter.

4.1.3. Experimental Evaluation

For evaluation purposes we use the pLSA model in our system to represent each image. We perform the evaluation in two different tasks, scene recognition and image retrieval. The reason for choosing scene recognition for a first evaluation and subsequent comparison to the results in a retrieval task is due to the availability of annotated scene recognition databases. This enables us to evaluate the results automatically whereas for image retrieval we need to perform user studies. Those user studies are time and labor consuming and therefore only a limited number of experiments is possible. Thus we extensively examine the features in a scene recognition task and then conduct a user study which evaluates different detector/descriptor combinations in an image retrieval scenario. The results of both tasks are subsequently compared.

Scene Recognition

To perform scene recognition we need to modify our image retrieval system (Figure 3.1) only slightly. Given a test image we search for the N most similar images according to our pLSA-based image description, i.e., their topic distributions, in the training set which consists of labeled images. As the similarity measure we use the L2 norm. Then we apply a k-Nearest Neighbor classifier (kNN). Here the test image is classified by the majority vote of the labels of its N neighbors. Note that we could apply more sophisticated distance metrics and/or machine learning algorithms such as SVMs to improve the classification results. As our main goal in this section is to compare different local feature descriptors in the context of topic-model-based image representations and not machine learning algorithms, we have chosen the simple kNN approach. Note that this scene recognition system is similar to the approach proposed by Bosch et. al. [15], which uses a pLSA model to represent images for scene recognition.

Experimental Setup:
We use the OT dataset [71] to evaluate the three different interest region detectors and descriptors in the context of scene classification. The database consists of a total of 2688 images from

category	scene type	number of images
1	coast	360
2	forest	328
3	highway	260
4	inside city	308
5	mountain	374
6	open country	410
7	street	292
8	tall building	356
total		2688

Table 4.1: Categories and number of images per category in the OT dataset.

Figure 4.6: Sample images for each category of the OT dataset.

eight different scene categories. The number of images as well as examples for each category are shown in Table 4.1 and Fig. 4.6, respectively. On this dataset we perform image classification by assigning each test image automatically to one of the eight categories.

We divide the images randomly into 1344 training and 1344 test images. We further subdivide the 1344 training images into a training and a validation set, of size 1238 and 106 respectively. We used the validation set to find the best parameter configuration for the pLSA model. In the model we fix the number of topics to 25 and optimize only the number of distinct visual words for the different detectors/descriptors. A number of 25 topics has been shown to give a good performance for this dataset [15]. We compute the visual words here by applying k-means clustering to a subset of features from the training images. As the number of visual words needed is relatively small, we are in this case able to apply the k-means algorithm directly without merging results of various subsets.

Having determined the optimal number of visual words for the current detector/descriptor combination we re-train the pLSA model with the all the training images by merging training and validation set. Final results are then computed on the test set, and detector/descriptor performances are compared.

In our experiments, we first analyze the suitability of three feature detectors in the scene clas-

sification task while holding the feature descriptor fixed. Then we pick the best performing detector to evaluate the local descriptors.

Interest Point Detectors:
We select the frequently used SIFT descriptor for the comparison of the three detectors. Their parameters are set as follows: the spacing d between grid points is set to 5 pixels, resulting in about 5250 features per image when using a factor of $2^{\frac{1}{4}}$ between different scales. Note that the images in the OT database all have the same size. The number of randomly sampled edge locations per image is set to 5000. On the average a number of 559 features is extracted per image with the DoG detector.

Figure 4.7 displays the resulting recognition rates on the validation set for different numbers of visual words W and all three detectors over the parameter k of the kNN algorithm. We observe that for the DoG detector, the dense grid detector over several scales, and the edge sampling detector $W = 1000$, $W = 1000$ and $W = 1500$ gives the best recognition results, respectively.

Using these parameter settings we train a pLSA model on the entire training set for each detector type and fit the test set images to this model in order to compute a topic vector representation for all images. The comparison of the recognition results on the test set can be seen in Fig. 4.8. The dense grid detector outperforms the other detectors followed by random edge sampling.

This may be due to several reasons: Firstly, both the dense grid detector and the random edge sampling algorithm compute more features per image than the DoG detector. Moreover they compute an equal number of features for each image. This may enable a better fitting of the pLSA model to the scene recognition problem. Secondly, the interest points and regions computed by the dense grid cover the entire image and thus the bag-of-words image representation also covers the entire image and not only regions close to edge pixels or scale-space extrema. A further reason might be that in a scene recognition task the repeatability of exact positions and scales, as provided by the DoG detector, may be not as important as in other tasks such as object recognition where one would like to match only the exact subpart. In contrast to the other detectors, the DoG detector offers scale invariance. Nevertheless this is also not as important in scene classification as, e.g., in object detection. In the OT database used for evaluation, all images of one category are taken at approximately the same scale. Note that the results are consistent with previous results [15], where a dense representation performed best, too.

Feature Descriptors:
The dense grid detector showed the best recognition performance in the evaluation above, thus we use this interest point detector for the comparison of local feature descriptors. First we determine the appropriate number of visual words in the pLSA model for each descriptor. This has already been done for the SIFT feature (see Figure 4.7). Figure 4.9 depicts the recognition rates for different k in the kNN and different numbers of visual words, for the geometric blur

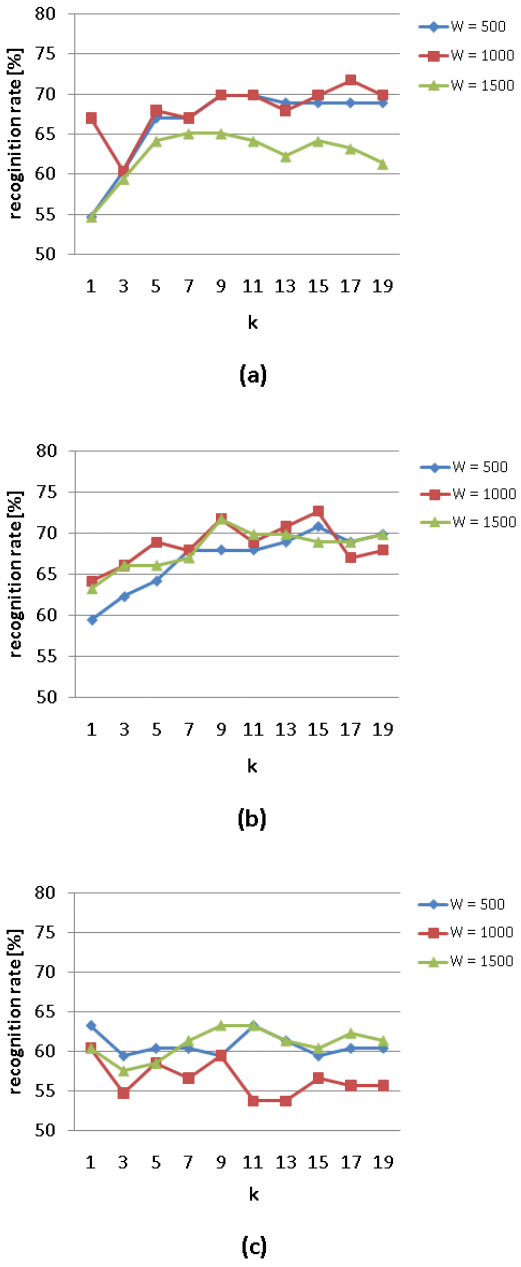


Figure 4.7: Recognition rates on the validation set for the three different detectors over parameter *k* of kNN for different numbers of visual words: (a) difference of Gaussian, (b) dense grid and (c) edge sampling.

descriptor and the self-similarity descriptor. The best results for both features are obtained using 1500 visual words.

For both descriptors we train a pLSA model on the entire training set, compute a topic vector representation for all training and test images and perform a kNN classification of the test images based on the topic distributions. Then we compare the results of all local features,

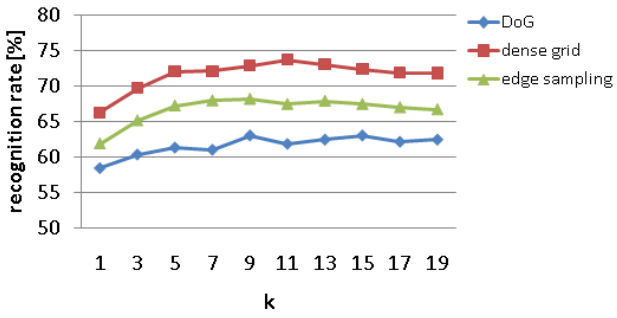


Figure 4.8: Recognition rates on the test set for three different detectors over k for kNN.

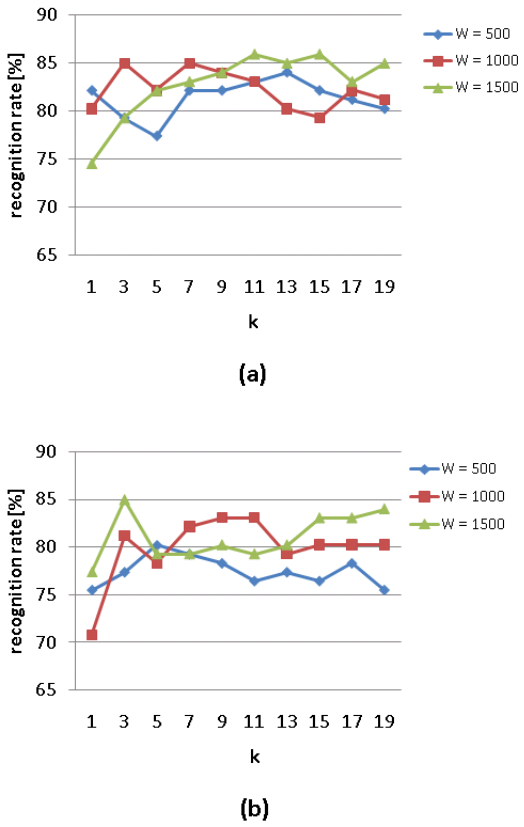


Figure 4.9: Recognition rates on the validation set for the geometric blur (a) and theself similarity (b) feature for different numbers of visual words and k in kNN.

including SIFT, in Figure 4.10.

It can be seen that both, geometric blur and self-similarity features outperform the commonly used SIFT feature by about 4% for the OT database. Moreover the geometric blur feature has a slightly better recognition rate, about 1% better, than the self-similarity feature, and the best recognition is achieved for $k = 11$ with 78.05%.

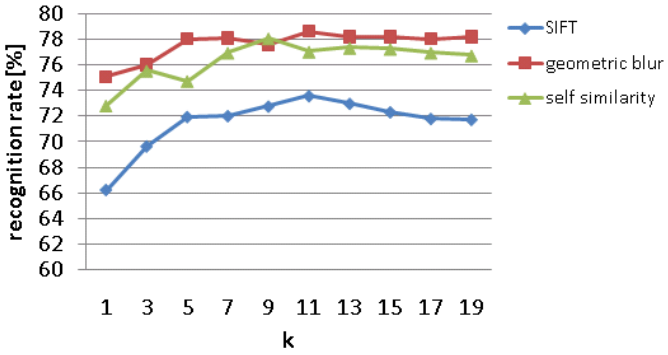


Figure 4.10: Recognition rates on the test set over k for kNN for different local feature types.

However, a performance difference of 1% does not seem to be significant given the small size of the OT dataset. It also has to be taken into account that the self-similarity descriptor is of lower dimensionality than the geometric blur features: 80 vs. 288 dimensions. This lower dimensionality makes computations such as clustering and visual word assignment much faster. Moreover, the self-similarity feature is computed without performing segmentation or edge detection as has to be done to compute the oriented edge channels for the geometric blur feature. We have experimented with simpler oriented-edge channel computations than the one presented by [63], however performance dropped drastically indicating that sophisticated edge channel computations are important. Thus, given the similar performance and the more than a magnitude lower computational complexity over geometric blur, the self-similarity feature is the preferred feature.

For a more detailed analysis of the results, the confusion tables for the best performing parameter settings for each descriptor are depicted in Fig. 4.11. Here it can be seen that there are some categories, such as *f*orest, *i*nside city and *s*treet, where all descriptors work almost equally well, showing a performance of over 80% and in the *f*orest category even achieving over 90% accuracy. We also noticed some confusion between closely related categories with similar visual appearance, e.g. *o*pen country and *c*oast, *t*all building and *i*nside city as well as *m*ountain and *o*pen country. In these cases, results might be further improved by including color.

The largest differences can be noticed in the category *t*all building where SIFT has an about 20% smaller recognition rate than both other features. The geometric blur descriptor significantly outperforms SIFT and self similarity in the category *c*oast, whereas the self-similarity feature performs best in the *o*pen country category.

Finally we would like to examine the variance in performance due to random initialization in both the k-means clustering algorithm and the pLSA implementation. Therefore we choose the parameter and feature setting of the best performing configuration so far (geometric blur descriptor, $W = 1500$, $k = 11$) and repeat the scene classification experiment on the test set ten times, each time computing the visual vocabulary and pLSA model with different random

	1	2	3	4	5	6	7	8
1	68,89	1,11	15,00	0,00	5,00	8,33	0,00	1,67
2	0,61	92,07	0,61	0,00	4,88	1,22	0,61	0,00
3	7,69	0,00	73,08	1,54	3,08	1,54	6,92	6,15
4	0,00	0,00	0,65	85,71	0,00	0,00	5,84	7,79
5	3,74	4,28	3,21	0,00	78,61	7,49	2,14	0,53
6	10,73	10,24	6,34	0,00	15,61	51,71	4,88	0,49
7	0,00	0,00	0,68	4,79	4,79	0,00	86,99	2,74
8	2,81	0,56	2,25	14,61	5,06	1,12	13,48	60,11

(a)

	1	2	3	4	5	6	7	8
1	78,33	0,56	7,78	0,00	0,56	12,78	0,00	0,00
2	0,00	93,29	0,00	0,00	1,83	1,83	3,05	0,00
3	16,15	0,00	70,77	2,31	1,54	4,62	4,62	0,00
4	4,55	0,00	0,00	81,82	0,00	1,30	3,90	8,44
5	0,53	4,81	4,28	0,00	79,14	7,49	3,74	0,00
6	21,46	4,39	5,85	0,00	6,83	59,02	2,44	0,00
7	0,00	0,00	2,05	7,53	1,37	0,00	85,62	3,42
8	0,00	1,69	1,12	8,43	0,56	0,00	3,93	84,27

(b)

	1	2	3	4	5	6	7	8
1	66,11	1,11	9,44	0,56	2,78	17,78	2,22	0,00
2	0,00	90,24	0,00	0,00	2,44	3,66	3,66	0,00
3	9,23	0,00	73,08	0,77	6,92	3,85	6,15	0,00
4	1,30	0,00	0,65	82,47	0,00	0,00	5,84	9,74
5	4,28	6,42	3,74	0,53	70,05	6,95	8,02	0,00
6	9,27	5,37	3,90	0,49	4,88	74,63	1,46	0,00
7	0,00	2,05	1,37	3,42	1,37	0,68	91,10	0,00
8	0,00	0,56	0,00	10,11	0,00	0,00	8,99	80,34

(c)

Figure 4.11: Confusion tables for results on the test set for different descriptor types and a dense grid region detector: (a) SIFT (W=1000, k=11), (a) geometric blur (W=1500, k=11), (c) self-similarity (W=1500, k=9). The numbers 1,2,...8 refer to the categories listed in Table 4.1.

initializations. The recognition rates are between 77.75% and 79.69% with an average value of 78.93% and a standard deviation of 0.58%. It can be seen that there aren't any large variations between different runs of the same experiment.

In summary it can be stated that for scene classification the geometric blur feature outperforms the other descriptors. In cases where fast computation is needed one should nevertheless consider using the lower dimensional and faster-to-compute self-similarity feature which shows a comparable performance to the geometric blur feature.

Image Retrieval

Next we compare our feature detectors and descriptors in an image retrieval task. We evaluate three different detector/descriptor combinations: the DoG detector in combination with the SIFT descriptor, the dense grid detector in combination with the SIFT descriptor and the dense grid detector in combination with the self-similarity feature. Due to the expensive computation of the geometric blur descriptor and its only slight improvement over the self-similarity descriptor we chose not to evaluate this feature in the context of large-scale retrieval. Additionally it is

of very high dimensionality (288) compared to the 128-dimensional SIFT or the 80-dimensional self-similarity descriptor. This is a disadvantage in a model with a large number of visual words as quantization of these descriptors to derive visual words is slow.

Experimental Setup:
We compute a pLSA model consisting of 50 topics and 2400 visual words for each detector/descriptor combination. It should be noted that validating the number of words is not possible in our image retrieval task as we use a large-scale real-world image database for our evaluation without any available ground truth. As the number of visual words in our model is relatively large, we compute the visual vocabulary by merging results from multiple k-means as described in Section 3.1: twelve small subsets consisting of randomly chosen 500,000 features are each clustered separately into 200 clusters using the k-means algorithm. The cluster centers of all subsets are then merged to produce the visual vocabulary. Finally each image is represented by its pLSA topic distribution and the performances of the detector/descriptor combinations are compared.

For our evaluation we use a database consisting of roughly 246,000 images in a retrieval-by-example task. The database has been described in detail in Section 3.3.1. We measure the performance of the different detectors and features as described in Section 3.3.4: 60 test images (depicted in Appendix A) are selected randomly and for each of them their respective 19 most similar images due to their topic-model-based representations and the L1 distance measure are determined. Then we ask eight users to judge the retrieval results by counting the number of correctly retrieved images for each query (including the query image itself). The average over all users and queries is then used as the final score of the respective detector/descriptor combination.

Comparison:
Figure 4.12 compares the resulting scores for our investigated features. The vertical bars mark the standard deviation between users.

Feature extraction at points on a dense grid over several scales outperforms the sparse DoG feature detection. This has also been observed in the context of scene recognition. One reason for this behavior may be, as in the scene recognition task (see above), that the bag-of-words image descriptions and therefore also the topic model represents the entire image and not only salient image regions. This is especially advantageous in query images that contain less structure or texture.

Furthermore it can be observed from Figure 4.12 that SIFT shows improved performance over the self-similarity descriptor for the dense grid detector. This is surprising as in the closely related scene recognition task geometric blur and self-similarity seemed to be more suited. It indicates that the appropriate choice of the descriptor might be more dependent on the database

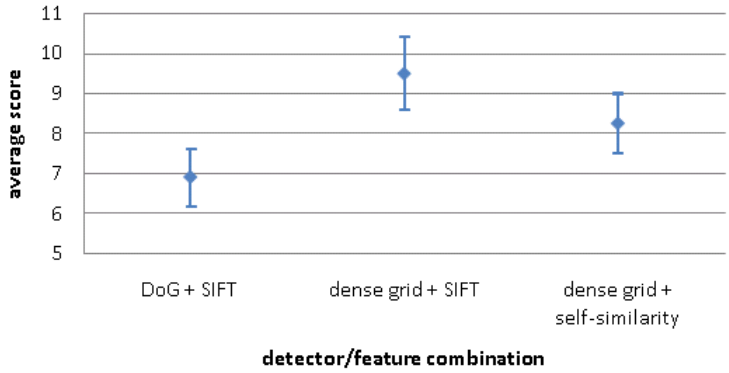


Figure 4.12: Comparison of local region detectors and descriptors for the image retrieval task.

Figure 4.13: Result obtained by our retrieval system using a dense grid region detector and the SIFT descriptor. The top left image shows the query image and the remaining images show the 19 most relevant images retrieved.

used and less on the type of content analysis task one is trying to solve. One should however note that the SIFT descriptor has 128 dimensions whereas the self-similarity feature consists of only 80 dimensions, which results in a faster vocabulary and bag-of-visual-words image representation computation.

We show two example retrieval results for different local region detectors and descriptors in Figure 4.13 and 4.14.

4.2. Fusion Models

In the previous subsection we have investigated various types of basic local image descriptors to build visual words. But much as other authors in their previous works [54, 15], we have considered only one local image descriptor type in our model. However we believe that the results in our retrieval task can be improved for some object categories and scenes by fusing

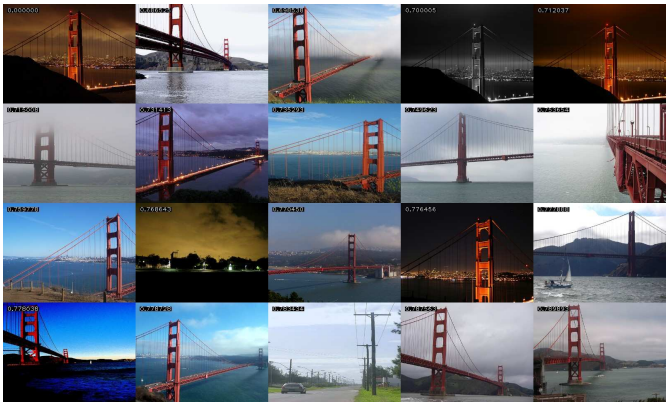


Figure 4.14: Result obtained by our retrieval system using a dense grid region detector and the self-similarity descriptor. The top left image shows the query image and the remaining images show the 19 most relevant images retrieved.

different local features, e.g., local texture and color descriptors, in our image representation. Previous works have fused local image descriptors directly during feature generation on the feature level [7, 12]. In [77] Quelhas and Odobez study two fusion approaches to combine color and texture information in a bag-of-visual-words representation, but they do not apply a topic model for deriving a high-level image representation. Thus, in the following we investigate the possibilities of fusing different types of visual words in the context of topic models. We present three approaches for fusion of different feature types, where fusion will be carried out at different stages of the models: at the visual word level, during topic generation and at the decision level.

As not all object classes benefit from visual descriptor fusion, we further examine which categories are best modeled by only one feature type and which category models are improved by taking into account different kinds of visual words. Here we compare the best-performing fusion model to simpler models using only one feature type.

We focus our study on the fusion of two descriptors, to a texture descriptor, the SIFT feature, and a simple color descriptor, namely color patches. However the proposed approaches can be applied similarly to more than two or any type of (visual) features. The detailed description of the computation and implementation of the used visual features is postponed until Section 4.2.3.

We build a visual vocabulary for each feature type separately as described in Section 3.1. Given the vocabulary for each feature type, we are able to describe each image as a collection of visual words (respectively, as a bag-of-visual-words model) by replacing each detected feature vector in an image by its most similar visual word of the same type: most similar is here defined as the closest word in the 128-dimensional (SIFT) or 149-dimensional (color patch) vector space.

We will limit our studies to the case that in each image d_i the same number of N_i (depending on the images' size, texture, etc.) color patch and SIFT features are extracted. Moreover color patch and SIFT features fused in our models are extracted at the same interest points and with

the same scale. Thus, for example we will consider color patch and SIFT words either both densely detected or both sparsely detected. This procedure enables us to fuse image descriptors directly at the word level in such a way that color patch and SIFT word occurrence at the same interest point are already fused while building the bag-of-words model (see fusion model B, Section 4.2.1).

4.2.1. Models

Now we will present our investigated fusion models. Note that all model are based upon the LDA model. Similar fusion models can be derived from the pLSA and CTM model.

Fusion Model A

Our first proposed fusion model consists basically of two completely independent learned LDA representations for the images in the database. One LDA model is learned for the bag-of-words image representation based on the color patch vocabulary and one for the representation based on SIFT features. The fusion is performed at the decision level, i.e., topic distributions are computed independently, and fusion of those two LDA models is carried out while measuring similarity during retrieval (see Section 4.2.2).

It should be noted that in this model topics are not 'shared' between features. Thus a topic is either purely a color patch topic or a topic defining a distribution over texture words. Topics, which are characterized by both color and texture, are not properly modeled here. However, the separation might be beneficial if combined with some active learning retrieval system. Such a system could learn through user feedback whether one or both features and thus the corresponding topics are important to find images of similar content. It should also be noted that in this model both features have the same weight in the final image representation.

The graphical representation of the LDA-based fusion model A is shown in Figure 4.15 (a). M indicates the number of images in the entire database and N_i denotes the number of visual words of each feature type that are detected in image d_i.

Fusion Model B

The second model fuses the feature types at the visual word level and assumes a joint observation of a color patch word and a SIFT word. Thus, each time a topic z_n is chosen, a color-patch word c_n and a SIFT word t_n – both coming from the same interest point and scale – are sampled from a multinomial probability conditioned on the topic z_n. Here we explore the fact that in each image we compute color patch features and SIFT features at the same locations and scales, resulting in the same number of features for both types.

In this model we have a joint distribution over color and texture words for each topic. The

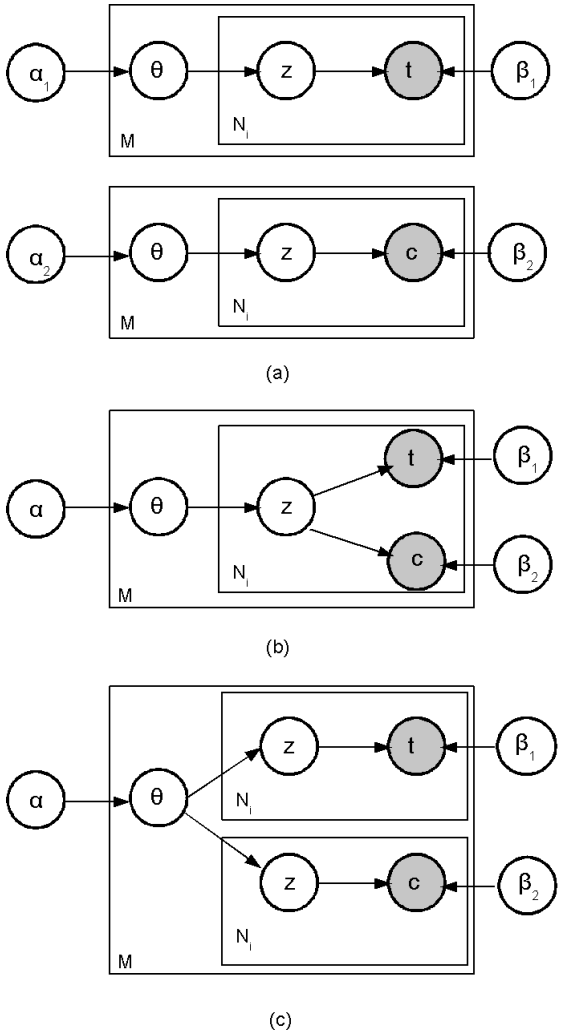


Figure 4.15: Graphical representation of the LDA-based fusion models: (a) fusion model A; (b) fusion model B; (c) fusion model C. M denotes the number of images in the database and N_i the number of detected visual words of a certain feature type in image d_i. The shaded nodes denote the observable random variables c and t for the occurrence of a color patch or SIFT word, respectively. z denotes the topic variable and θ the topic mixture variable.

likelihood of the occurrence of a combination of a specific texture word t_j and a color patch word c_j in an image according to this model is then given by:

$$P(t_j, c_j|\alpha, \beta) = \int P(\theta|\alpha) \left[\sum_{k=1}^{K} P(z_j = k|\theta) \cdot P(t_j, c_j|z_j = k, \beta) \right] d\theta \tag{4.6}$$

Note that this model does not allow topics representing only visual words of one feature type, as visual words are already fused at the word level.

The graphical representation of the LDA-based fusion model B is shown in Figure 4.15 (b).

Fusion Model C

The third model aims to enable topics to represent either words of only one of the feature types or a combination. Here the latent topics for each sampled visual word (either color-patch or SIFT) can vary while the topic mixture θ is fixed; thus θ denotes a probability distribution over the hidden topics and in turn each visual word originates from one of those topics. This is nothing else than concatenating the collection of visual words of both types to describe an image d_i, i.e., we can represent d_i by $\mathbf{w_i} = \{t_1, t_2, ..., t_{N_i}, c_1, c_2, ..., c_{N_i}\}$. The likelihood of an image d_i according to this model is then given by:

$$P(\mathbf{w_i}|\alpha,\beta) = \int \left[P(\theta|\alpha) \left(\prod_{j=1}^{N_i} \sum_{k=1}^{K} P(z_j = k|\theta) P(t_j|z_j = k, \beta_1) \right) \cdot \left(\prod_{j=1}^{N_i} \sum_{k=1}^{K} P(z_j = k|\theta) P(c_j|z_j = k, \beta_2) \right) \right] d\theta \quad (4.7)$$

The graphical representation of the LDA-based fusion model C is shown in Figure 4.15 (c).

It should be noted that although the model allows topics purely representing words of one type of local descriptor, describing images that contain objects only characterized by one feature type (e.g., texture) is not possible as every visual word needs to be 'explained' by one topic. Thus the topic distribution will have to account for words based on the second visual descriptor type (e.g., color patch words), too. This problem could be solved by using a relevance feedback algorithm as e.g., the active learning approach presented in Section 3.4. Here the system is able to learn the components of the image representation from user feedback, e.g., the topic distribution, that are important to separate relevant from irrelevant images.

Parameters of all three fusion models are calculated by variational inference as described in [14]. Again, learning the models involves finding the parameters α_n and β_n such that the log marginal likelihood of the training set is maximized. Probabilities are assigned to all images in the database by maximizing the log marginal likelihood of the respective image given the corpus level parameters.

4.2.2. Image Similarity Measure

In our considered example-based image retrieval task, we search in the database for those images with content most similar to a given query image. Thus, once we have trained one of the LDA-based fusion models and computed a probabilistic representation for each image in the database based on those, we need to define a similarity measure in order to perform image retrieval.

In Section 3.3.4 we investigated various similarity measures for image retrieval based on the

topic mixtures, denoted by θ, that indicate to which degree a certain topic is contained in the respective image. We choose the distance measure which has been adopted from language-based information retrieval and which outperformed all other similarity measures in our retrieval task. As described in more detail in Section 3.2, each document is indexed by the likelihood of its model generating the query document, i.e., the most relevant documents are the ones whose model maximizes the conditional probability on the query terms. Assuming that we represent a query image d_a as a sequence of N_a visual words $\mathbf{w_a}$, we can write this likelihood as in Equation 3.5. Applying this measure to our three fusion model leads to:

Fusion Model A:

$$P(\mathbf{w_a}|M_b) = \prod_{j=1}^{N_a} P(t_j^a|M_b^t) \cdot \prod_{j=1}^{N_a} P(c_j^a|M_b^c) \tag{4.8}$$

We compute two independent LDA models for each type of visual vocabulary, thus we have two models for image d_b, M_b^t denotes the model based on the texture vocabulary and M_b^c the one stemming from the color patch vocabulary, respectively. The total number of visual words in one image is given by $2 \cdot N_a$ as we extract N_a color patches and the same number of SIFT features in image d_a.

Fusion Model B:

$$P(\mathbf{w_a}|M_b) = \prod_{j=1}^{N_a} P(t_j^a, c_j^a|M_b) \tag{4.9}$$

Here each term w_j^a in the document is built from a combination of a color patch and a SIFT word, i.e., $w_j^a = \{t_j^a, c_j^a\}$ and thus each image gives rise to N_a combined terms.

Fusion Model C:

$$P(\mathbf{w_a}|M_b) = \prod_{j=1}^{N_a} P(t_j^a|M_b) \cdot \prod_{j=1}^{N_a} P(c_j^a|M_b) \tag{4.10}$$

Again each image d_a is represented as a collection of $2 \cdot N_a$ visual words. Compared to model A, we also have two kinds of words, but only one model.

In Section 3.2 we describe in detail the IR measure first introduced by Wei and Croft [103]. It combines a topic model and the simple unigram model with Dirichlet smoothing to estimate the terms $P(w_j^a|M_b)$. We will now apply it in the context of fusion models. Equation 3.6 then turns into:

$$P(w_j^a|M_b) = \lambda \cdot P_u(w_j^a|M_b^u) + (1-\lambda) \cdot P_{f_m}(w_j^a|M_b^{f_m}) \tag{4.11}$$

where $P_u(w_j^a|M_b^u)$ is specified by the unigram document model with Dirichlet smoothing [106] according to Equation 3.7. The maximum likelihood probabilities $P_{ML}(w_j^a|M_b^u)$ and $P_{ML}(w_j^a|D)$

in Equation 3.7 are measured separately for each vocabulary type if fusion model A or fusion model C is considered. For model B those likelihoods are calculated for the joint visual words $\{t_j^a, c_j^a\}$.

The term $P_{f_m}(w_j^a|M_b^{f_m})$ in Equation 4.11 refers to the probability of a visual word (combination) w_j^a in image d_a given the currently considered fusion model m of image d_b. These probabilities are given by:

Fusion Model A:

$$P_{f_A}(w_j^a|M_b^{f_A}) = \sum_{k=1}^{K} P(w_j^a|z_j = k, \beta) \cdot P(z_j = k|\theta^b, \alpha) \tag{4.12}$$

where w_j^a may denote a color c_j^a or texture t_j^a word and the according LDA model representation of image d_b, i.e., its topic mixture θ^b, is applied.

Fusion Model B:

$$P_{f_B}(w_j^a|M_b^{f_B}) = P_{f_B}(c_j^a, t_j^a|\alpha, \theta^b, \beta) = \sum_{k=1}^{K} P(c_j^a, t_j^a|z_j = k, \beta) \cdot P(z_j = k|\theta^b, \alpha) \tag{4.13}$$

Fusion Model C:

$$P_{f_C}(w_j^a|M_b^{f_C}) = \sum_{k=1}^{K} P(w_j^a|z_j = k, \beta) \cdot P(z_j = k|\theta^b, \alpha) \tag{4.14}$$

where w_j^a denotes either a color word c_j^a or a texture word t_j^a and the corresponding β has to be inserted.

4.2.3. Experimental Evaluation

Experimental Setup

Again, we perform our experiments on a real-world large scale database consisting of approximately 246,000 images. This database has been used in experiments in previous chapters as well, and details can be found in Section 3.3.1. The models are evaluated in a query-by-example retrieval task and results are judged by ordinary users purely based on the visual similarity of the retrieved images.

As stated above we fuse two types of local features in our models, SIFT and color patches. In our experiments here we consider two different possibilities of defining interest points and scales for feature extraction:

- *Sparse features*: Interest points are detected at local extrema in the difference of Gaussian pyramid [62]. A position and scale are automatically assigned to each point and thus the extracted regions are invariant to these properties.
- *Dense features*: Interest points are defined at evenly sampled grid points. Feature vectors are then computed based on three different neighborhood sizes, i.e., at different scales, around each interest point. These three different scales should allow for a (very) limited degree of scale invariance in the representation.

Then visual features are computed to describe the detected regions of interest: color patch features and rotation invariant SIFT features. Color patch features are computed from normalized 7×7 pixel RGB patches. For each color channel a 49-dimensional feature vector is computed from the patches' pixel values. By combining the values of all three channels we obtain a 147-dimensional feature vector. 128-dimensional SIFT features [62] are computed as described in detail in the previous section.

For both feature types we compute a visual vocabulary from twelve randomly selected non-overlap-ping subsets, each consisting of 500,000 local features. Each of those subsets produces 200 visual words giving a total vocabulary size of 2400 visual words for each type. In order to keep the overall number of visual words approximately constant, we compute for fusion model B only 70 visual SIFT words and 70 color patch words, giving in total 4900 possible combinations of SIFT and color patch words. Vocabularies are computed for sparsely and densely extracted features separately.

The LDA-based fusion models are learned on a training corpus consisting of 25,000 randomly chosen images from the dataset. The number of topics was set to 100 in fusion models B and C, whereas it was set to 50 in each of the two LDA models in fusion model A. This also gives in total 100 topics for model A, 50 for the color patch-based model and 50 for the SIFT-based model.

The Dirichlet prior μ for the IR distance measure was set to 50 for our experiments.

To judge the performance of the proposed fusion models in a query-by-example retrieval task we select a test set consisting of a total of 60 query images (depicted in Appendix A), five query images per category are chosen at random. Our evaluation methodology is then similar to the one described in Section 3.3.4: For each query image the 19 most similar images derived by the distance measure presented in the previous subsection are presented to eight test users. They are asked to judge the retrieval results by counting how many of the retrieved images show content similar to the query image. The average number of similar images over all categories gives the final score for the considered model.

In the second part of our experimental evaluation we study different local descriptors and their combination with respect to their suitability to model various image categories. In these experiments we selected ten images randomly per category from our database. However, the randomly

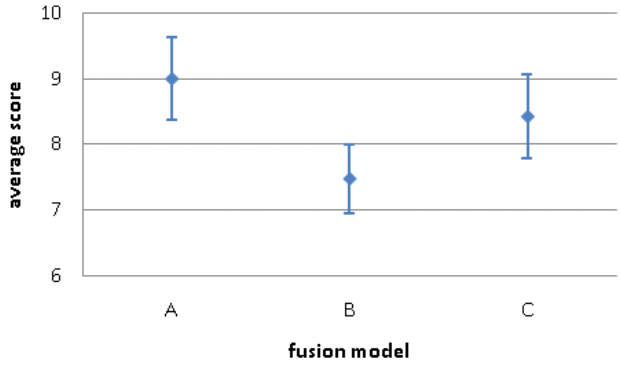


Figure 4.16: Resulting scores for the comparison between the three fusion models applied to sparsely extracted features.

obtained images were filtered to fit the category. For instance a motorcycle image in the category buildings was skipped. This may occur as we divide the database images into the twelve categories shown in Table 3.1 purely based on user tags. Those tags have been assigned to the respective image by their author/owner, thus they are very subjective and result in noisy categories (see Figure 3.3). As we would like to evaluate the suitability of the feature descriptors as well as their fusion to model certain image categories, we need to select our test images accordingly and delete noisy query images.

For these test images, we then compare the retrieval results obtained by the best-performing fusion model to the retrieved images by an LDA image representation based on color patch features and one based on SIFT features. For this purpose we train two 50 topic LDA models, one on the SIFT bag-of-words representation and another on the color patch representation. Evaluation is again performed by user studies as described above, except that the average is computed per category as each category is treated separately.

Different Types of Fusion Models

In our first experiments we aim to evaluate the proposed fusion models. We performed two experiments: In the first one we compared the retrieval results obtained by the models using sparse features as the basic building block, while in the second experiment the models obtained from densely extracted features were used. The results of both experiments are depicted in Figure 4.16 and Figure 4.17. The vertical bars mark the standard deviation of the eight test users' scores.

In both experiments model A performs best, followed by model C. Model B shows the worst performance. As the mean scores for the models A and C are close we conducted a t-test with $\alpha = 0.01$ to verify the hypothesis that model A performs equally well to or better than model C. This hypothesis is valid for both experiments. The results indicate that computing

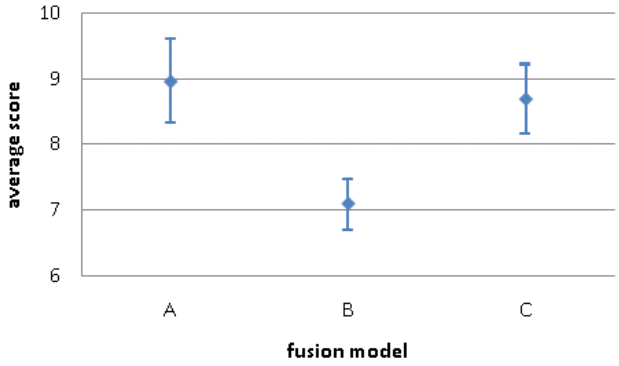


Figure 4.17: Resulting scores for the comparison between the three fusion models applied to densely extracted features.

Figure 4.18: Result obtained by our fusion model A applied to sparsely extracted features. The top left image shows the query image and the remaining images show the 19 most relevant images retrieved.

two separate LDA-models for image representation – one for each feature type – and fusing the information at the decision level (late fusion) gives the best results in the unsupervised retrieval task. Moreover, the computational complexity is lower for model A.

Figures 4.18 to 4.20 display some very good retrieval results obtained by the best-performing fusion model A.

Model Selection

Having determined the most appropriate fusion approach, we will now examine the two different local descriptors, color patches and SIFT, as well as their combination with respect to their suitability to model certain image categories. We consider the twelve categories in our database separately. Figure 4.21 and Figure 4.22 show the results for sparse and dense feature extraction, respectively. For this experiment only five test users were available. We depict their average scores and the most suitable model is marked in yellow.

Figure 4.19: Result obtained by our fusion model A applied to densely extracted features. The top left image shows the query image and the remaining images show the 19 most relevant images retrieved.

Figure 4.20: Result obtained by our fusion model A applied to densely extracted features. The top left image shows the query image and the remaining images show the 19 most relevant images retrieved.

As expected, categories that are highly textured such as *graffiti* and *signs* are best modeled by a SIFT-based LDA model. The *wildlife* category contains also many textured objects such as tigers and lions, whereas the *bird* category is best described by color and shape and thus benefits from the fusion of color patches (which model color as well as intensity changes) and SIFT features. *Flower* retrieval is also improved by the fusion. Altogether, the resulting scores show that many categories benefit from the fusion of both models.

Color patches alone are not appropriate for category modeling, as they only show superior performance in the two categories *food* and *building(s)* if dense feature extraction is considered. It should be noted that the standard deviation between users were large in the *building(s)* and in the *sign(s)* category indicating that the shown content was not obvious and thus it was interpreted diversely by the test users.

category	color-patch	model A	SIFT
wildlife animal(s) cat(s)	2.04	3.56	3.68
dog(s)	3.68	5.24	5.16
bird(s)	4.46	5.08	3.90
flower(s)	11.30	11.78	6.40
graffiti	4.06	7.04	10.38
sign(s)	2.98	3.52	3.86
surf(ing)	8.44	11.28	8.24
night	1.86	3.74	3.74
food	5.18	6.70	3.64
building(s)	2.32	2.56	2.56
goldengate(bridge)	4.38	8.08	10.96
baseball	15.82	16.56	12.32

Figure 4.21: Average scores per category for the comparison between retrieval results based on LDA (fusion) models applied to sparsely extracted features.

category	color-patch	model A	SIFT
wildlife animal(s) cat(s)	2.26	4.08	4.16
dog(s)	4.44	4.78	5.16
bird(s)	5.44	5.86	4.52
flower(s)	9.10	9.82	4.68
graffiti	4.52	5.92	7.50
sign(s)	2.38	2.98	5.12
surf(ing)	8.00	11.20	7.62
night	3.46	3.94	3.90
food	5.54	5.10	4.14
building(s)	3.22	2.90	2.68
goldengate(bridge)	7.88	9.02	8.68
baseball	14.16	14.80	16.66

Figure 4.22: Average scores per category for the comparison between retrieval results based on LDA (fusion) models applied to densely extracted features.

4.3. Summary

In this chapter we have focused on the local image features as the basic building block for our model. In the first section we have examined different local image features in combination with the pLSA model in order to determine their suitability for a scene recognition and a query-by-example image retrieval task. Three different local region detectors, namely the DoG detector, a dense grid over several scales and random edge sampling, have been investigated, as well as three descriptors, the commonly used SIFT feature, the geometric blur feature and the self-similarity descriptor. Our experimental results show that the dense grid over several scales detector performed best in both tasks. The geometric blur feature performed best in the scene recognition task closely followed by the self-similarity descriptor whereas in a retrieval-by-example scenario the SIFT descriptor outperformed the self-similarity descriptor. Thus the appropriate choice of descriptor depends on the database used.

In the second part of this chapter we proposed three different LDA-based fusion models for combining two local image feature types, in our case a texture and a color feature, in order to

take advantage of complementary region descriptions. Although the presented models fused only two different region descriptors, they can be easily extended to combine multiple feature types. The proposed approaches carry out fusion at different stages of the topic model: at the visual word level, during topic generation and at the decision level. Our experimental results showed that a model that fuses the features at the decision level, i.e., that learns two independent topic models, one for each feature type, and fuses the derived image representations during similarity measurement at retrieval time, outperformed the other approaches. In our experiments we also investigated the SIFT feature and the color patch descriptor as well as their combination with respect to their suitability to model certain image categories. It was shown that retrieval results in some categories are improved by fusing two different features, while other categories are better modeled by only one descriptor type.

5. Continuous Vocabulary Models

As topic models have originally been designed for text analysis, words are modeled as discrete variables. In the visual domain we are challenged with the fact that visual features describing an image, especially local image descriptors, are often continuously distributed in some high-dimensional space. Thus visual features are quantized into a fixed-size visual vocabulary in order to be able to apply the original pLSA model to an image analysis tasks.

In most related efforts and also in our system described in the previous chapters, quantization is done by clustering descriptor vectors, representing each cluster by one visual feature vector (the so-called cluster center), and subsequently mapping each feature vector to its closest cluster center in order to get a visual word representing the descriptor. However, the mapping from continuously distributed local features to a discrete visual vocabulary does not necessarily lead to optimal performance since, for example, it does not account for the distance of features to their associated cluster center.

In speech recognition it has been shown that introducing continuous variable models, especially in the case of Hidden-Markov-Models (HMMs), significantly improves performance [105]. In this work we will now consider continuous vocabulary models which do not require a quantization of the high-dimensional feature vectors. In the following we introduce and study models in which continuous visual vocabularies are considered, and thus we model words as continuous, high-dimensional feature vector distributions. We propose three different approaches that extend the discrete pLSA model.

In the context of latent topic models there has been very little work in this area. In order to model annotated data, Blei et al. used a multivariate Gaussian to represent image regions conditioned on a topic variable in two extensions of the LDA [12]. The two most closely related works to our approach are the work by Ahrendt et al. [5] and the work of Larlus and Jurie [52]. The first work [5] proposes the so called Aspect Gaussian Mixture Model (AGMM), which extends the pLSA model to the case of continuous feature vectors. This model is equivalent to our second proposed model, the SGW-pLSA. The model is evaluated in a music genre classification task. However, the authors use supervised training with known concepts for each training sample, while we learn the model's parameters in a completely unsupervised fashion. The second work [52] proposes an extension to a continuous vocabulary for the LDA model. Gibbs sampling is used for parameter estimation, and the model is applied in an object categorization task. In contrast, in this chapter we propose different models and we evaluate them in a scene

recognition task as well as an image retrieval scenario. We consider the pLSA model and we perform parameter estimation via the EM algorithm.

This chapter is organized as follows. We present our three proposed continuous vocabulary pLSA models in Section 5.1. In Section 5.2, we show how parameter estimation and inference is performed for each of the models. Then, in Section 5.3 we evaluate their performance using the results from a discrete pLSA model as the baseline. We describe the experimental setup, as well as show and discuss results.

5.1. Models

As mentioned above, the quantization procedure for continuous feature vectors that is necessary when applying the discrete pLSA model directly to image data is not optimal. We describe now three different ways to model the probability of feature vectors under each topic directly thus making the quantization of the descriptors obsolete.

Ideally we would like to have a separate probability distribution over the feature space for each topic. We do this by using Gaussian Mixture Models (GMM). We call this model GM-pLSA. But a model of this complexity is expensive to train, both in time and data. Thus we also test two simplifications that reduce the model complexity.

In a slightly simpler approach we learn Gaussians that are shared across all topics. Therefore, we propose the SGW-pLSA model that learns the means and covariances of a single set of Gaussians as part of the topic determination algorithm.

A further computational simplification is possible if we cluster the feature data in advance, much as is done for discrete pLSA. Then we represent each cluster by a Gaussian distribution and learn the probability of each cluster for a given topic. This model is called FSGW-pLSA. Figure 5.1 shows an overview of the different model structures.

We represent each image d_i as consisting of N_i local feature descriptors f_j.

5.1.1. pLSA with Shared Gaussian Words (SGW-pLSA)

In the SGW-pLSA approach we modify the original pLSA model such that each word is represented by a multivariate Gaussian distribution´, and we assume that each high-dimensional feature vector is sampled from one of those Gaussian distributions. This results in modeling the topics, i.e., the probabilities $P(w|z)$, by a multivariate mixture of Gaussian distributions, where Gaussians are shared between the different topics.

This approach is similar to the model presented by Larlus et al. [52] for the case of the LDA – a pLSA related model. For the pLSA case a similar model has been presented but the authors

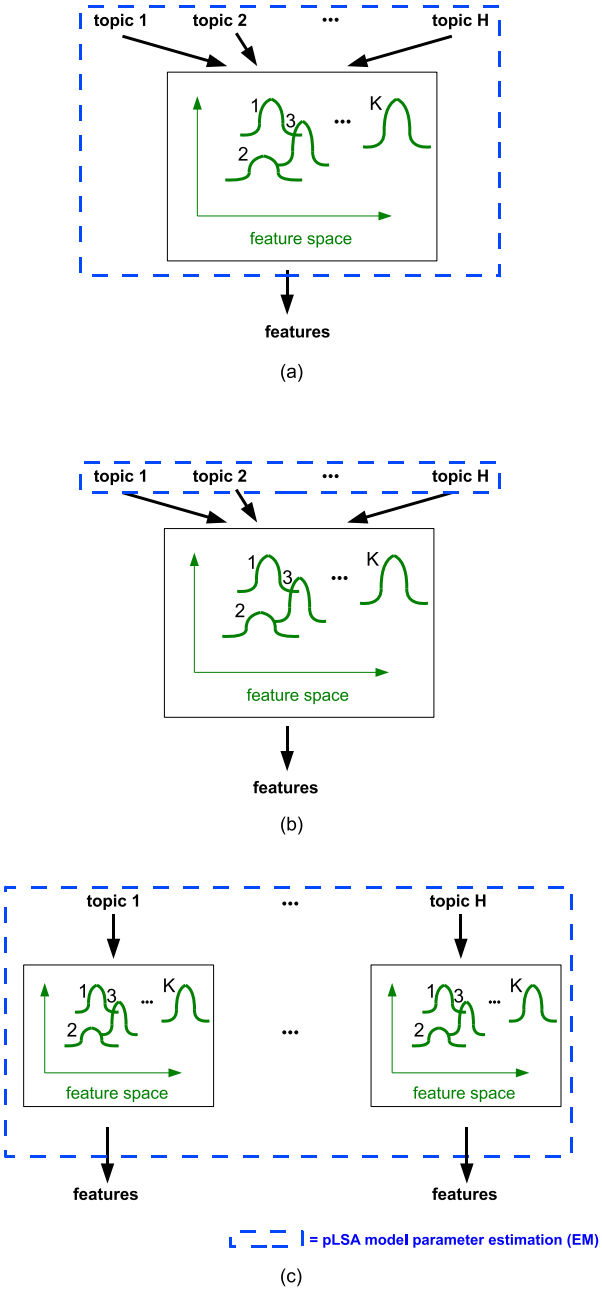


Figure 5.1: Model structure of the three proposed continuous vocabulary pLSA approaches: SGW-pLSA (a), FSGW-pLSA (b), GM-pLSA (c).

consider only supervised learning [5].

The SGW-pLSA model assumes the following process for sampling the n-th feature descriptor f_n in an image d_i:

- Pick a document d_i with prior probability $P(d_i)$.
- Select a latent topic label z_n with probability $P(z_n|d_i)$.

- Choose a Gaussian component g_n depending on the chosen topic z_n with probability $P(g_n|z_n)$.
- Sample a descriptor f_n with probability $P(f_n|g_n)$, which is a multivariate Gaussian distribution over the feature vector space modeling the selected Gaussian component g_n.

According to this generative process Equation 2.5 becomes:

$$P(f_j, d_i) = P(d_i) \sum_{h=1}^{H} \sum_{k=1}^{K} P(f_j|g_j = k) \cdot P(g_j = k|z_j = h) \cdot P(z_j = h|d_i) \tag{5.1}$$

where

$$P(f_j|g_j = k) = N(f_j|\mu_k, \Sigma_k). \tag{5.2}$$

Here H and K denote the total number of the topics and Gaussian words in the model, respectively and μ_k and Σ_k are the parameters of the k-th Gaussian distribution.

It can be seen that the parameters of the Gaussian distributions, i.e., of the continuous visual vocabulary, become part of the model. Thus, those parameters are estimated simultaneously with the other model parameters in the learning algorithm (see Section 5.2). Additionally we can also omit the computation of the co-occurrence table/vector in our retrieval system (see Figure 3.1).

When applying the discrete pLSA model to images the necessary local feature quantization is performed before the actual model is trained and thus the quantization does not account for the probabilistic model learned in the subsequent step. Therefore the joint learning of the Gaussian distributions with the other pLSA parameters may be advantageous. On the other hand, the number of parameters that have to be learned is much higher for the SGW-pLSA model, as additionally the means and covariance matrices have to be estimated.

5.1.2. pLSA with Fixed Shared Gaussian Words (FSGW-pLSA)

In order to examine the influence of modeling the visual words continuously by Gaussian distributions, we proposes the FSGW-pLSA. Here we assume the same probabilistic model as in the SGW-pLSA. However, during the model estimation we do not explicitly estimate the parameters of the Gaussian distributions representing the words.

We learn an ordinary Gaussian mixture model representing the shared continuous vocabulary on the extracted local image descriptors of the training image set in advance. Then, in the subsequent probabilistic model computation of the SGW-pLSA we assume the parameters of the Gaussians, i.e., the means μ_k and covariance matrices Σ_k, are fixed and only the topic and component probabilities $P(z|d_i)$ and $P(g|z)$ are estimated.

Summarizing, in the FSGW-pLSA, the words are modeled by a continuous distribution over the feature space, making quantization unnecessary. In contrast to the SGW-pLSA, the param-

eters of the Gaussian distributions modeling the continuous visual vocabulary are computed separately previous to the topic model parameter estimation.

5.1.3. pLSA with Gaussian Mixtures (GM-pLSA)

In the above two approaches (SGW-pLSA and FSGW-pLSA) all topics share a single visual vocabulary. It may be beneficial to allow for different means and covariance matrices for each topic, i.e., no sharing of Gaussian components between topics, as this allows the Gaussian components to adapt directly to their topic and thus enables smaller mixtures per topic. This results in modeling each topic, i.e., the probabilities $P(f|z)$, by an individual multivariate Gaussian mixture model over the feature space. Thus given a topic we select a Gaussian mixture component out of the Gaussian mixture model associated with the topic, and depending on the mixture component the feature is sampled.

We assume that each feature f_n in image d_i is generated as follows:

- Pick an image d_i with prior probability $P(d_i)$.
- Select a latent topic label z_n with probability $P(z_n|d_i)$.
- Choose a Gaussian component $g_n^{z_n}$ depending on the topic z_n with probability $P(g_n^{z_n}|z_n)$, where $g_n^{z_n}$ is a Gaussian component associated with topic z_n.
- Sample a descriptor f_n with probability $P(f_n|g_n^{z_n})$, which is a multivariate Gaussian distribution over the feature vector space modeling the selected Gaussian component $g_n^{z_n}$.

According to this generative process, we introduce a multivariate Gaussian mixture over the feature space for each topic z

$$P(f_j|z_j = h) = \sum_{k=1}^{K} P(g_j^{z_j=h} = k|z_j = h) \cdot N(f_j|\mu_{kh}, \Sigma_{kh}) \tag{5.3}$$

where μ_{kh} and Σ_{kh} are the parameters of the k-th Gaussian distribution associated with the h-th topic. This yields the following model for a descriptor f_j in image d_i:

$$P(f_j, d_i) = P(d_i) \sum_{h=1}^{H} \left(P(z_j = h|d_i) \cdot \sum_{k=1}^{K} P(g_j^{z_j=h} = k|z_j = h) \cdot N(f_j|\mu_{kh}, \Sigma_{kh}) \right) \tag{5.4}$$

In contrast to the model described in the previous subsections, here the multivariate Gaussian distributions modeling the feature space are not shared, thus the means and covariances of the K Gaussians are different for each topic. On the one hand this enables to use the optimal means and covariances for each topic. On the other hand, as we need more Gaussians in total to model all topics, the number of parameters in the model is significantly larger for the same number of Gaussians per topic. Having observed that most topics are only represented by a small number of words/Gaussians compared to the entire number of visual words/Gaussians in the model, we

should be able to reduce the number of Gaussians per topic without performance degradations. Thus, in our experiments we use fewer Gaussians to represent a topic compared to the SGW-pLSA and FSGW-pLSA approaches. However the total number of Gaussians and parameters for this third model will still be larger than the number for the other two models.

As in the SGW-pLSA model, in the GM-pLSA model the computation of the Gaussian distributions parameters and therefore the continuous visual vocabulary becomes part of the model estimation. Thus no computation of the co-occurrence table/vector in our retrieval system is needed for this model (see Figure 3.1).

5.2. Parameter Estimation

We will now present the algorithms for parameter estimation and inference in the three proposed continuous vocabulary pLSA models.

5.2.1. SGW-pLSA

According to the SGW-pLSA model (Equations 5.1 and 5.2), the log likelihood l of all images in the database is given by:

$$l = \sum_{i=1}^{M} \sum_{j=1}^{N_i} \log \left(\sum_{h=1}^{H} \sum_{k=1}^{K} [P(d_i) \cdot P(z_j = h|d_i) \cdot \; P(g_j = k|z_j = h) \cdot N(f_j|\mu_k, \Sigma_k)] \right) \tag{5.5}$$

where M denotes the number of images in the database and N_i the number of local descriptors representing the image d_i.

During model estimation we need to learn the topic and component probabilities $P(z|d_i)$ and $P(g|z)$ as well as the parameters of the Gaussian distributions, i.e., μ_k and Σ_k. Due to the existence of the sums inside the logarithm, direct maximization of the log-likelihood by partial derivatives is difficult. Thus we use the Expectation Maximization (EM) algorithm [23]. The EM algorithm is an iterative optimization method that alternates between two update steps. The expectation step (E-step) in the EM algorithm consists of estimating the posterior probabilities for the latent variables taking as evidence the observed data and the current parameter estimates. Thus in the E-step we calculate the variables β_{kh}^{ij}.

$$\beta_{kh}^{ij} = \frac{P(z^h|d_i) \cdot P(g^k|z^h) \cdot N(f_j|\mu_k, \Sigma_k)}{\sum_{h=1}^{H} \sum_{k=1}^{K} P(z^h|d_i) \cdot P(g^k|z^h) \cdot N(f_j|\mu_k, \Sigma_k)} \tag{5.6}$$

where z^h denotes the h-th topic, i.e., $z = h$, and g^k the k-th Gaussian component, i.e., $g = k$. Thus β_{kh}^{ij} can be seen as the probability that the feature f_j in image d_i was generated by the topic z^h and Gaussian g^k.

The M-step consists of maximizing the expected complete data-likelihood $E(l^{comp})$:

$$E(l^{comp}) = \sum_{i=1}^{M}\sum_{j=1}^{N_i}\sum_{h=1}^{H}\sum_{k=1}^{K}\left(\beta_{kh}^{ij}\cdot\log\left[P(d_i)\cdot P(z_j=h|d_i)\cdot\ P(g_j=k|z_j=h)\cdot N(f_j|\mu_k,\Sigma_k)\right]\right) \tag{5.7}$$

Then, the update equations for the M-step become:

$$\mu_k^{new} = \frac{1}{p_k}\sum_{i=1}^{M}\sum_{j=1}^{N_i}\sum_{h=1}^{H}\beta_{kh}^{ij}\cdot f_j \tag{5.8}$$

$$\Sigma_k^{new} = \left(\frac{1}{p_k}\sum_{i=1}^{M}\sum_{j=1}^{N_i}\sum_{h=1}^{H}\beta_{kh}^{ij}\cdot f_j^2\right) - (\mu_k^{new})^2 \tag{5.9}$$

where

$$p_k = \sum_{i=1}^{M}\sum_{j=1}^{N_i}\sum_{h=1}^{H}\beta_{kh}^{ij} \tag{5.10}$$

and

$$P(z^h|d_i)^{new} = \frac{\sum_{j=1}^{N_i}\sum_{k=1}^{K}\beta_{kh}^{ij}}{N_i} \tag{5.11}$$

$$P(d_i)^{new} = \frac{N_i}{\sum_i N_i} \tag{5.12}$$

$$P(g^k|z^h)^{new} = \frac{\sum_{i=1}^{M}\sum_{j=1}^{N_i}\beta_{kh}^{ij}}{\sum_{k=1}^{K}\sum_{i=1}^{M}\sum_{j=1}^{N_i}\beta_{kh}^{ij}} \tag{5.13}$$

In fact the solution to $P(d_i)$ is trivial, thus we will not have to learn those probabilities.

In order to estimate $P(z|d_l)$ for test images d_l, we fix the trained Gaussian mixtures, i.e., $P(g|z)$ and the associated Gaussian distributions, i.e., Σ_k and μ_k, and perform the remaining steps of the above algorithm.

The derivations for these E- and M-step equations can be found in Appendix B.

As the iterative EM algorithm does not necessarily converge to the optimal solution, it is important to initialize the model, especially the parameters of the Gaussian distributions, properly in order to avoid local minima. We initialize the means and covariances of the Gaussians representing the visual vocabulary by computing an ordinary multivariate Gaussian mixture model of size K using all local features extracted in our training images. The topic and component probabilities are initialized randomly. It should be noted that we consider only the case of diagonal covariance matrices in our experiments.

5.2.2. FSGW-pLSA

In order to learn a FSGW-pLSA model, we perform exactly the same EM iteration steps as described in the previous subsection, however we do not update the μ_k's and Σ_k's of the Gaussian distributions in the M-step. Compared to the SGW-pLSA the FSGW-pLSA is computationally less expensive, as the means and covariances of the Gaussian do not have to be estimated in the EM iterations.

To derive the parameters of the Gaussians representing the fixed continuous vocabulary, we compute a multivariate Gaussian mixture model on the local feature vectors of the training set in advance. The Gaussian mixture model computation is initialized with the outcome of a k-means clustering on a feature subset of the training set. Note that again, we only consider the case of diagonal covariance matrices.

5.2.3. GM-pLSA

The log likelihood of the images in the database according to the GM-pLSA model is given by:

$$l = \sum_{i=1}^{M} \sum_{j=1}^{N_i} \log \left(\sum_{h=1}^{H} \sum_{k=1}^{K} \left[P(z_j = h|d_i) \cdot P(d_i) \cdot \; P(g_j^{z_j=h} = k|z_j = h) \cdot N(f_j|\mu_{kh}, \Sigma_{kh}) \right] \right) \tag{5.14}$$

As before, the existence of the sums inside the logarithm makes direct maximization of the log-likelihood by partial derivatives difficult. Thus we again use the EM algorithm to iteratively estimate the parameters. We derive the following update equation for the variables β_{kh} in the E-step:

$$\beta_{kh}^{ij} = \frac{P(z^h|d_i) \cdot \pi_{kh} \cdot N(f_j|\mu_{kh}, \Sigma_{kh})}{\sum_{h=1}^{H} \sum_{k=1}^{K} P(z^h|d_i) \cdot \pi_{kh} \cdot N(f_j|\mu_{kh}, \Sigma_{kh})} \tag{5.15}$$

where we introduce the notation π_{kh} for the probability of the k-th Gaussian component associated with the topic h, i.e., $\pi_{kh} = P(g^{z=h} = k|z = h)$.

The M-step updates result in:

$$\mu_{kh}^{new} = \frac{1}{p_{kh}} \sum_{i=1}^{M} \sum_{j=1}^{N_i} \beta_{kh}^{ij} \cdot f_j \tag{5.16}$$

$$\Sigma_{kh}^{new} = \left(\frac{1}{p_{kh}} \sum_{i=1}^{M} \sum_{j=1}^{N_i} \beta_{kh}^{ij} \cdot f_j^2 \right) - (\mu_{kh}^{new})^2 \tag{5.17}$$

where

$$p_{kh} = \sum_{i=1}^{M} \sum_{j=1}^{N_i} \beta_{kh}^{ij} \tag{5.18}$$

and

$$P(z^h|d_i)^{new} = \frac{\sum_{j=1}^{N_i}\sum_{k=1}^{K}\beta_{kh}^{ij}}{N_i} \quad (5.19)$$

$$P(d_i)^{new} = \frac{N_i}{\sum_i N_i} \quad (5.20)$$

$$\pi_{kh}^{new} = \frac{\sum_{i=1}^{M}\sum_{j=1}^{N_i}\beta_{kh}^{ij}}{\sum_{k=1}^{K}\sum_{i=1}^{M}\sum_{j=1}^{N_i}\beta_{kh}^{ij}} \quad (5.21)$$

Again, the solution to $P(d_i)$ is trivial and does not need to be estimated in our iterative algorithm. Computing $P(z|d_l)$ for test images d_l is performed by keeping the parameters of the Gaussian mixtures, i.e., μ_{kh}, Σ_{kh} and π_{kh}, fixed and fitting only the $P(z|d_l)$ parameters during the EM iterations.

Appendix B describes in detail the derivation of the update equations.

An important aspect of this model is the choice of the number of Gaussian mixtures per topic. Here we have to trade-off accuracy to represent the feature distributions per topic against computational complexity as well as the ability to fit a model with a very large number of parameters. In addition, due to local maxima, special care has to be taken to initialize the model appropriately. In this work we initialize the parameters of the Gaussian mixtures by using the results of the SGW-pLSA. Only the K most important Gaussians per topic, i.e., the Gaussians with the highest probability of occurrence in each topic, are chosen. All other parameters are initialized randomly. Again we only consider the case of diagonal covariance matrices.

5.3. Experimental Evaluation

We first evaluate the proposed continuous vocabulary models extensively in a scene recognition scenario, since for this task the annotated OT image database [71] is available which enables us to automatically compute performance scores. As we need users to evaluate the results in image retrieval tasks, we subsequently perform only limited experiments for a query-by-example retrieval scenario to verify the results obtained in the closely related scene recognition task.

5.3.1. Scene Recognition

Experimental Setup

For this work we choose the well-known 128-dimensional SIFT [62] features as local image descriptors and a DoG interest region detector. For a detailed description of both the reader is referred to Section 4.1. Note that each image usually leads to a different number of features even

if two images have the same size. The number of feature computed depends on the structure and texture of the image.

After having computed the SIFT feature vectors for each image we perform a whitening PCA to extract the 75 most important components from the 128-dimensional descriptors. This is done by only keeping the 75 components belonging to the largest eigenvalues. The lower dimensionality ensures faster computation of the pLSA models. Our experiments also showed that no/very little performance is lost due to this dimensionality reduction, compared to the original 128-dimensional feature vectors. Note that due to the whitening procedure each dimension has unit variance.

We evaluate the three proposed continuous vocabulary models by means of scene classification experiments on the OT dataset. The OT dataset [71] has already been used for the evaluations in Section 4.1.3 and consists of a total number of 2688 images from eight different scene categories: *coast*, *forest*, *highway*, *inside city*, *mountain*, *open country*, *street*, and *tall building*. Table 4.1 and Figure 4.6 show the number of images as well as sample images for each category, respectively. To assess the performance of the different models each test image is assigned automatically by our system, based on the respective image representation, to one of the eight categories, and the achieved recognition rate is used as the performance measure throughout our experiments.

We proceed as in Section 4.1.3: we divide the images randomly into 1344 training and 1344 test images. We further subdivide the 1344 training images in a training and a validation set of size 1238 and 106, respectively. The validation set is used to find the best parameter configuration for the respective pLSA-based model. In the model we fix the number of topics to 25 and optimize the number, K, of visual words/Gaussian distributions as well as the number of EM iterations performed for the different models. A number of 25 topics has been shown previously to result in good performance on this dataset [15]. It should be noted that pLSA-related models are susceptible to overfitting, thus an early termination may help with this issue.

Having determined the best parameter setting for the number of visual words and EM iterations, we pick the corresponding model and apply it to all training images, i.e., the image set resulting from merging training and validation set, and all test set images in order to derive a topic distribution for each image. The topic mixture for each image is then used to determine the most similar images in the training set to a query (test) image and thus to finally determine the test image's category by applying a k-Nearest Neighbor classifier.

Scene recognition is performed on all images in the test set. Based on these recognition results we compare the different proposed models. We use the performance of the discrete pLSA model as a baseline.

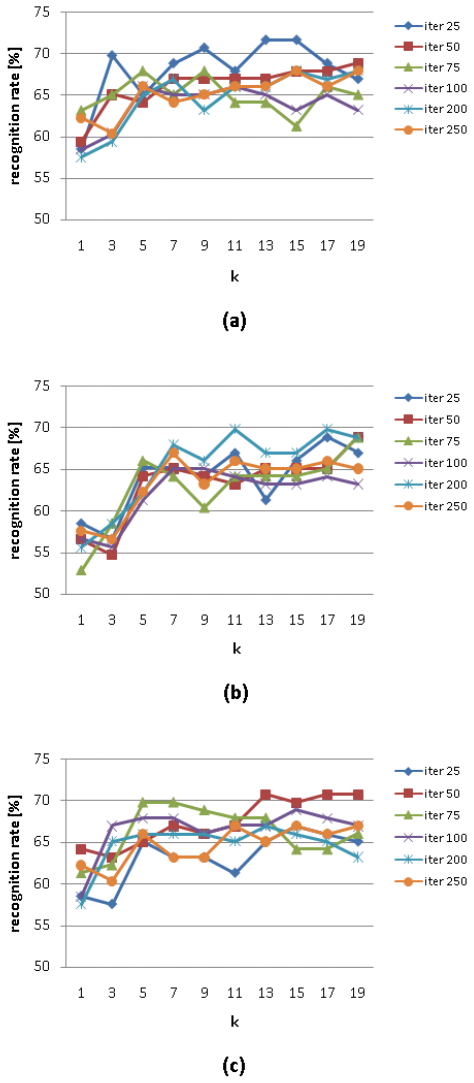


Figure 5.2: Recognition rates of the original pLSA model on the validation set for various k's of the kNN algorithm, different numbers of iterations of the EM algorithm, and different numbers of visual words K in the model: (a) $K = 500$, (b) $K = 750$, (c) $K = 1500$.

Discrete pLSA

The recognition rates of the original discrete pLSA model on the validation set for different numbers of visual words K, different k's of the kNN algorithm, and different numbers of EM iterations are depicted in Figure 5.2. We can see that the best recognition rates on the validation set are achieved for a vocabulary size of 500 using 25 iterations. The best recognition rate of approximately 71% is obtained for $k = 13$ and $k = 15$.

The results of the original pLSA model on the test set using the entire training set for $K = 500$ and 25 EM iterations will serve in this subsection as a baseline for the evaluation of the proposed pLSA models with continuous vocabulary representations.

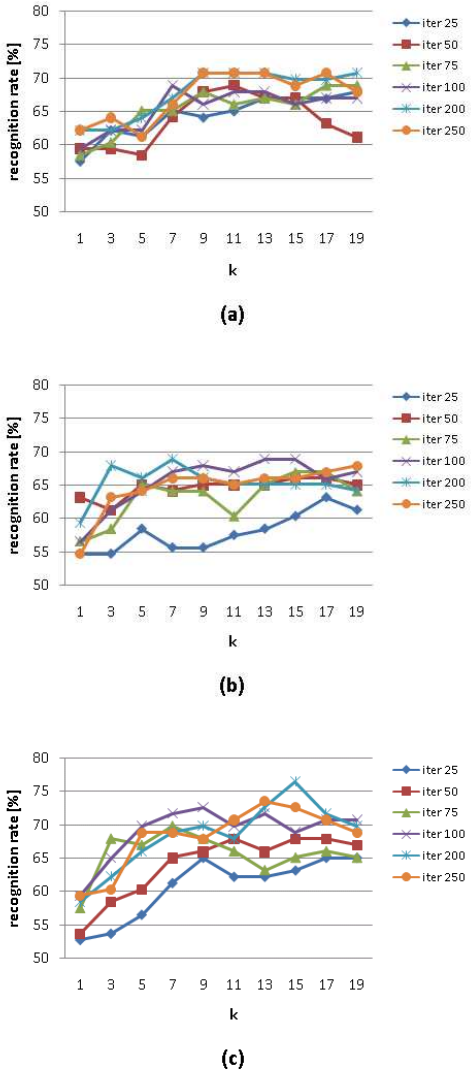


Figure 5.3: Recognition rates of the SGW-pLSA on the validation set for various k's of the kNN algorithm, different numbers of iterations of the EM algorithm, and different numbers of Gaussians K in the model: (a) $K = 500$, (b) $K = 750$, (c) $K = 1500$.

SGW-pLSA

Next we perform the above experiments with varying parameter configurations for the proposed SGW-pLSA model. The results are displayed in Figure 5.3. We clearly see that the performance of a visual vocabulary size of $K = 1500$ is better than the one obtained for 500 and 750 Gaussian distributions in the mixtures, respectively. A recognition rate of about 76% is achieved for $K = 1500$, $k = 15$ and 200 EM iterations. Thus we choose this parameter setting for computing the results on our test set.

It can be also seen in Figure 5.3 that the model needs about 100 iterations to stabilize its performance. Thereafter the performance improves only slightly – in some cases even gets slightly

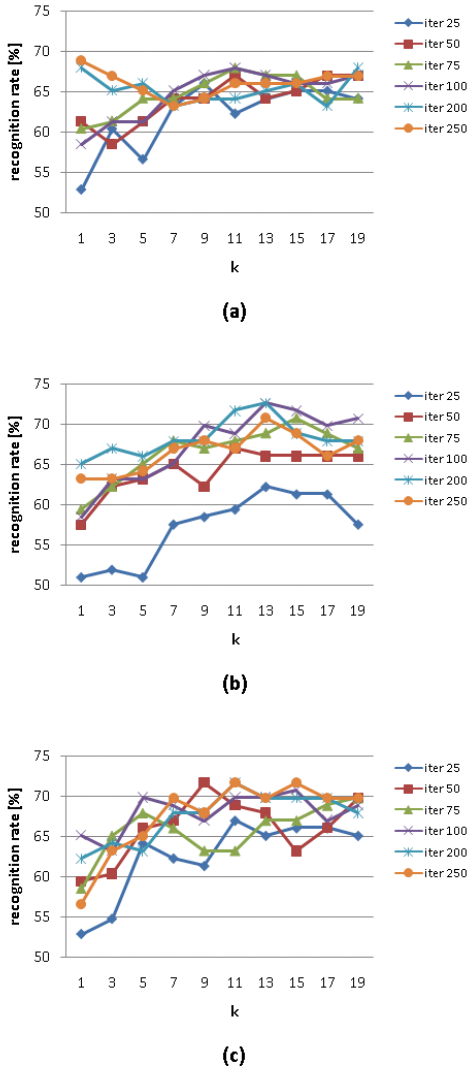


Figure 5.4: Recognition rates of the FSGW-pLSA on the validation set for various k's of the kNN algorithm, different numbers of iterations of the EM algorithm, and different numbers of Gaussians K in the model: (a) $K = 500$, (b) $K = 750$, (c) $K = 1500$.

worse. This could be a sign of overfitting. It should also be noted that a training set size of 1238 images, each producing in average about 550 local descriptors, is relatively small for the number of parameters that need to be estimated for a SGW-pLSA model containing 1500 Gaussian distributions.

FSGW-pLSA

Figure 5.4 shows the results obtained for the FSGW-pLSA model on the validation set. Again the size of the vocabulary, the parameter k in the kNN algorithm and the number of EM iterations in model estimation have been varied.

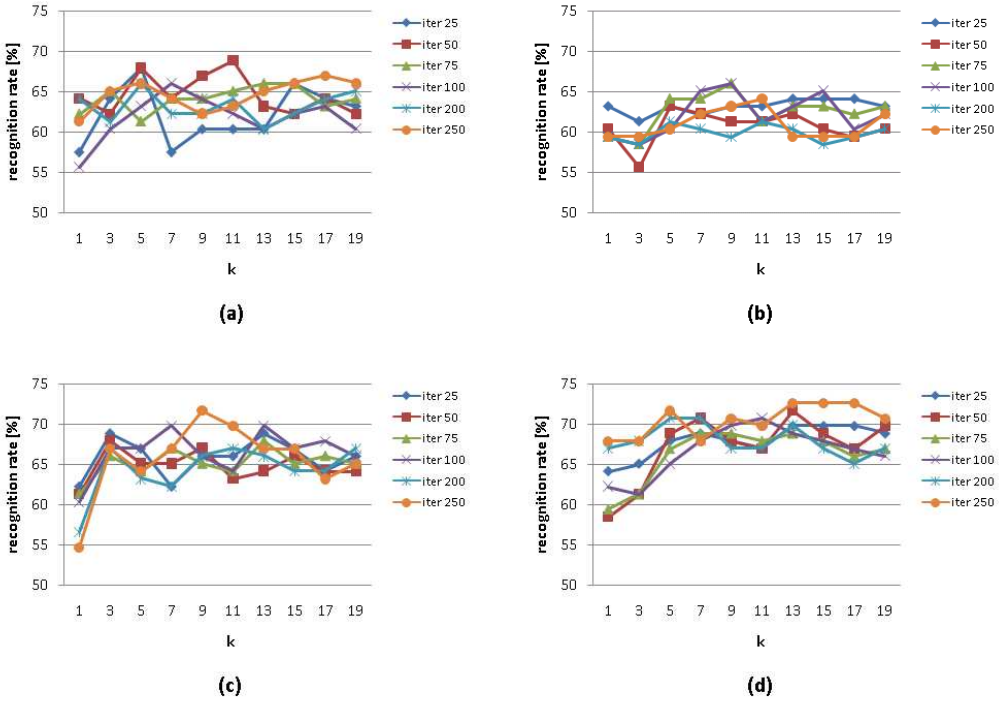


Figure 5.5: Recognition rates of the GW-pLSA on the validation set for various k's of the kNN algorithm, different numbers of iterations of the EM algorithm, and different numbers of Gaussians K in the model: (a) $K = 20$, (b) $K = 30$, (c) $K = 60$, (d) $K = 120$.

It can be seen that vocabularies consisting of 750 respectively 1500 Gaussians give the best results. Especially the results for 1500 words and 250 iterations performed best with a recognition rate of about 72% for $k = 11$ and $k = 15$. As the results for $K = 1500$ and 250 iterations are close to the 70% for a larger range of k values compared to the results for $K = 750$ and 100 EM iterations, we will use the former parameter setting for the final model comparison on the test set.

GM-pLSA

In Figure 5.5 the recognition rates of the GM-pLSA on the validation set for various parameter settings are shown. The number of Gaussians K per topic ranges between 20 and 120. This results in a total number of between 500 and 3000 Gaussians in the model.

The results show that 20 and 30 Gaussians per mixture do not seem to be sufficient as results improve with larger K. The best result is obtained for $K = 120$, i.e., a total number of 3000 Gaussians in the model, and 250 EM iterations. Here we obtain recognition rates of more than 72% for $k = 13, 15, 17$. Thus this parameter setting will be used to compute the performance of the model on the test set.

The results show that the total number of Gaussians in this model needs to be larger than in

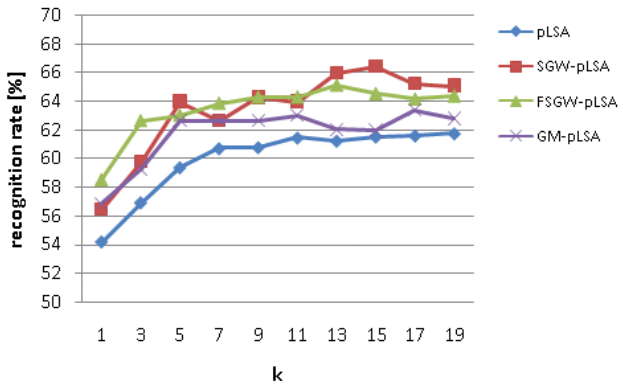


Figure 5.6: Recognition results for all models on the test set.

the previously examined models. Likely, the results will further improve when going to even larger numbers of Gaussian distributions per topic. Nevertheless, this gets computationally very expensive. The required larger total number of Gaussians might be explained by the fact that topics will still partly use similar Gaussians, but those have to be estimated for each topic separately.

Comparison

We will now compare the results of the different models using the parameter sets that have led to the best performance on the validation set. We merge the training and validation set and use the computed models for the selected parameter sets to perform inference on the test set. Given the topic distribution on the test images, each test image is classified based on the dominant scene label in its k-Nearest Neighbor (kNN) set consisting of labeled training images.

Figure 5.6 depicts the achieved recognition rates on the test data set for different numbers of k of the kNN classifier. All three proposed continuous vocabulary models clearly outperform the original pLSA. The best performing model is the SGW-pLSA model, which only slightly outperforms the second best model, the FSGW-pLSA. Both approaches show a performance improvement of roughly 2% to 4% over the pLSA. The third best model, the GM-pLSA shows a recognition rate which is about 1% to 2% above the performance of the pLSA.

It should be noted that in the case of SGW-pLSA, we need to compute the parameters of 1500 Gaussians, whereas in the case of GM-pLSA we compute estimates for a total number of 3000 multivariate Gaussian distributions. Parameter optimization on the validation set has shown that we do need this large number of Gaussians in the GM-pLSA to accurately model the database images. Nevertheless, the surprisingly lower performance compared to the SGW-pLSA may be an result of having not enough training data to reliably learn this large number of parameters in the GM-pLSA model.

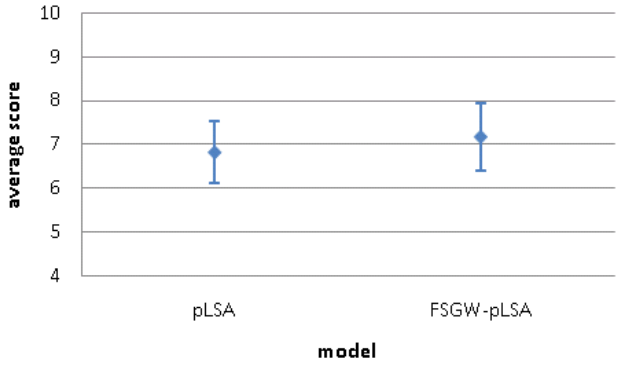


Figure 5.7: Comparison of discrete pLSA and FSGW-pLSA model for the image retrieval task.

In summary, we conclude that a continuous pLSA model describes the visual environment better than a discrete pLSA model as used in previous topic model based scene recognition work. In this application domain the SGW-pLSA model and the FSGW-pLSA model outperform the GM-pLSA model, which has also the disadvantage of being computationally more expensive. Furthermore the performance improvement of the SGW-pLSA over the FSGW-pLSA is small, thus if low computational cost is required one should consider using the FSGW-pLSA over the SGW-pLSA.

5.3.2. Image Retrieval

Experimental Setup

To evaluate our models in a retrieval-by-example task we use the real-world large-scale database described in Section 3.3.1. It consists of about 246,000 images from twelve categories. 60 test images (depicted in Appendix A) are picked randomly, five for each category, and we evaluate the performance of our models by a user study. As the aim of a query-by-example task is to find images with similar content to the query image, we asked eight users to count the correctly retrieved images from the top 20 images, including the query images. The top 20 images are derived for each model by using the L1 norm in combination with the respective topic distribution. The average number of correctly retrieved images over all queries and users is then our final score for the model. This evaluation is similar to the comparisons performed in previous sections.

As the basic image feature we chose sparse SIFT. We first compute 128-dimensional SIFT features for each image in the database and we then perform a whitening PCA to extract the 75 most important components from each 128-dimensional vector, as we did for the scene recognition task. For the discrete pLSA model we derive a visual vocabulary of size 2400 by merging results of k-means clustering on twelve feature subsets, each producing 200 visual

Figure 5.8: Example retrieval result obtained by the FSGW-pLSA model. The top left image shows the query image and the remaining images show the 19 most relevant images retrieved.

Figure 5.9: Example retrieval result obtained by the FSGW-pLSA models. The top left image shows the query image and the remaining images show the 19 most relevant images retrieved.

words (for more details see Section 3.1). For the FSGW-pLSA we also use 2400 Gaussians to model the feature distribution given a topic.

Out of our proposed model, we will only compare the FSGW-pLSA to the pLSA as the baseline. The FSGW-pLSA model showed improved performance over the pLSA in the scene recognition task, but it was outperformed by another examined model, SGW-pLSA. However the SGW-pLSA model is computationally more expensive during training, thus we will show in this thesis only result for the FSGW-pLSA in the context of image retrieval. It is left as future work to examine the other continuous vocabulary models in an image retrieval scenario.

The number of topics is chosen to 50 in both models, the original, discrete pLSA and the FSGW-pLSA. The pLSA was trained on 50,000 images, whereas the FSGW-pLSA was learned from 5,000 images. It should be noted that the parameters of the models such as the number of visual words or Gaussians, number of topics etc. have to be chosen carefully, as there is no ground truth and thus no validation data available to automatically determine suitable values for those parameters.

Comparison

We compare the average result scores of the pLSA and the FSGW-pLSA model based on the eight test users in Figure 5.7, the vertical bars mark the standard deviations. It can be seen that the FSGW-pLSA outperforms the discrete pLSA model. The difference in performance is small but significant as a paired t-test with $\alpha = 0.01$ has shown, thus indicating that using continuous models improves retrieval performance on large-scale databases. This is the same behavior as we experienced in the scene recognition task (see Section 5.3.1).

Furthermore, from the results of scene classification one would expect the SGW-pLSA model to further improve the results. However, showing this is left for future research.

Finally we show some example retrieval results for the FSGW-pLSA in Figure 5.8 and 5.9.

5.4. Summary

In this chapter we have proposed and evaluated three different extensions to the original, discrete pLSA in which continuous visual vocabularies are considered. As we modeled words as continuous, high-dimensional feature vector distributions, the quantization of the local feature vectors that is necessary when applying discrete topic models to image databases becomes obsolete.

For each of our proposed models we have presented algorithms for parameter estimation and inference. The experimental evaluation in an automatic scene classification task shows that the proposed approaches outperform the discrete pLSA model. We we found that the SGW-pLSA performed best closely followed by the FSGW-pLSA. Further, in an query-by-example scenario on a large-scale database, the FSGW-pLSA has shown that it improves retrieval results over the discrete pLSA model. In future work, we will have to examine the performance of the other continuous vocabulary models in an image retrieval task as well.

6. Deep-Network-Based Image Retrieval

The topic models used in the previous chapters to derive a high-level representation of the images' visual content use one hidden layer only. However being able to compute hierarchical models representing the image content with more than one layer of latent variables enables a richer and possibly more powerful image representation.

In this chapter we apply deep network models to derive such a multi-level representation of each image. A deep network (DN) consists of multiple, non-linear, latent feature layers each capturing the strong correlations of the feature activations in the level below. Assuming again a bag-of-visual-words image description as our input, deep networks may reduce the dimensionality of the input vector by decreasing the number of units in each higher layer.

Besides enabling a multi-level image representation the applied deep networks are, once they are trained, used as feed-forward models. In contrast to probabilistic topic models that need to infer iteratively the topic distributions for each new document/image not contained in the training set, the feed forward architecture only requires a matrix multiplication followed by a non-linearity per network unit when computing the image representation. This is an advantage when fast retrieval performance in very large image databases is required.

Previously, deep network models have been successfully applied in the context of information retrieval [83], and they have been used for performing image recognition in [96]. In the latter work the authors use a global image description, and they apply the model to a labeled image database as well as to a web database with images of size 40×40 pixels, containing mostly only one object.

The chapter is organized as follows: We review the deep network model in the first section. Then, in Section 6.2 we describe how the basic model needs to be modified for its application in the image retrieval context. In Section 6.3 we present experimental results, we evaluate the deep model in a retrieval-by-example task and compare its performance to probabilistic topic models.

6.1. Deep Networks

The deep network we adapt in this chapter uses multiple, non-linear hidden layers; it was introduced by Hinton et al. in [39] and [83]. The learning procedure for such a deep model consists of two stages. In the first stage, the pretraining, an initialization based on restricted

Boltzmann machines (RBM) is computed. In the second stage this initialization is refined by using backpropagation.

First we will review restricted Boltzmann machines (RBM). RBMs provide a simple way to learn a layer of hidden features without supervision. They consists of a layer of visible units which are connected to hidden units using symmetrically weighted connections. Note that an RBM does not have any visible-visible or hidden-hidden connections. An example RBM is shown in Figure 6.1.

Assuming binary vectors as our input, the energy of the joint configuration of visible, stochastic, binary units $\mathbf{v}$ and hidden, stochastic, binary units $\mathbf{h}$ is given by [39]:

$$E(\mathbf{v},\mathbf{h}) = -\sum_i b_i v_i - \sum_j b_j h_j - \sum_{i,j} v_i h_j w_{ij} \tag{6.1}$$

where v_i and h_j are the binary states of the visible and hidden units respectively, b_i and b_j their biases and w_{ij} the symmetric weights. The probability of a visible vector $\mathbf{v}$ given this model can be computed as follows:

$$p(\mathbf{v}) = \frac{\sum_{\mathbf{h}} \exp(-E(\mathbf{v},\mathbf{h}))}{\sum_{\mathbf{u},\mathbf{g}} \exp(-E(\mathbf{u},\mathbf{g}))}. \tag{6.2}$$

Given the states of the visible units, the probability that a hidden unit h_j is activated is:

$$p(h_j = 1|\mathbf{v}) = \sigma(b_j + \sum_i v_i w_{ij}) \tag{6.3}$$

where $\sigma(x)$ denotes the logistic function. Similarly it holds that

$$p(v_i = 1|\mathbf{h}) = \sigma(b_i + \sum_j h_j w_{ij}) \tag{6.4}$$

In order to iteratively learn the variables of a RBM, i.e., the weights w_{ij} and the biases b_i, b_j, we apply one step contrastive divergence [38]:

$$\Delta w_{ij} = \varepsilon(< v_i h_j >_{data} - < v_i h_j >_{recon}) \tag{6.5}$$

where Δw_{ij} denotes the update of the weight parameters and ε is the learning rate. $< v_i h_j >_{data}$ is the frequency with which visible unit i and hidden feature j are on together when the features are being driven by the visible data from the training set and $< v_i h_j >_{recon}$ is the corresponding frequency when the features are driven by the reconstructed data. The reconstructed data is derived by applying Equation 6.4 to the stochastically activated features. A similar update equation is used for learning the biases.

To extend this and construct a deep network, Hinton et al. [39] propose to learn additional layers of features by treating the hidden states (or activation probabilities) of the lower-level RBM as the visible data for training a higher-level RBM that learns the next layer of features.

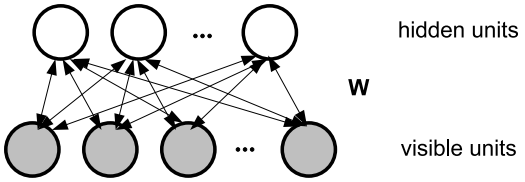


Figure 6.1: Restricted Boltzmann machine.

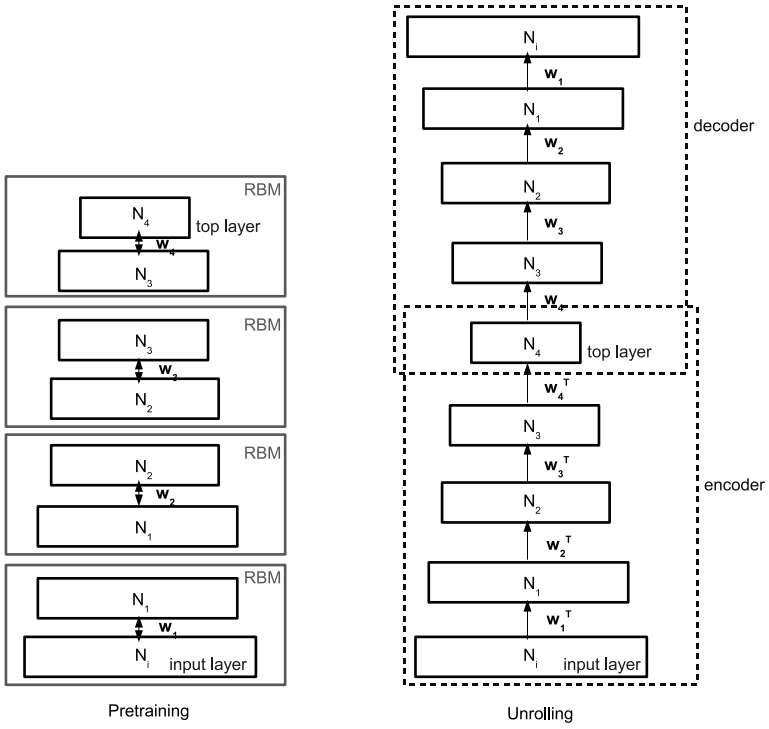


Figure 6.2: Deep network models: layer-by-layer pretraining (left); unrolling and fine-tuning (right).

By repeating this greedy layer-by-layer training several times, we train a deep model that is able to capture higher-order correlations between the input units. This procedure is depicted in Figure 6.2 (left).

After having greedily pretrained all layers, the parameters of the deep model are further refined. This is done by replacing the stochastic activities of the binary features by deterministic real-valued probabilities and unrolling the layers to create an autoencoder as proposed in [39]. On the right side of Figure 6.2 the autoencoder construction is shown. Using the pretrained biases and weights as initializations, backpropagation is used to fine-tune the parameters for optimally reconstructing the input data.

6.2. Image Retrieval

Assuming our original image retrieval system presented in Chapter 3, we start by building a bag-of-words representation for each image based on local features. To derive a high-level representation we then apply a deep network instead of a topic model and use the top level unit values as our final image description. We achieve a dimensionality reduction (similar as in the topic model case) by downsizing the number of hidden units in each layer compared to its previous layer, thus resulting in a low-dimensional top layer-based image representation which enables fast retrieval while using a small amount of memory resources only.

When applying the deep network model described in the previous section to our image data, some modifications to that model are necessary as the input vector, i.e., the bag-of-words representation, at the lowest layer is a visual word count vector and in its general form not binary. Thus, we first divide each entry of the respective vector by the total number of visual visual words detected in the current image. This creates a discrete probability distribution over the finite visual vocabulary for each image.

To model the probability distributions in the input layer, the probabilities of the visible units given the hidden ones can be modeled by a so called 'softmax' unit [39]:

$$p_{v_i} = \frac{\exp(b_i + \sum_j h_j w_{ij})}{\sum_k \exp(b_k + \sum_j h_j w_{kj})} \tag{6.6}$$

where p_{v_i} denotes the value of the i-th visible unit given the hidden ones. The update equations for learning the weights are not affected by this. However, the weights w_{ij} from visible unit i to hidden unit j are multiplied by the number of detected features N_m in image d_m, whereas the weights from hidden units to visible units remain w_{ij}. This is done to account for the fact that each image d_m may contain a different number of visual words depending on its size in case of densely extracted features or size and image structure in case of sparsely extracted features.

All other units in the deep network remain binary. However, there are different possibilities for choosing the type of unit at the top level of the network. In this work we will evaluate two different types of units: logistic and linear.

After pretraining the layers of the deep network, an autoencoder is created as described in the previous section. The parameters of the autoencoder are initialized with the pretrained biases and weights and refined using the backpropagation algorithm. We use the multi-class cross-entropy error function in the backpropagation algorithm:

$$e = -\sum_i v_i \log(\hat{v}_i) \tag{6.7}$$

where $\hat{v}_i$ denotes the reconstruction of v_i by the autoencoder and v_i is the i-th component of the normalized input vector.

Once the deep network is trained we derive a low-dimensional representation of each for our images by applying the learned model to the images in the database and using the respective top-level unit values as their low-dimensional description. It should be noted that the mapping from the co-occurrence vector, i.e., the basic image description, to the high-level representation only consists of a single matrix multiplication and single squashing function per network unit.

Query-by-example image retrieval then proceeds as in the original system described in Chapter 3: Given a test query image, we retrieve images of similar content by comparing the high level image representations based on some similarity measure. In this work we use the simple L1 distance metric.

6.3. Experimental Evaluation

6.3.1. Experimental Setup

All experiments are performed on a real-world database consisting of 246,348 images. This database has been used for evaluations in previous chapters as well, and details can be found in Section 3.3.1.

To evaluate our retrieval approach, we judge its performance by users in a query-by-example task. Here the objective is to obtain images with similar content to the given query image. We selected a total of 60 test query images (depicted in Appendix A) in a random fashion. For each query image the 20 most similar images (including the query image) according to the L1 distance measure are returned to the users.

Our evaluation methodology proceeds then as described in Section 3.3.4: For each experiment we asked ten users to judge the retrieval results by counting how many of the retrieved images show content similar to the query image. As the query image is counted, too, the lowest number of correctly retrieved images is one and the largest 20. The average number of correct result images over all queries and users for one setting, i.e., model, feature and parameter configuration, is computed to give the final score. Note that the users' judgment is subjective. As our test users varied for each experiment, we may derive different average results in different experiments even for the same approach and parameters. Therefore we will also show the standard deviation in addition to the average number of correctly retrieved images.

We evaluate the performance of the deep model for different local feature types and compute the visual vocabulary for each detector/descriptor combination from twelve randomly selected non-overlapping subsets, deriving a total vocabulary of size 2400. The trained deep networks consisted of four hidden layers with a 2400-1000-500-250-50 structure. Thus, we obtain 50-dimensional top-level descriptions for retrieval. We used 50,000 images for training the deep network, 25 iterations for pretraining each layer and 50 iterations to optimize the autoencoder.

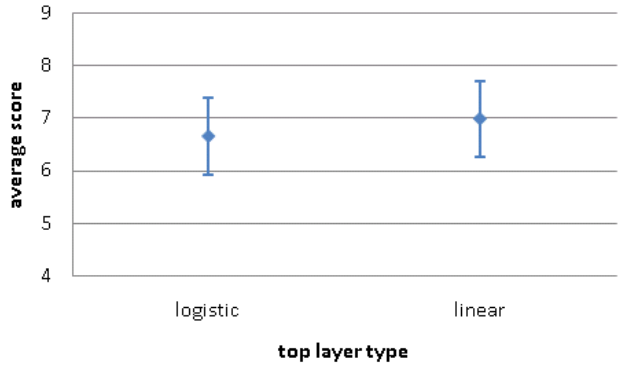


Figure 6.3: Average number of correctly retrieved images using deep-network-based image models with two different types of top layer units: logistic and linear.

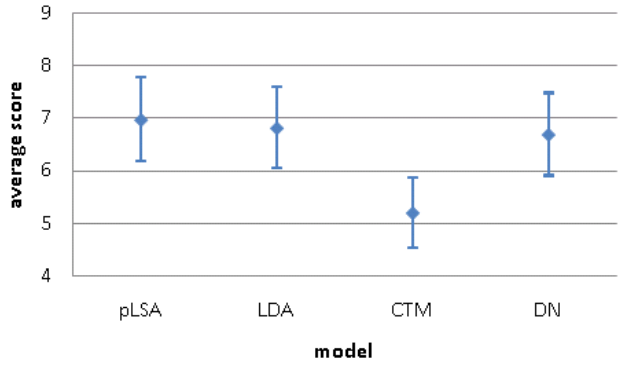


Figure 6.4: Average number of correctly retrieved images using different image models.

6.3.2. Results

First we examine the influence of the top layer type on our retrieval results. We compare logistic units with the linear units using SIFT features extracted at DoG extrema as our basic feature type. As can be seen from the results in Figure 6.3, the performances of both types differs only slightly with a small advantage for the linear units. Performing a paired t-test with $\alpha = 0.01$ also showed that the hypothesis that linear top units perform equally well or better than binary units is valid. Thus we will use the linear units for our subsequent experiments.

Our second experiment compares the results of the deep-network-based image representation to topic-model-based representations, more specifically to the CTM, LDA and pLSA model. All three models also derive a low-dimensional topic mixture-based image representation from the original word count vectors. We train each model with 50,000 images and set the number of topics to 50, resulting in a 50-dimensional topic distribution as image representation.

Figure 6.4 displays the average number of correctly retrieved images for each approach. Again we assume sparsely extracted SIFT descriptors as the basic image feature. Clearly, the CTM

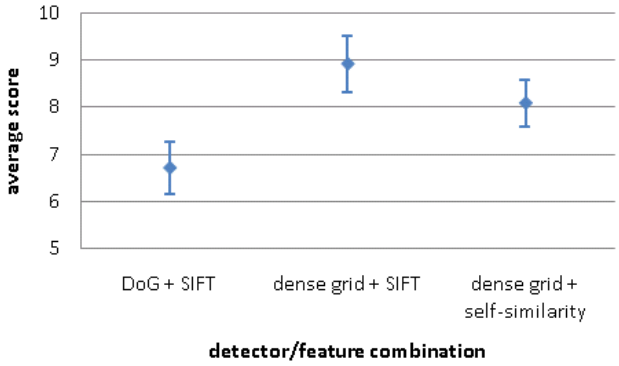


Figure 6.5: Average number of correctly retrieved images using deep-network-based image models with different local image detectors and descriptors: DoG detector and SIFT feature, dense grid detector and SIFT feature, as well as dense grid detector and self-similarity feature.

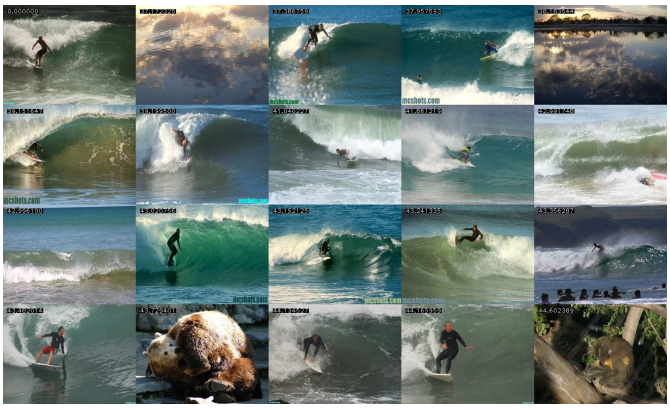


Figure 6.6: Result obtained by the deep network model using a dense grid region detector and a SIFT descriptor. The top left image shows the query image and the remaining images show the 19 most relevant images retrieved.

shows the worst performance. This has already been noticed in the evaluations in Chapter 3. Further, it can be observed that deep network (DN), LDA and pLSA perform almost equally well, with a very slight advantage for the pLSA.

However, a deep network has the advantage of modeling each image by multiple layers of feature activations. Here we only use the highest level in the model to represent an image. Nevertheless, there are possibilities of extending the approach by using multi-level descriptions. Further, the mapping from the high-dimensional word count vector to the low-dimensional representation is much faster for the deep network model compared to inference in the LDA and pLSA model. As inference in those models requires multiple iterations of the (variational) EM algorithm, it is more costly than the feed-forward structure of the deep network, requiring only a matrix multiplication followed by a non-linearity per unit for each layer.

In our last experiment we compare three different combinations of visual features detectors and descriptors as the basic building block in the context of deep networks: DoG in combination

Figure 6.7: Result obtained by the deep network model using a dense grid region detector and a self-similarity descriptor. The top left image shows the query image and the remaining images show the 19 most relevant images retrieved.

with SIFT, a dense grid over several scales detector in combination with SIFT as well as a dense grid detector in combination with the self-similarity feature. The result is depicted in Figure 6.5. It can be observed that dense extraction outperforms the sparse feature extraction. Furthermore the dense SIFT descriptor shows slightly better results than the densely extracted self-similarity features. Note that the results here conform to the ones derived by using a pLSA model in combination with different detectors and descriptors (see Section 4.1.3).

Finally we show two retrieval examples obtained by the deep network model for different types of features in Figure 6.6 and 6.7.

6.4. Summary

In this chapter we applied deep network models to derive a low-dimensional image description. Experimental evaluations showed that the derived image representations are suited for retrieval in large, real-world databases. The deep-network-based description performs as well as image representations based on probabilistic topic models such as pLSA and LDA.

One advantage of the deep network models is their feed-forward structure that, once the network's parameters have been learned, enables very fast mapping from the high-dimensional word count vector to the low-dimensional representation. Moreover deep networks offer a multi-level hierarchical image content description. Exploring this multi-level representation is an interesting direction for future research works.

7. Models for Metadata Fusion

So far our image retrieval system relies solely on visual features to derive a representation for the image content. However in many on-line image repositories of the Web 2.0 images are associated with different types of metadata. Examples of such metadata are geotags, user tags, or the date when the image was taken or uploaded. This metadata provides additional, useful information about the image content and can thus be used to improve visual feature-based retrieval.

In this chapter we present and compare three approaches to effectively fuse different metadata types with the image features. The first approach is similar to the best-performing fusion approach for visual features discussed in Chapter 4, the other two approaches are novel hierarchical models. One of these is based on the pLSA topic model and the other is based on deep networks. All three fusion models start by building a bag-of-words representation for each image and modality. In case of the hierarchical models a hierarchy of topics/features is then used to derive a high-level, low-dimensional image representation based on multiple modalities, whereas in the first examined approach only a single layer of hidden topics is used to model the image content.

The proposed models can be used with any types of metadata, they might even be used to fuse different features within one modality. However, in this chapter we describe the approaches in detail only for the case of fusing two modalities. Nevertheless, from these derivation it becomes obvious how to extend the models as well as their training and inference rules to more than two modalities. We focus our experiments on fusing user-generated tags and visual features.

Previous works on using topic models for annotated image databases include [7], [12] and [68]. Those models were designed to automatically annotate images and/or image regions, and they were trained and tested in most cases on the carefully annotated and almost noise-free COREL database. However, in the here considered Flickr database tags are assigned to an image by its owner, they typically reflect the photographer's personal view with respect to the uploaded image. Thus, in such a real-world database, tags associated with an image do not necessarily refer to the visual content shown (see Figure 3.3), they are subjective, noisy and ambiguous and cannot be used directly for retrieval purposes. Moreover, models that try to associate image regions directly with tags are difficult to learn and apply.

Related work on hierarchical models includes [88] where the the authors adapt the hierarchical Latent Dirichlet Allocation (hLDA) model [11], which has been developed originally for the

unsupervised discovery of topic hierarchies in text, to the visual domain. They use the model for object classification and segmentation. However, appropriate initialization of this complex model is difficult.

The chapter is organized as follows: First we describe our single-layer model adapted from the fusion of multiple feature types explained in detail in Section 4.2. In Section 7.2 we then present our pLSA-based hierarchical fusion model, called mm-pLSA. The deep-network-based fusion model is discussed in Section 7.3. All three modality fusion approaches are then experimentally evaluated and compared in Section 7.4. Here we also give implementation details and details regarding the basic features used for building the bag-of-words representation.

7.1. Metadata Fusion via Concatenating Topic Vectors

In Section 4.2 various probabilistic topic models for fusing multiple types of local image features have been discussed. The experimental evaluation in a large-scale retrieval scenario has shown that a model performing late fusion at the decision level gives the best performance. Assuming two visual features to be fused, this winning model consists of two independently trained LDA models, one for each feature type. Fusion is then carried out at retrieval time while measuring image similarity based on both topic distributions. We will now adapt this model for fusing different modalities, but we will focus on another topic model, the pLSA.

Thus our first proposed modality fusion model is very simple and consists of two separately trained pLSA representations for the images in the database. One pLSA model is trained with the bag-of-words image representation based on content features, i.e., visual words, the second is derived from a bag-of-words description of the images based on associated tags. The fusion of both low-dimensional, but independent high-level image representations is then carried out by simply concatenating both topic vectors, thereby deriving a fused topic distribution based on both modalities for each image. This concatenated topic distribution is then used in combination with the L1 distance measure to find the most similar images.

It should be noted that in this model topics are not ‘shared’ between modalities, and thus we are not able to model topics that are characterized by both modalities properly. Furthermore an equal weight is given to each modality in the final image representation.

7.2. Metadata Fusion via Multilayer Multimodal pLSA (mm-pLSA)

The fusion approach described in the previous section applies the pLSA model to unimodal data only, i.e., separately to tags and visual features. However applying the pLSA to multimodal data directly is challenging. The apparently straightforward application of pLSA to multimodal data

by simply generating one large vocabulary consisting of words from the various modes (which are generally derived from appropriate features of the respective modality) does not lead to the expected improvement in retrieval performance, as will be shown in the experimental results in Section 7.4.

One reason for this may be the difference in the order of magnitude with which words occur in the respective mode. For instance, a few thousand extracted visual features are usually computed for each images, in contrast to an estimated average from about five to 20 tags that are associated with each image. Another reason may be the often occurring difference in the size of the respective vocabularies of each modality.

An approach with top-level topics will solve these issues. Our basic idea is to have in a first layer topic variables for each mode separately, and then to fuse these in a second hidden topic layer where these topics denote distributions over the first-layer topics. Thus, the final representation, the top-level topic distribution for each document, describes each image as a 'distribution over topic distributions' and thereby fuses the visual and the tag-based features.

While we describe this stacking of multiple topic layers in the following only for two modalities, content features and tags, in the first layer and a second topic layer to fuse both, it is obvious that the proposed layering can be extended to more than two layers and applied to more than just two modalities.

The multilayer multimodal pLSA (mm-pLSA) model considering two modes with their respective observable word occurrences and hidden topics as well as a single top-level layer of hidden aspects is graphically depict in Figure 7.1. Assuming a document d_i is represented as a set of words from two different modes x (here: $x \in \{v,t\}$ with v standing for *v*isual and t for *t*ext), where N_i^v denotes the number of visual words in image d_i and N_i^t the number of tags associated with the image, it is generated as follows:

- Pick a document d_i with prior probability $P(d_i)$.
- For each of the N_i^v visual words in document d_i:
  - Select a latent top-level concept label $z_{top,j}$ with probability $P(z_{top,j}|d_i)$.
  - Select a visual topic label $z_{v,j}$ with probability $P(z_{v,j}|z_{top,j})$.
  - Generate a visual word $w_{v,j}$ with probability $P(w_{v,j}|z_{v,j})$.
- For each of the N_i^t tags associated with the document:
  - Select a latent top-level concept label $z_{top,j}$ with probability $P(z_{top,j}|d_i)$.
  - Select an image tag topic label $z_{t,j}$ with probability $P(z_{t,j}|z_{top,j})$.
  - Generate an image tag $w_{t,j}$ with probability $P(w_{t,j}|z_{t,j})$.

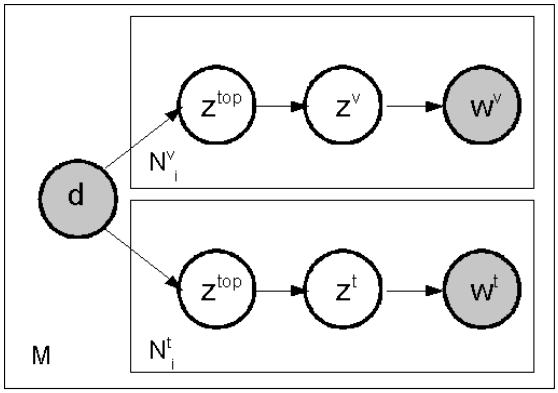


Figure 7.1: Multilayer multimodel pLSA model illustrated by combining two modalities.

Thus the probability of observing a visual word $w_{v,j}$ or a tag $w_{t,j}$ respectively in document d_i is

$$P(w_{v,j},d_i) = \sum_{l=1}^{L}\sum_{k=1}^{K} P(d_i)P(z_{top,j}=l|d_i)P(z_{v,j}=k|z_{top,j}=l)P(w_{v,j}|z_{v,j}=k) \quad (7.1)$$

$$P(w_{t,j},d_i) = \sum_{l=1}^{L}\sum_{p=1}^{P} P(d_i)P(z_{top,j}=l|d_i)P(z_{t,j}=p|z_{top,j}=l)P(w_{t,j}|z_{t,j}=p) \quad (7.2)$$

An important aspect of this model is that every image consists of one or more 'part' topics in each mode, which in turn are combined to one or more higher-level topics. This is very natural since images consist of multiple object parts and multiple objects.

7.2.1. Training and Inference

Given our model (see Fig. 7.1) with its implicit independence assumption between the generated words, the likelihood L for our database consisting of the observed pairs $(d_i,w_{v,m})$ and $(d_i,w_{t,n})$ from both modes is given by

$$L=\prod_{i=1}^{M}\left[\prod_{m=1}^{N_i^v} P(w_{v,m},d_i)\prod_{n=1}^{N_i^t} P(w_{t,n},d_i)\right]. \quad (7.3)$$

where M denotes the number of images in the database. Assuming we have N^v and N^t different words in our visual and tag vocabulary respectively and denoting with w_v^j and w_t^j the j-th words of the visual and tag vocabulary respectively we can write the above likelihood alsoin terms of the co-occurrence table entries:

$$L=\prod_{i=1}^{M}\left[\prod_{m=1}^{N^v} P(w_v^m,d_i)^{n(w_v^m,d_i)}\prod_{n=1}^{N^t} P(w_t^n,d_i)^{n(w_t^n,d_i)}\right]. \quad (7.4)$$

Taking the log to determine the log-likelihood l of the database

$$l = \sum_{i=1}^{M} \left[\sum_{m=1}^{N^v} n(w_v^j, d_i) \log P(w_v^j, d_i) + \sum_{n=1}^{N^t} n(w_t^j, d_i) \log P(w_t^j, d_i) \right] \tag{7.5}$$

and plugging Equation 7.1 and 7.2 into Equation 7.5, it becomes apparent that there is a double sum inside of both logs, making direct maximization with respect to the unknown probability distributions difficult. Therefore, we learn the unobservable probabilities distributions $P(z_{top}|d)$, $P(z_v|z_{top})$, $P(z_t|z_{top})$, $P(w_v|z_v)$ and $P(w_t|z_t)$ from training data using the EM algorithm [23].

As stated before, the expectation step (E-step) in the EM algorithm consists of estimating the posterior probabilities for the latent variables taking as evidence the observed data and the current parameter estimates. Thus we calculate the variables c_{lk}^{im} and d_{lp}^{in} in the E-step:

$$c_{lk}^{im} = \frac{P(d_i)P(z_{top}^l|d_i)P(z_v^k|z_{top}^l)P(w_v^m|z_v^k)}{\sum_{l=1}^{L}\sum_{k=1}^{K} P(d_i)P(z_{top}^l|d_i)P(z_v^k|z_{top}^l)P(w_v^m|z_v^k)}. \tag{7.6}$$

$$d_{lp}^{in} = \frac{P(d_i)P(z_{top}^l|d_i)P(z_t^p|z_{top}^l)P(w_t^n|z_t^p)}{\sum_{l=1}^{L}\sum_{k=1}^{K} P(d_i)P(z_{top}^l|d_i)P(z_t^p|z_{top}^l)P(w_t^n|z_t^p)} \tag{7.7}$$

where z_{top}^l denotes the l-th top-level topic, i.e., $z_{top} = l$, z_v^k is the k-th visual topic, i.e., $z_v = k$, and z_t^p denotes the p-th tag topic ($z_t = p$). Note that c_{lk}^{im} can be seen as the probability of the visual word w_v^m in image d_i being generated by the l-th top level topic and the k-th visual topic. d_{lp}^{in} may be interpreted analogously.

The M-step consists of maximizing the expected complete data-likelihood $E(l_{comp})$:

$$\begin{aligned} E(l_{comp}) \quad = \quad & \sum_{i=1}^{M}\sum_{m=1}^{N^v} n(w_v^m, d_i) \sum_{l=1}^{L}\sum_{k=1}^{K} c_{lk}^{im} \log[P(d_i)P(z_{top}^l|d_i)P(z_v^k|z_{top}^l)P(w_v^m|z_v^k)] \\ & + \sum_{i=1}^{M}\sum_{n=1}^{N^t} n(w_t^n, d_i) \sum_{l=1}^{L}\sum_{p=1}^{P} d_{lp}^{in} \log[P(d_i)P(z_{top}^l|d_i)P(z_t^p|z_{top}^l)P(w_t^n|z_t^p)] \end{aligned} \tag{7.8}$$

For legibility of the M-step estimates, we set

$$\gamma_{lk}^{im} := n(w_v^m, d_i) c_{lk}^{im} \tag{7.9}$$

$$\delta_{lp}^{in} := n(w_t^n, d_i) d_{lp}^{in} \tag{7.10}$$

and obtain:

$$P(d_i)^{new} = \frac{\sum_{m=1}^{N^v} n(w_v^m, d_i) + \sum_{n=1}^{N^t} n(w_t^n, d_i)}{\sum_{i=1}^{M} \left(\sum_{m=1}^{N^v} n(w_v^m, d_i) + \sum_{n=1}^{N^t} n(w_t^n, d_i) \right)} \tag{7.11}$$

$$P(z_{top}^l|d_i)^{new} = \frac{\sum_{m=1}^{N^v}\sum_{k=1}^{K} \gamma_{lk}^{im} + \sum_{n=1}^{N^t}\sum_{p=1}^{P} \delta_{lp}^{in}}{\sum_{l=1}^{L} \left(\sum_{m=1}^{N^v}\sum_{k=1}^{K} \gamma_{lk}^{im} + \sum_{n=1}^{N^t}\sum_{p=1}^{P} \delta_{lp}^{in} \right)} \tag{7.12}$$

$$P(z_v^k|z_{top}^l)^{new} = \frac{\sum_{i=1}^{M}\sum_{m=1}^{N^v} \gamma_{lk}^{im}}{\sum_{k=1}^{K}\sum_{i=1}^{M}\sum_{m=1}^{N^v} \gamma_{lk}^{im} + \sum_{p=1}^{P}\sum_{i=1}^{M}\sum_{n=1}^{N^t} \delta_{lp}^{in}} \tag{7.13}$$

$$P(z_t^p|z_{top}^l)^{new} = \frac{\sum_{i=1}^{M}\sum_{n=1}^{N^t} \delta_{lp}^{in}}{\sum_{k=1}^{K}\sum_{i=1}^{M}\sum_{m=1}^{N^v} \gamma_{lk}^{im} + \sum_{p=1}^{P}\sum_{i=1}^{M}\sum_{n=1}^{N^t} \delta_{lp}^{in}} \tag{7.14}$$

$$P(w_v^m|z_v^k)^{new} = \frac{\sum_{i=1}^{M}\sum_{l=1}^{L} \gamma_{lk}^{im}}{\sum_{m=1}^{N^v}\sum_{i=1}^{M}\sum_{l=1}^{L} \gamma_{lk}^{im}} \tag{7.15}$$

$$P(w_t^n|z_t^p)^{new} = \frac{\sum_{i=1}^{M}\sum_{l=1}^{L} \delta_{lp}^{in}}{\sum_{n=1}^{N^t}\sum_{i=1}^{M}\sum_{l=1}^{L} \delta_{lp}^{in}} \tag{7.16}$$

Clearly Equation 7.11 is constant across all iterations and does not need to be recomputed.

A complete derivation of the update equations can be found in Appendix C.

Given a new test image d_{test}, we estimate the top-level topic probabilities $P(z_{top}|d_{test})$ with the same E-step equations as for learning and Equation 7.12 for $P(z_{top}|d_{test})$ as the M-step. The probabilities of $P(z_v|z_{top})$, $P(z_t|z_{top})$, $P(w_v|z_v)$ and $P(w_t|z_t)$ have been learned from the training corpus and are kept constant during inference.

7.2.2. Fast Initialization

More complicated probabilistic models always come with an explosion in required training time. This issue is becoming the more severe the more hidden layers a model consists of. Moreover a proper initialization of the hidden layer parameters is essential for the model's performance.

Thus, we suggest to compute an initial estimation of the conditional probabilities in a strictly

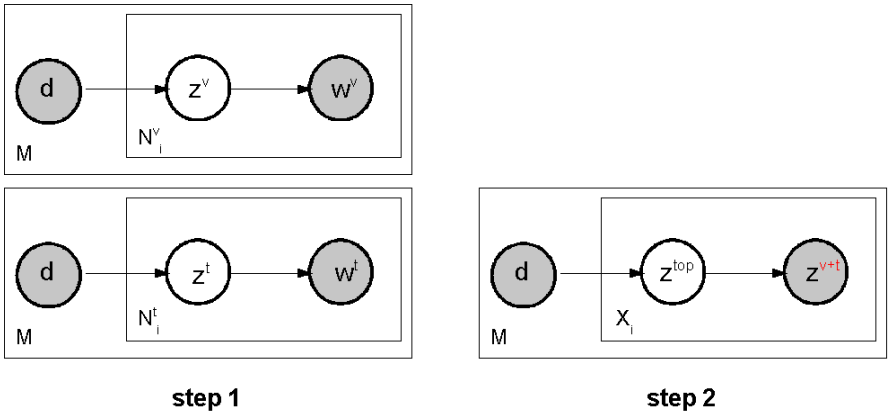


Figure 7.2: Fast initialization of the multilayer multimodal pLSA model computed in two separate steps.

stepwise forward procedure. First we train an independent pLSA model for each modality to initialize the lower layer conditional probabilities. The topics in these two pLSA models are only linked through the documents, i.e., the same images (see step 1 in Fig. 7.2). Next the computed topic mixtures of each modality in this lower layer and for each training document are merged and taken as the observed words at the next higher level (see step 2 in Fig. 7.2). This procedure can continue until the top-level pLSA is trained and thus initial values for all conditional probabilities are found.

It should be noted that this forward initialization is conceptually very similar to the deep network model learning procedure described in Chapter 6.

7.3. Metadata Fusion via Deep Networks

The third approach we propose is to use a deep network as described in [39] and [83] to build a hierarchical model with multiple, non-linear hidden feature layers to fuse different modalities. Such a deep network is trained in two stages. First an initialization based on RBMs is computed which is then refined by using backpropagation. This learning procedure has been described in detail in the previous chapter for the case of training a deep network to represent the visual content of an image.

Similar to the mm-pLSA model presented above, we use one or more layers in the lower levels to learn hidden feature representations for each modality separately. Those separately learned features are then fused by one or multiple higher-level layers. The features in the higher level layers are derived from the separate feature activations of the lower layer features for each modality separately and thus fuse the different modalities.

As before we will in the following only consider the case of fusing two modalities, visual features and tags. However it is straightforward to extend the model to multiple modalities.

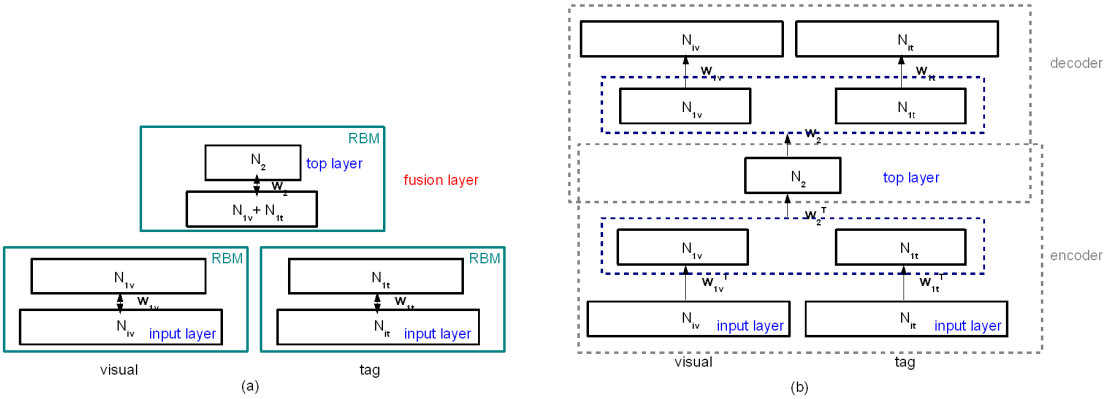


Figure 7.3: Deep-network-based fusion model A: pretraining (a) and finetuning (b).

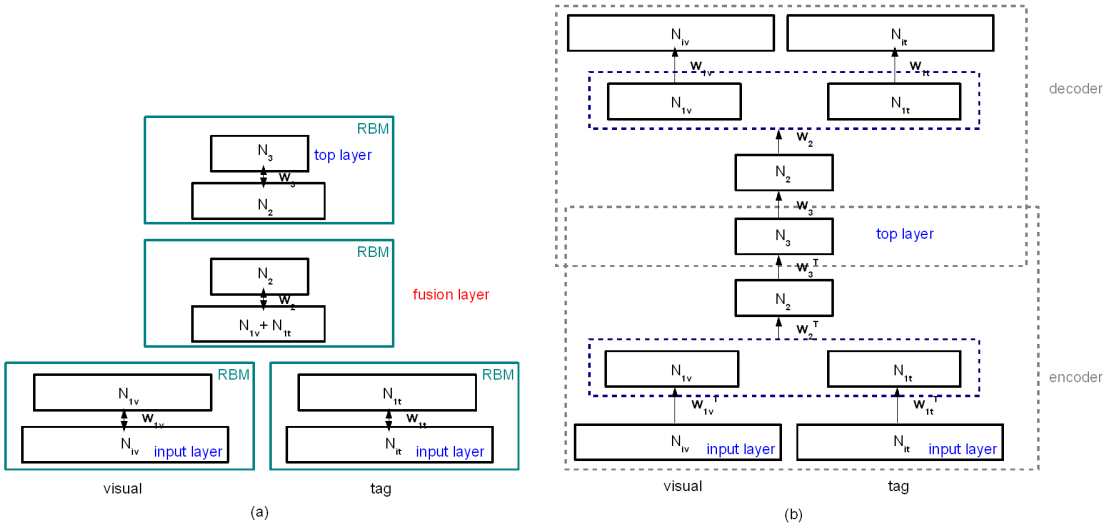


Figure 7.4: Deep-network-based fusion model B with an additional linear layer: pretraining (a) and finetuning (b).

In this work we examine three different deep network model architectures for modality fusion. Our first model, the deep network fusion model A, uses one hidden layer for each of the two modalities separately before fusing them in a second hidden layer. Both hidden feature layers are assumed to be binary. The pretraining procedure that learns and stacks RBMs is shown in Figure 7.3 (left) for this model. After pretraining the model is unrolled to create an autoencoder (see Figure 7.3 (right)), and the model parameters are finetuned to optimally reconstruct the input data.

It has been shown in the experimental evaluation of deep networks on visual features in Section 6.3 that having linear units instead of binary ones at the top level improves retrieval performance. Thus our second architecture, called the deep network fusion model B, has one additional linear hidden layer at the top of the network. The pretraining and unrolling of the resulting network is depicted in Figure 7.4.

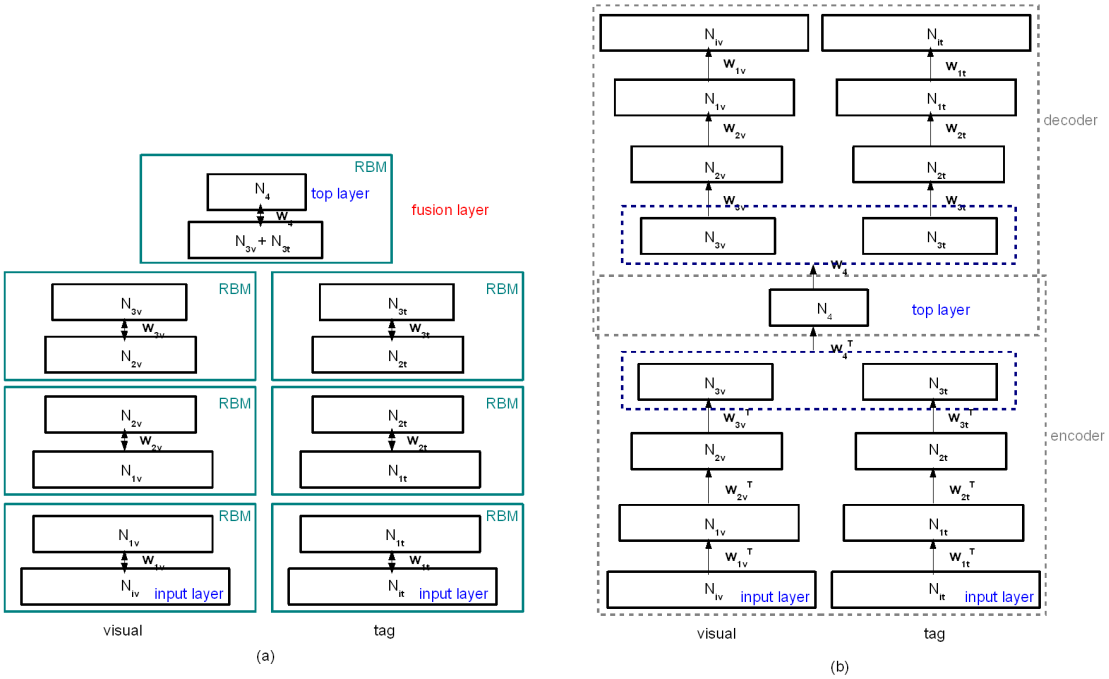


Figure 7.5: Deep-network-based fusion model C with additional feature layers for each modality separately: pretraining (a) and finetuning (b).

Our last model, the deep network fusion model C, uses multiple hidden feature layers for each modality separately before fusing them in a top-level layer. Figure 7.5 shows the structure of the model.

Note that the input to all models is a word count vector for each training image for both, the visual and the tag mode. As this vector is not binary we first derive a discrete probability distribution over the respective finite vocabulary for each modality by dividing the entries of the count vector by the total number of (visual) words detected in the current image. The softmax is used to model the probabilities in the input units as described in the previous chapter.

Having learned the network parameters, we derive a fused image representation appropriate for retrieval by applying the trained model to each image in the database and using its top-level unit values as our image description. By reducing the number of features in each layer we derive a low-dimensional representation enabling fast retrieval.

7.4. Experimental Evaluation

7.4.1. Basic Features

As stated above, we consider two different modalities, visual features and tag-based features, in our evaluation. As all proposed models assume a bag-of-words representation for each modality as their input data, we need to derive a finite vocabulary for both modalities.

The visual vocabulary is computed as described previously. We first detect interest points by densely sampling the images with a vertical and horizontal step size of 10 pixels across an image pyramid created with a scale factor of 1.2. SIFT descriptors computed over a region of 41×41 pixels are then used to describe the grayscale image region around each interest point in an orientation-invariant fashion.

Next these 128-dimensional real-valued local image features are quantized into discrete visual words to derive a finite vocabulary. Quantization of the features into visual words is performed by *k*-means clustering on thirteen small feature subsets, each producing 200 clusters, and taking the cluster centers as our visual words. Merging those visual words into one visual vocabulary results in a vocabulary size of 2600.

We derive a bag-of-words representation for each image based on visual features by mapping the extracted local features to their closest visual word in the high-dimensional space and counting the occurrences of each visual word per image.

The second modality we consider are tags – the free-text annotations provided by the authors of the images. In the following a *t*ag denotes a single word and is used interchangeably with 'word' and 'term'. To build a finite tag vocabulary we start by listing all tags associated with at least one image in our database. Next we filter this list to keep only those tags that are used by a certain number of authors – 30 in our implementation. Additionally we remove all tags containing numbers or special characters.

To map singular and plural forms of objects into one single word, we apply a very simple scheme which pools words that are equal up to the letter 's' at the end of the word. Note a more sophisticated approach to building the vocabulary may be useful to consider for future work, as we do not consider any type of translation of the words.

Our final tag vocabulary consists of 2534 words. Having chosen the vocabulary we can represent each image by a word-count vector, i.e., its bag-of-words model. The resulting representation is in most cases very sparse as users typically only use up to 10 terms to tag an image.

7.4.2. Experimental Setup

We evaluate our approaches on a dataset consisting of 261,901 Flickr images. To derive this dataset we downloaded up to 10,000 images from each of the following 29 categories: *aircraft, beach, bicycle(s), bird(s), boat, bottle(s), building, bus(es), car(s), cat(s), chair(s), city, coast, cow(s), desert, dining table, dog, forest, horse(s), tv monitor(s), motorcycle(s), mountain(s), people, potted plant, sheep, sofa, street(s), table(s), tree(s).* The database has not been cleaned or post-processed. Example images from the dataset can be seen in Figure 7.6. Note that the dataset contains objects as well as scene categories.

To evaluate our simple one-layer fusion approach, we train two 25-topic pLSA models, one based on visual feature and the second one based on tags. Then we concatenate the topic

Figure 7.6: Example images from the database used for experimental evaluation.

vectors of each image to derive a 50-dimensional image representation. The pLSA models are trained using 500 EM iterations and 50,000 images, and for inferring the topic distributions of all images in the database, 200 iterations are used.

Our examined mm-pLSA model consists of 200 topics for each modality in the lowest layer. Then these 400 topics are fused to 50 topics in the second layer. We scale the tag co-occurrence table such that the total number of tags for an image is equal to the total number of visual words detected in order to give both modalities the same weighting in the image likelihood. The pLSA models used for initializing the mm-pLSA are each trained on 10,000 images, whereas the final mm-pLSA model is optimized with 5,000 images. All pLSA models are trained using 500 EM iterations, the mm-pLSA model is trained with 100 iterations.

For the mm-pLSA, the only probability distributions computed during inference are the probability distributions $P(z_{top}|d_i)$ of the top-level topics given the documents. Therefore the EM algorithm converges faster than during training, and the number of iterations can be reduced. Thus to infer these topic distributions for all images in the database we use 50 iterations.

The three deep-network-based models have the following structures: 2600/2534 - 200/200 - 50, 2600/2534 - 200/200 - 200 - 50 and 2600/2534 - 1000/1000 - 500/500 - 200/200 - 50. Thus we obtain a 50-dimensional image description for each topic. We use 25 iterations to train the various RBMs and 50 iterations to finetune the models using for both a training set of size 50,000. The multi-class cross-entropy error function (see Equation 6.7) is applied in the backpropagation algorithm.

We consider three baseline models: First we compute two 50-topic pLSA models, one based on visual features only and one based on tags only. The performance of these models are used to evaluate whether or not we obtain a boost in our retrieval by fusing both modalities. We

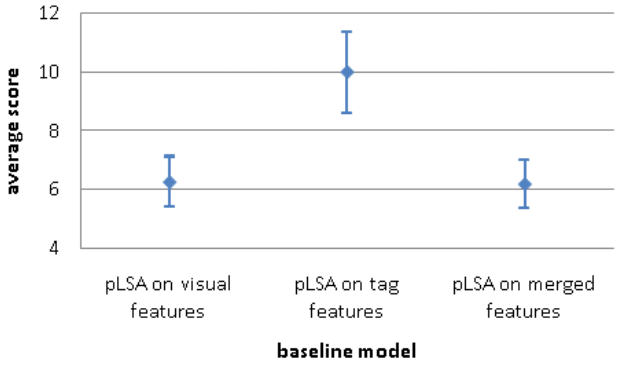


Figure 7.7: Average number of correctly retrieved images using different baseline models.

also train a 50-topic pLSA model based on the concatenated bag-of-words representations of both modalities in order to evaluate the fusion performance of our models compared to this straightforward fusion approach.

We evaluate all models in a query-by-example task by user studies as described in the previous chapters. We select two test images per category resulting in a total of 58 test images. Those test images are depicted in Appendix A. The L1 distance measure is used to find the 19 most similar images to each query image. Eight users were then asked to count the number of result images that show content similar to the query. Note that we only show the test image, not the tags associated with that image as those representing the photographer's personal view about the picture's content. As the query image is counted, too, the lowest number of correctly retrieved images is one and the largest 20. A mean score is calculated for each user, i.e., the average number of correctly retrieved image per query, and the mean over all users' means yields the final score of the approach being evaluated.

7.4.3. Results

In our first experiment we compare our three baseline models. The results are depicted in Figure 7.7. It can be seen that retrieval based on tags outperforms retrieval solely based on visual features. Moreover, simply merging the visual and tag vocabulary by concatenating their co-occurrence tables does not lead to the expected improvement in performance. The performance of using a large, merged vocabulary is similar to the case of using only visual features. This may be due to the fact that the average number of detected visual words in an image is usually much larger than the number of tags associated with the image and thus the trained topics may tend to represent mainly visual word occurrences.

Summarizing we will use the results obtained by the pLSA model based on tag features as the baseline to judge the fusion approaches.

Next, we evaluate the three proposed deep-network-based fusion models. The average numbers

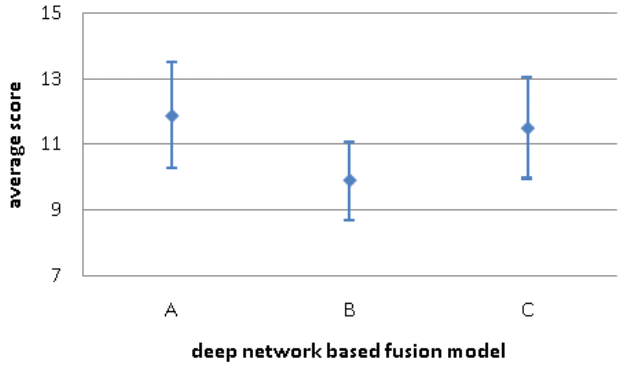


Figure 7.8: Average number of correctly retrieved images using different deep-network-based fusion models.

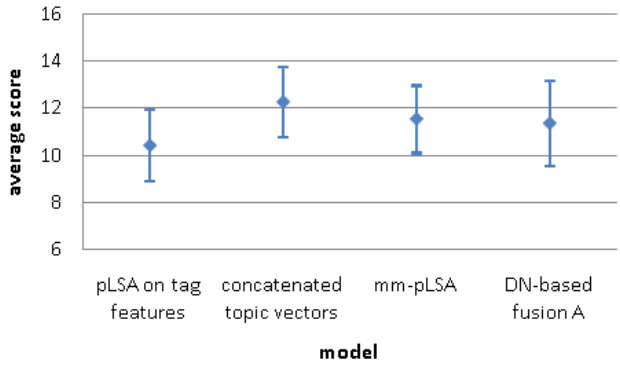


Figure 7.9: Average number of correctly retrieved images using the baseline model as well as the different proposed fusion models.

of correctly retrieved images for the models are shown in Figure 7.8. Here the deep-network-based fusion models A and C perform almost equally well and outperform the model B. This is surprising as using a linear top layer instead of a binary one improved performance in the case of one modality (see Section 6.3).

Moreover, we would have expected the model using multiple hidden feature layers for both modalities, the model C, to outperform the simple model A. The similar performance may be explained by the fact that we finetuned both model parameters using 50 iterations. While the optimization procedure in the case of model A had almost converged, using more iterations in the case of model C would have most likely further improved the reconstruction error. This can be explained by the much larger number of parameters in model C than in model A.

As compared to fusion model C, model A is trained much faster, due to the smaller number of layers, we will use the results of the deep-network-based fusion model A for the subsequent comparison of fusion approaches.

Finally we compare the different fusion models, i.e., the late fusion model based on two sepa-

Figure 7.10: Result obtained by concatenating the separately learned topic vectors for each modality. The top left image shows the query image and the remaining images show the 19 most relevant images retrieved.

Figure 7.11: Result obtained by the mm-pLSA model. The top left image shows the query image and the remaining images show the 19 most relevant images retrieved.

rately learned pLSAs, the mm-pLSA and the deep-network-based fusion model A to our best-performing baseline approach, the pLSA on tag features. The resulting average scores are depicted in Figure 7.9. It can be seen that all proposed fusion approaches outperform the baseline. Our first fusion algorithm, simply concatenating the topic vectors of the two pLSAs, obtained the highest average retrieval score, followed by both almost equally well performing hierarchical approaches. This result is surprising and has to be investigated in more detail in future research work. Part of this behavior may be explained by the small number of noisy tags per image and the very diverse set of images belonging to the respective tag categories. This makes it potentially very difficult to automatically learn correlations given the relatively small size of the training set (5,000 images). Thus training the hierarchical model with a larger number of images, possibly more than 100,000, may lead to much better performance scores.

However, it should be noted that the fused image representation based on deep networks is faster to compute than the pLSA-based descriptions due to the deep networks' feed forward structure. Thus, in cases where very fast inference is needed, one may prefer this fusion model over the

Figure 7.12: Result obtained by the deep-network-based fusion model. The top left image shows the query image and the remaining images show the 19 most relevant images retrieved.

best-performing model based on the concatenated pLSA topic vectors.

We show some example retrieval results obtained by our proposed fusion models in Figure 7.10 to 7.12.

7.5. Summary

In this chapter we have presented work in progress on fusing multiple modalities for image retrieval. We focused in our approaches on fusing two modalities, namely visual features and tags. However, the presented models can be easily extended to more than two modalities.

All three fusion models started by building a bag-of-words representation for each image and modality. Our first approach then learned two independent pLSA models, one for each modality, and fused the separate image representations during similarity measuring by simply concatenating the topic vectors from both models. Our second and third proposed fusion approach were hierarchical models, one based on the pLSA, the other based on deep networks. In both models a hierarchy of topic/features has been used to derive a high-level, low-dimensional image representation.

We have evaluated our proposed models experimentally and we found that all proposed fusion approaches outperform the baseline approach, i.e., the pLSA model on tag features only. Furthermore, our first fusion algorithm, which simply concatenated the topic vectors of the two pLSAs, obtained the highest average retrieval score, followed by both almost equally well performing hierarchical approaches. This surprising result may be due to the relatively small size of our training set. It has to be examined in more detail in future research work.

8. Image Ranking

In this chapter we present an approach for finding the most relevant images, i.e., very representative images, in a large web-scale collection to a given query term. Thus the user enters a query term much as it is done in common text search instead of a query image. In the following we only consider a single term as the query but the extension of our approach to more than one term is straightforward.

We work in a domain with billions of diverse photographs (see Figure 8.1), and we assume that the most representative image is the most likely image related to the query term. This is justified as the most relevant shots or images are taken frequently by many different people, and thus they agree that these are interesting shots. For instance, if there are many images of the *Golden Gate bridge* from a similar angle and under similar lightning conditions, we hypothesize that these pictures capture a very relevant shot. In essence, people are voting for their favorite images of the query term by their picture (camera clicks). Moreover, we determine the most representative images not only based on the image content but based on multiple sources of information, i.e., the image content as well as several metadata types such as tags, date and time, location, etc. For instance, if most people agree and tag a shot of the *Colosseum* with *Italy* and *Rome*, then we use this common metadata agreement to improve our relevance calculation. Note that such an approach can only work when we have a large collection of images, all independently taken by many different people.

In order to determine the most likely image for a given query term, we train a probabilistic model. This model is then used to rank the image highest whose content and its various metadata types give us the highest probabilities according to this model. The proposed model is trained in an unsupervised fashion.

In contrast to previous work, we do not consider only a certain class of images such as landmarks [50] or objects [10] but we describe a general framework for all kinds of images such as objects, places, events, etc. The only category of search we exclude in this preliminary work is the search for images of a particular person, as we believe this should be solved differently, i.e., it may require face recognition techniques because humans are so sensitive to faces.

Previously, there have been many approaches that aim to rank images or text given a query term. Those that use labeled data about the classes perform classification [34]. Other systems use feedback from user searches and then learn the results that are most likely to be selected [48]. Our approach uses none of this information, just the frequency of each type of image. We

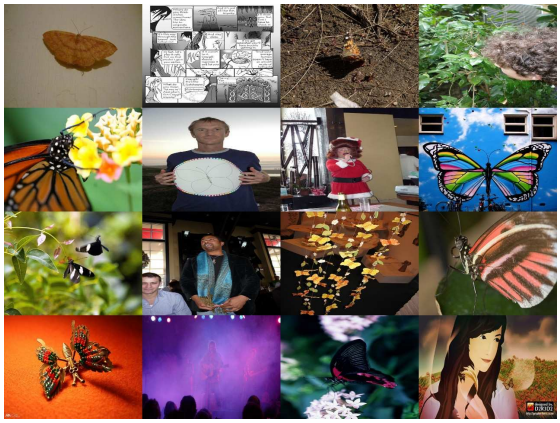


Figure 8.1: Modern image collections are diverse. These images show examples the range of images in Flickr labeled *butterfly*.

call this an unsupervised approach. In a real system one might initially use an unsupervised approach and then augment it with supervised data, i.e., click data, as it becomes available.

There are a number of prior works that address the issue of finding iconic or most representative images from a collection of photos. One approach builds a graph connecting similar images and then uses either eigenvector similarity [47] or spectral clustering [19] to find the images that are at the 'center' of the graph. Similarity in these approaches is computed on an item-by-item basis using feature detectors such as SIFT. Another technique builds clusters of images and then uses either intra-cluster similarity [50] or cluster centroids [10] to find the most representative images. Berg's [10] work, in particular, extends the clustering idea by finding images that have a clear foreground object and thus are more likely to represent 'good' images. The work by Hsu performs pseudo-relevance feedback to improve search ranking using both cluster-based similarity [42] and graph-based similarity [43]. Our work takes a more direct approach for ranking. The earlier works are different approximations for the model we present here. Both tightly coupled graphs of photos and image-cluster centroids are found near the peaks in a probability distribution.

This chapter is organized as follows: In the next section we introduce our model. We evaluate the proposed approach for the case of two modalities, i.e., visual content and tags. We also present the feature we use to model those two modalities in this section, too. Implementation details are given in Section 8.2 and we evaluate our approach experimentally in Section 8.3.

8.1. Model

As stated in the introduction, we assume that a representative image is the most likely image related to the query term. We find such images by looking at peaks in a simple probabilistic model. To train such a model for a given query, we need to start with a large set of images that

may satisfy the query term. These images build the (possibly) very noisy training set for the respective query. In our experiments we consider all images that contain the query term t in their associated tags. Of course, this simple approach can be enhanced by more sophisticated techniques, e.g., query-expansion.

Next we define the model for images tagged with t by

$$P(d_j|t) = P(f_j^1, f_j^2, f_j^3, ..., f_j^n|t) \tag{8.1}$$

where f_j^i is the i-th feature that describes the visual content and/or the metadata of image d_j. To compute our model we first need to choose appropriate features $f^1, ..., f^n$ that accurately reflect the available images, then we build the model that captures the distribution of the images in this feature space.

Even in very large collections such as the Flickr repository, getting enough data to train a full model is prohibitive. So in this work we assume that each feature is statistically independent of the others and thus the image probability in Equation 8.1 becomes

$$P(d_j|t) = \prod_i P(f_j^i|t) \tag{8.2}$$

In cases were enough data is available and correlations between two or more features can be identified, we get better accuracy by building joint probability models of the related features. Dividing entire model into smaller submodels and assuming independence between those models is less restrictive than assuming independence between each and every (metadata) feature but less expensive than computing a joint distribution for all of the cues. Finally, we note that the choice of features is arbitrary, and one can balance the information in different features or even feature dimensions by learning a weight per feature (dimension). Thus, the more general form of the probability model is given by

$$P(d_j|t) = \prod_i [P(F_j^i|t)]^{\alpha_i} \tag{8.3}$$

where $P(F_j^i|t)$ is a full model of the probability of feature set i, i.e., assuming the feature set i consists of features f_j^n to f_j^m then $P(F_j^i|t) = P(f_j^n, f_j^{n+1}, ..., f_j^m|t)$, and α_i is a weighting factor.

The probabilities of our models can now be learned from our training image set for the respective query term t. We estimate nonparametric densities for all different metadata cues we chose to represent our images.

Having computed our models, we define the rank R of an image d_j to be inversely proportional to the probability of the current image d_j given the model for the given term t:

$$R \propto 1/P(d_j|t) \tag{8.4}$$

Note that we only need to compute the rank for a prefiltered set of images containing the tag and that the models can be calculated in an unsupervised manner. Moreover, they can be computed off-line and stored, such that at query time they only have to be applied to the current data.

In our experimental evaluation we consider two different modalities, visual features and tag features. Both types of features are based on feed-forward networks, trained on large sets of real-world images from Flickr. Such bottom-up systems compute features efficiently which is important as complicated features do not scale to web-size datasets. Note however that any other metadata type or feature could be used instead of or in addition in our model.

We will now describe the features we compute to represent our images, i.e., visual features from pixel values and tag features from the images' associated tags.

8.1.1. Visual Features

To learn locally shift-invariant, sparse representations we combine two recently proposed algorithms [78, 49]. Instead we could as well use the visual features based on topics models as described in Chapter 3, or high-level features derived from a deep network (see Chapter 6). However, as we consider web-size datasets in our experiments, efficient feature computation becomes important and the feed-forward structure of the in the following presented features computes the visual image representation faster than the iterative procedure used for inference in topic models.

At the core of the model for feature computation is a sparse coding algorithm which is extended to learn locally shift-invariant representations, i.e., they are invariant to small distortions. By stacking the resulting model, i.e., applying the algorithm multiple times, each time to the outcome of the previous model, a feature hierarchy consisting of multiple layers is produced. Details of the visual feature computation are given in the following.

We start with a feed-forward approximation [49] of Olshausen and Field's [72] sparse coding model. Given a vectorized input image patch $I \in R^M$ Olshausen and Field's model aims to find the code $Z \in R^N$ that reconstructs the input and is sparse. Therefore the following cost function is minimized:

$$L(I,Z;W_d) = ||I - W_d Z||^2 + \lambda \sum_k |Z_k| \tag{8.5}$$

Here W_d is a $M \times N$ a matrix that is learned and $\lambda \in R^+$ controls the sparsity of the representation. If N is chosen much smaller than M we achieve a dimensionality reduction.

Training and especially inference with such a model is very inefficient. Therefore we train a non-linear regressor to map input patches directly to sparse codes:

$$F(I;D,W_e) = D\tanh(W_e I) \tag{8.6}$$

Here D is a diagonal matrix of coefficients, and W_e is a $N \times M$ filter matrix. The aim of the

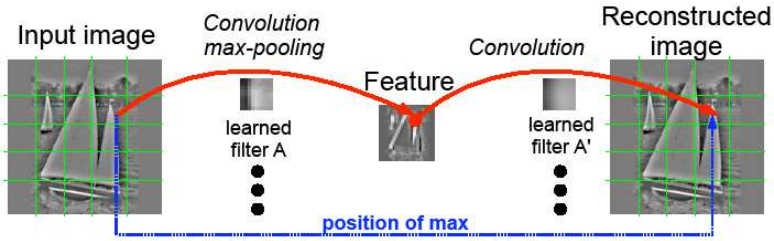


Figure 8.2: One stage of the image-coding/decoding pipeline. The feature we use to represent the image is shown in the middle of the pipeline. The transformation parameters that represent the location of the maximum in each window, are passed from coder to decoder because they represent spatial data we want the representation to ignore [44].

algorithm is now to make the prediction F as similar as possible to the optimal code Z. To ensure this, a term is added to Equation 8.5, resulting in the following objective function which is minimized during training:

$$L(I,Z;W_d,D,W_e) = ||I - W_d Z||^2 + \lambda \sum_k |Z_k| + ||Z - D\tanh(W_e I)||^2 \tag{8.7}$$

The training algorithm alternates between a minimization over Z and a parameter update step over (W_d, W_e, D). Note that the rows of the matrix W_e can be interpreted as trainable filters that are applied to the input.

In order to make the code translation invariant over small spatial neighborhoods we use the above-mentioned filters convolutionally over the input image patch (which is not vectorized and whose spatial resolution is larger than the support of the filters) [78]. Then we take the maximum across non-overlapping windows as our feature value, resulting into invariance with respect to translations within the corresponding window. The reconstruction is done convolutionally as well by first placing the code units in the feature maps at the locations where the maxima where determined. Next the resulting feature maps are convolved with the reconstruction filters, and finally summed up to produce the reconstruction of the input. Figure 8.2 shows the coder and decoder.

The learning algorithm is not affected by adding the spatial invariance.

As the algorithm does not make any assumptions regarding the input, such models can be stacked to produce a feature hierarchy, analogous to the deep network training scheme described in Chapter 6 and by Hinton in [39]. Therefore, the algorithm is first trained using image patches. Once we have learned the filter banks, we use the feed-forward mapping function to directly predict sparse and locally shift-invariant codes that are subsequently used to train another layer. We repeat the same greedy process for multiple layers. This results in sparse, locally shift-invariant features that are produced by a simple feed-forward pass through a few stages of convolution and max-pooling.

8.1.2. Tag Features

To transform tags associated with an image to high-level image features, we use a bag-of-words description in combination with a deep network similar to the one described in the previous chapter. The deep network we apply uses multiple, non-linear hidden layers, it was introduced in [39] and [83]. This deep network computes, similar to the model for pixels from the previous subsection, a low-dimensional representation from which the tags associated with an image can be reconstructed with a small error.

The learning procedure for such a deep model consisting of multiple hidden feature layers proceeds in two stages. In the first stage, the pretraining, we compute an initialization based on restricted Boltzmann machines (RBM) by using one-step contrastive divergence [38]. The second stage refines the representation by using backpropagation to optimally reconstruct the input data. We have described the model and its training in detail in Chapter 6.

The input vector from tags to such a deep network is a word count vector. We first divide each entry of the respective vector by the total number of tags associated with the current image. This creates a discrete probability distribution over the finite tag vocabulary for each image. To model the probability distributions in the input layer, we use a softmax at the visible units in the first-level RBM while its hidden units and also all other units in the deep network are binary. However the output units at the top level of the network are linear. We use the multi-class cross-entropy error function to refine the weights and biases in the backpropagation algorithm.

Once the deep network is trained we derive a low-dimensional representation for each of our images in the semantic space by applying the learned model to each image in the database and using its top-level unit values as its low-dimensional description. It should be noted that the mapping from the word count vector, i.e., the basic tag description, to a high level semantic feature only consists of a single matrix multiplication and single squashing function per network unit.

8.1.3. Density Estimation

Having computed the features for all of the images of a tag category, i.e., for a specific query term t, we perform non-parametric density estimation in order derive the probabilities $P(f_j^i|t)$. We compute a one-dimensional probability density for each of the features' dimensions, assuming those are independent. Denoting the set of training samples for one feature dimension by $\mathbf{x} = \{x_1, ...x_D\}$, the density estimator employs an isotropic kernel $K_h(x,x')$ with a fixed width h, and the density is computed as:

$$P(x) = \frac{1}{D}\sum_{i=1}^{D} K_h(x, x_i) \tag{8.8}$$

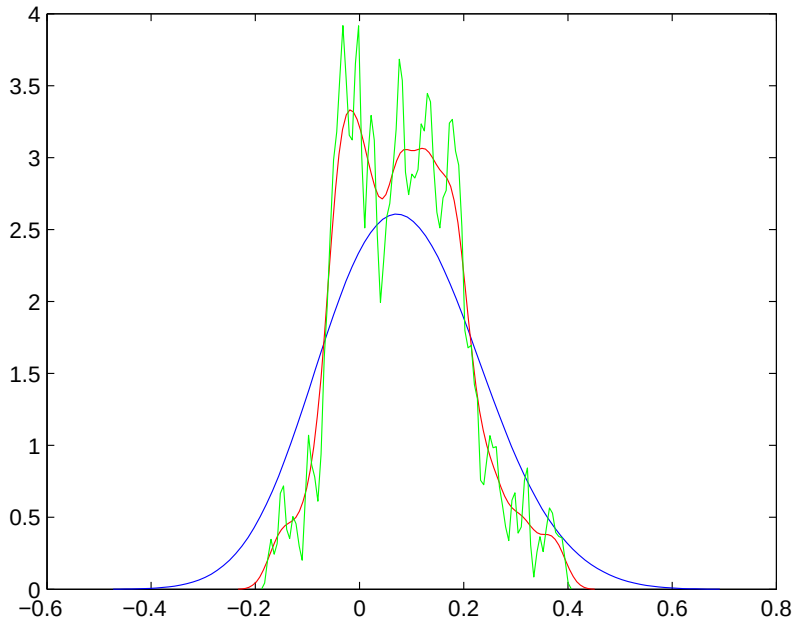


Figure 8.3: Three examples of density estimation, one too smooth (blue), one too detailed (green), and one that maximizes the likelihood of the test data (red).

We use a Gaussian kernel:

$$K_h(x,x') = \frac{1}{(2\pi h^2)^{\frac{1}{2}}} \exp\left(-\frac{(x-x')^2}{2h^2}\right) \tag{8.9}$$

To pick the appropriate kernel width we perform 10-fold cross validation. As the goal of the density estimation is to build a model of the data that accurately reflects the underlying probability distribution, we determine through cross validation, the kernel variance that gives us a model that best predicts, i.e., that gives the highest probability, for held-out test data. An example for the necessity of the kernel width cross validation is shown in Figure 8.3.

It should be noted that the probability densities are often bimodal or skewed. The product of these distributions then forms our simple model for the image likelihood as a function of the image features.

8.2. Implementation

We consider the following 20 tag categories: *baby*, *beetle*, *butterfly*, *carnival*, *chair*, *christmas*, *cn tower*, *coast*, *colosseum*, *flower*, *forest*, *golden gate bridge*, *highway*, *horse*, *mountain*, *sailboat*, *sheep*, *statue of liberty*, *sunset*, *wedding*. Note that this list includes objects, landmarks, scenes, events and places.

We downloaded nearly 4.8 million public, geotagged Flickr images, all with one of the tags listed above. The number of images per category ranged from 3,700 to 683,000.

The weighting factor α_i is set to 1 in our experiments.

8.2.1. Visual Feature Implementation

In our experiments we use three layers of convolution and max-pooling to derive the final feature representation. Each image is preprocessed by converting it to YUV, down-sampling so that the longest side is 158 pixels and high-pass filtering the Y-channel to remove changes in illumination. Finally we zero-pad all image patches to size 140×140.

The filter banks of the first two stages have kernels of size 9×9, and the max-pooling is performed over 4×4 and 5×5 non-overlapping windows at the first and second stage, respectively. At the third stage, no pooling is performed as the filters have support of size equal to the one of the second stage feature maps. The number of feature maps is equal to 128 at the first stage, 512 at the second stage and 1024 at the third stage. Thus our final visual image representation consists of 1024 dimensions.

8.2.2. Tag Feature Implementation

As a training set to build our tag feature we sample a number of images from all categories and use this set to train a deep network that can be used to compute features for each image, no matter which tag category it belongs to.

The first step here is to choose a vocabulary. This is done by listing all tags associated with at least one training image. Next we filter this list to keep only those tags that are used by a certain number of authors/owners – ten in our implementation. Additionally we remove all tags containing numbers. To map singular and plural forms of objects into one single word we use a very simple scheme which pools words that are equal up to the letter ‘s’ at the end of the word. Note that a more sophisticated approach to building the vocabulary may be useful to consider for future work, as we also do not consider any type of translation of the tags.

Our final vocabulary consists of 3674 words. Having chosen the vocabulary we can represent each image by a word-count vector. The resulting representation is in most cases very sparse as users typically spend up to 10 words/expressions to tag an image.

The trained deep networks consists of three hidden layers with a 3674-1000-400-100 architecture. Thus, we obtain a 100-dimensional semantic representation for each image. We used 84,000 images from all categories for training the deep network, 25 iterations for pretraining each layer, and 50 iterations to optimize the autoencoder.

8.2.3. Densities

We randomly select 10,000 images per tag category to estimate the probability distributions. If less than 10,000 image were available we used all of them. Moreover we only consider one (randomly chosen) image per owner and tag as we do not want the densities biased towards the images of one photographer. This can easily happen as there are photographers who upload thousands of images all associated with one tag and all very similar but not representative of the tag.

In order to store the estimated non-parametric densities we evaluate them at 5000 equally distributed points between the minimum and maximum feature value, and keep the found values. Normalizing their sum to 1 gives us discrete probabilities for each of those points. Thus, when performing ranking we map each feature to its nearest value in the current dimension, i.e., performing a kind of quantization, and we directly derive the probability associated. This ensures fast computation of image ranks while at the same time storage consumption is low.

We also implemented a cleaning algorithm to reduce the effects of common (background) images in our density calculation. This process removes images that are not surrounded by other images of the same class. For each class we perform a nearest-neighbor calculation using 10,000 in-class and 10,000 background images. We keep an image if more than 50% of its neighbors are in the same class. Note that the number of training images in some categories is less than 10,000 even before filtering, thus making it harder in those cases to reach the required 50%.

8.2.4. Diversity

When applying our approach the aim is to obtain a set of highly relevant images corresponding to the current query term. However, at the same time we want our result set to show diverse images. More specifically the retrieved images should be visually diverse when showing one object or scene, for instance, different prominent views of a landmark under different lighting or the various appearances of an object with different backgrounds or in different situations. The results should also be diverse with respect to their meaning, i.e., if we ask for *beetle* images our result set needs to contain images showing the animal as well as the car.

To enforce diversity we pick the top K result images, denoted by S_K, of a certain query term after ranking, then we wish to obtain a set of L diverse images out of those where $L < K$. We apply our diversity approach only to the images in S_K, i.e., to the top K results, to ensure that all images in the set are still highly relevant. The top K images may be chosen by using multiple features/modalities in the probability model, however, for diversification we only consider the visual representation, i.e., the visual features, of the images.

Our diversification approach is based on the assumption that visually similar images have feature vectors that are close in many dimensions. As we store the probability densities as discrete

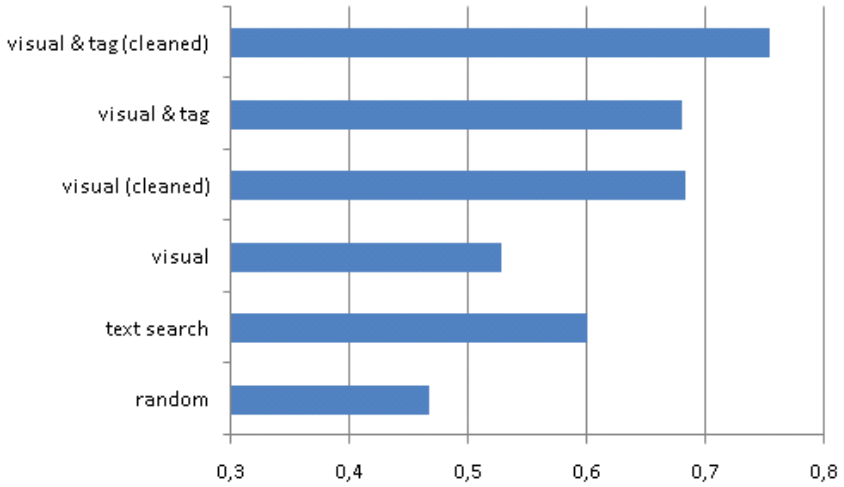


Figure 8.4: Mean scores for the top-15 images over all categories for our baseline method (random), a text-based search engine, and four different variants of the unsupervised ranking algorithm.

distributions, the feature values of similar images will be mapped to similar or at least very close values in these discrete densities. Thus, we propose an iterative approach to derive a visually diverse set of images that selects one result image in each round and subsequently reduces the mapped probability values corresponding to the selected image in the densities afterwards. This ensures that out of a pool of visually very similar images only one is picked.

Our proposed algorithm starts by selecting the most relevant image and setting the respective probability values in the discrete densities corresponding to its value in each feature dimension to a very small number, 0.000001 in our experiments. Note that we do not only set the probabilities corresponding exactly to the visual features of the image to a very small value, but we also include the neighboring values. The neighboring values are here defined as the p feature values smaller and larger in the discrete distribution. In our experiments we set p to 5, and we do not renormalize the densities after each round. Next we re-rank the remaining images in the set S_K according to their updated probabilities and pick the highest ranked image. The described process is iterated until we have selected L diverse result images.

It should be noted that there are other approaches for diversification that could be used, e.g., clustering the visual representation of the images, etc. Diversification approaches have also been examined in [97].

	Random	Text Search	Unsupervised
baby	0.52	0.23	**0.76**
beetle	0.60	**0.83**	**0.83**
butterfly	0.75	0.78	**0.88**
carnival	0.28	0.43	**0.81**
chair	0.41	0.39	**0.79**
christmas	0.21	0.26	**0.67**
cn tower	0.46	0.65	**0.98**
coast	0.27	0.40	**0.76**
colosseum	0.77	**0.99**	0.93
flower	0.77	**0.82**	0.71
forest	0.12	0.68	**0.79**
golden gate	0.56	**0.95**	0.64
highway	0.10	0.33	**0.37**
horse	0.29	**0.86**	0.72
mountain	0.27	0.61	**0.68**
sailboat	0.72	0.59	**0.77**
sheep	0.51	0.62	**0.79**
statue of liberty	0.42	0.56	0.**99**
sunset	**0.76**	0.63	0.62
wedding	**0.58**	0.41	**0.58**

Table 8.1: Detailed relevance scores for the different categories. The winning score for each category is shown in bold.

8.3. Results

To test our proposed ranking approach for images, we compute the described likelihood models for each of our 20 categories. We then calculate the probabilities for all of our 4.8 million images using models built from visual features only as well as models built using both image content and text data, i.e., pixel and tag features. We compare the ranking performance of those models, with and without training data cleaning (see Section 8.2.3), to two different baseline measures: a random set of images from the respective category, and images that are ranked highly by the Flickr search API (which uses text matching).

We evaluate the performance of our approach in a user study by displaying the top 15 images for each of the 20 test categories. Note that we only display one image per author in our results. Then we ask six test users to judge the result images by assigning an image that they consider relevant to the query 1 point, each irrelevant image 0 points and each image that they found somewhat relevant 0.5 points. We report the mean score per image and tag over all test users and result images for each model.

Figure 8.4 compares the mean scores over all queries of the different approaches. Using visual features alone gives a better performance than a random set of images. Adding tags to the model

Figure 8.5: Sample results for the *chair* query for visual and tag model trained using cleaned data.

Figure 8.6: Sample results for the *carnival* query for visual and tag model trained using cleaned data.

improves the relevance scores, as does cleaning the models' training data sets to remove images that were common to many categories. Summarizing our approach improves performance compared to both baselines.

Table 8.1 summarizes the results as a function of the query term. Images of objects with a low visual diversity, such as *colosseum* and *statue of liberty*, are easy. Categories such as *wedding* or *highway* are more difficult because of the wide range of images. Many *wedding* pictures, for example, consist of groups of happy people. They were probably taken at weddings, but it is hard for humans to judge their relevance. Moreover the quality of images in some categories such as *sunset*, *flower* or *butterfly* is good even for the random approach indicating that a majority of the images tagged with this terms are showing the content mentioned. Other categories, e.g., *christmas* or *forest*, are much harder because they are less distinct; nevertheless our approach still shows good performance.

Three example results are depicted in Figures 8.5 to 8.7. We show the resulting top-16 images for the categories *chair*, *carnival* and *sheep*. We observe that even for object categories con-

Figure 8.7: Sample results for the *sheep* query for visual and tag model trained using cleaned data.

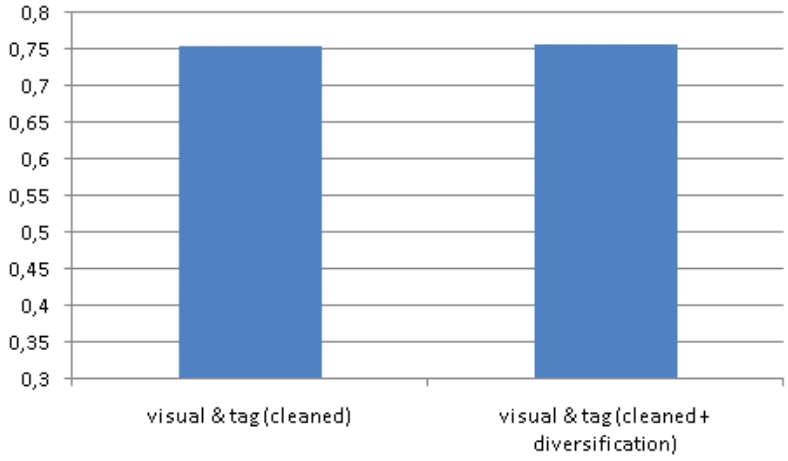


Figure 8.8: Mean scores for the top-15 images over all categories for the unsupervised ranking algorithm using the visual and tag model on cleaned data without and with diversification of the result images.

taining many complicated images such as the *sheep* or *chair* category our approach finds highly relevant images, all of them showing sheep or chairs, respectively.

In our next experiment we compare the mean score per image and tag for the best-performing model, which uses visual and tag features and is trained on cleaned data, before and after applying our diversification algorithm in Figure 8.8. As can be seen the diversification does not reduce the relevance of the search results.

Diversification is especially necessary when looking for landmark images, as here many photographers shoot visually very similar images from the same perspective. We show the top 16 result images before and after diversification for the query terms *cn tower* and *statue of liberty* in Figures 8.9 and 8.10. As one can see, the proposed algorithm produces a more diverse result set. Another example is shown in Figure 8.11 for the category *beetle*. Here our unsupervised

Figure 8.9: Sample results for the *cn tower* query for visual and tag model trained using cleaned data without and with diversification.

Figure 8.10: Sample results for the *statue of liberty* query for visual and tag model trained using cleaned data without and with diversification.

approach combined with the diversification successfully discovers both meanings of the term; we obtain images of insects as well as cars.

8.4. Summary

In this chapter we have presented an approach to determine image relevance and thus to find images in a large web-scale collection that are very representative to a given query term. Our approach ranks the images highest whose content and their various metadata types give the highest probabilities according to a model we learn for this tag. Our model is learned without supervision.

Figure 8.11: Sample results for the *beetle* query for visual and tag model trained using cleaned data without and with diversification.

We do not only want a set of highly relevant result images, but at the same time we want our result set to show visually diverse images. Thus we propose a diversification approach that is based on the assumption that visually similar images have feature vectors that are close in many dimensions.

The experimental evaluation showed improved performance of the proposed approach over the baseline algorithms.

9. Conclusion

9.1. Summary

In this thesis we have studied the representation of images by topic models in the context of retrieval on large, real-world databases.

First, we have presented a complete system for query-by-example image retrieval which relies on a topic-model-based representation. In this system we rely solely on the image content. Three different basic topic models, the pLSA, LDA and CTM have been analyzed in detail. It has been shown that the pLSA and LDA models are more appropriate than the CTM to model the images in a retrieval-by-example scenario. Furthermore we proposed and evaluated different distances measure for similarity judgment, and it was found that a probabilistic measure combining a topic model and a unigram representation outperformed the other measures.

Further, we applied an active learning algorithm to our topic-model-based image description. Retrieval results were further improved by means of a novel preprocessing scheme that prunes the set of candidate images used during active learning.

Next, we evaluated different promising local image features, i.e., three different feature detectors and descriptors, in combination with the pLSA topic model to determine their suitability in scene recognition and image retrieval tasks. It was found that the dense grid over several scales detector performed best in both tasks. The geometric blur descriptor performed best in the scene recognition task, closely followed by the self-similarity descriptor whereas in a retrieval-by-example scenario on a large-scale database, the SIFT descriptor outperformed the self-similarity descriptor. We concluded that the appropriate choice of descriptor depends on the database used.

Further, we proposed and explored three topic models for fusing different types of local features. The models were evaluated experimentally for the case of fusing two different local features, a color patch feature and the SIFT descriptor, and it was shown that a model that performs late fusion, i.e., learns two independent topic models, one for each feature type, and fuses the representations during similarity measurement at retrieval time, performed best. We also examined the two local descriptors as well as their combination with respect to the suitability to model certain image categories. It was shown that retrieval results in some categories are improved by using more than one modality, whereas other categories are better modeled using only one feature type.

As topic models have been originally developed in the context of text collections, quantization of the extracted local image features is necessary to derive a discrete visual vocabulary. We proposed three extensions to the pLSA which model the visual vocabulary continuously and thus make this quantization step obsolete. In a scene classification task all three models have shown superior performance compared to the original pLSA model. We also evaluated one of the models, the FSGW-pLSA, in a query-by-example scenario and showed that it improves the retrieval results over the discrete pLSA model.

In this thesis we also exploited deep network models for deriving a low-dimensional description of the image content. Due to their feed-forward structure those networks are, once their parameters have been learned, fast to apply and additionally they offer a multi-level hierarchical image content description. Our experiments showed that their retrieval performance is comparable to the performance of various topic models.

In many public repositories, the images are associated with different types of metadata such as tags, date or camera parameters. Thus retrieval results would possibly be enhanced by integrating other modalities besides the image content into our models. In this thesis we presented work in progress on fusing multiple modalities for image retrieval. We explored three fusion models of which two are hierarchical. One of the proposed hierarchical models is based on the pLSA, and the other one is based on deep networks. In our experiments we fused visual features and semantic features based on tags and evaluated both models.

Finally, we proposed an approach to find the most relevant images, i.e., very representative images, in a large web-scale collection given a query term. Our approach ranks the images highest whose content and their various metadata types give the highest probabilities according to a model we learn for this tag. Our model is trained without supervision and the experimental evaluations showed improved performance of the proposed approach over the baseline algorithms.

9.2. Future Work

We can think of several directions for future research on topic-model-based image representations in the context of large scale image retrieval. Also, there are several extensions to the system analyzed in this work that could further improve retrieval results. Some of those are:

- In Chapter 6 we showed that an image representation based on deep networks achieves a comparable performance to topic models. Moreover deep networks provide a multi-level representation of the image content. This hierarchical representation could be explored in future work for improving image retrieval.
- The presented approaches on fusing multiple modalities in Chapter 7 are work in progress. Thus, in future, these models need to be analyzed in more detail. We experimented with

fusing visual content and tags, however images in the Flickr database are associated with many other, promising and potentially useful, metadata types. It has to be investigated which of those are appropriate to improve retrieval and how they can be integrated into the existing models.

- The proposed and evaluated image content models in this work ignore the geometric relationships among the visual words in images as they start from a bag-of-visual-words image description. An improvement in retrieval performance is expected for future models that account for such spatial informations.
- As large-scale public image repositories grow daily, two important aspects need to be addressed when developing models in future research. First, the novel approaches need to be designed for their use in very large databases. They need to take into account such factors as the scalability of training and inference algorithms, as well as computational and memory resources. Second, it will be necessary to design on-line algorithms to continuously (re-)learn the parameters of the proposed image models, as the content of real-world databases is permanently modified by the constant upload or deletion of images.
- In our experimental evaluations we use databases consisting mainly of object, scene and landmark categories. On such databases the proposed approaches have shown to perform well. However, it is not clear how the performance of our system is affected when searching for abstract themes such as *love*, *anger*, *success*, etc. In future works this should be examined in more detail.

9.3. Related Publications

Parts of the work presented in this thesis have been published in the following papers:

- E. Hörster, M. Slaney, M.A. Ranzato and K. Weinberger. Unsupervised Image Ranking. *Workshop on Large-Scale Multimedia Retrieval and Mining*, 2009, to appear.
- R. Lienhart, S. Romberg and E. Hörster. Multilayer pLSA for Multimodal Image Retrieval. In *ACM International Conference on Image and Video Retrieval (CIVR)*, 2009. To appear.
- S. Romberg, E. Hörster and R. Lienhart. Multimodal Plsa on Visual Features and Tags. In *IEEE International Conference on Multimedia and Expo (ICME)*, 2009. To appear.
- E. Hörster and R. Lienhart. Deep Networks for Image Retrieval on Large-Scale Databases. In *ACM International Conference on Multimedia (ACMMM)*, pp. 643-646, 2008.
- E. Hörster, R. Lienhart and M. Slaney. Continuous Visual Vocabulary Models for pLSA-Based Scene Recognition. In *ACM International Conference on Image and Video Retrieval (CIVR)*, pp. 319-328, 2008.

- E. Hörster, T. Greif, R. Lienhart and M. Slaney. Comparing Local Feature Descriptors in pLSA-Based Image Models. In *30th Annual Symposium of the German Association for Pattern Recognition (DAGM)*, 2008.
- E. Hörster, R. Lienhart and M. Slaney. Image Retrieval on Large-Scale Image Databases. In *ACM International Conference on Image and Video Retrieval (CIVR)*, pp. 17-24, 2007.
- E. Hörster and R. Lienhart. Fusing Local Image Descriptors for Large-Scale Image Retrieval. In *IEEE Computer Vision and Pattern Recognition (International Workshop on Semantic Learning Applications in Multimedia)*, 2007.

A. Test Images

Figure A.1: Test images used for the experimental evaluation in Chapter 3, 4, 5 and 6.

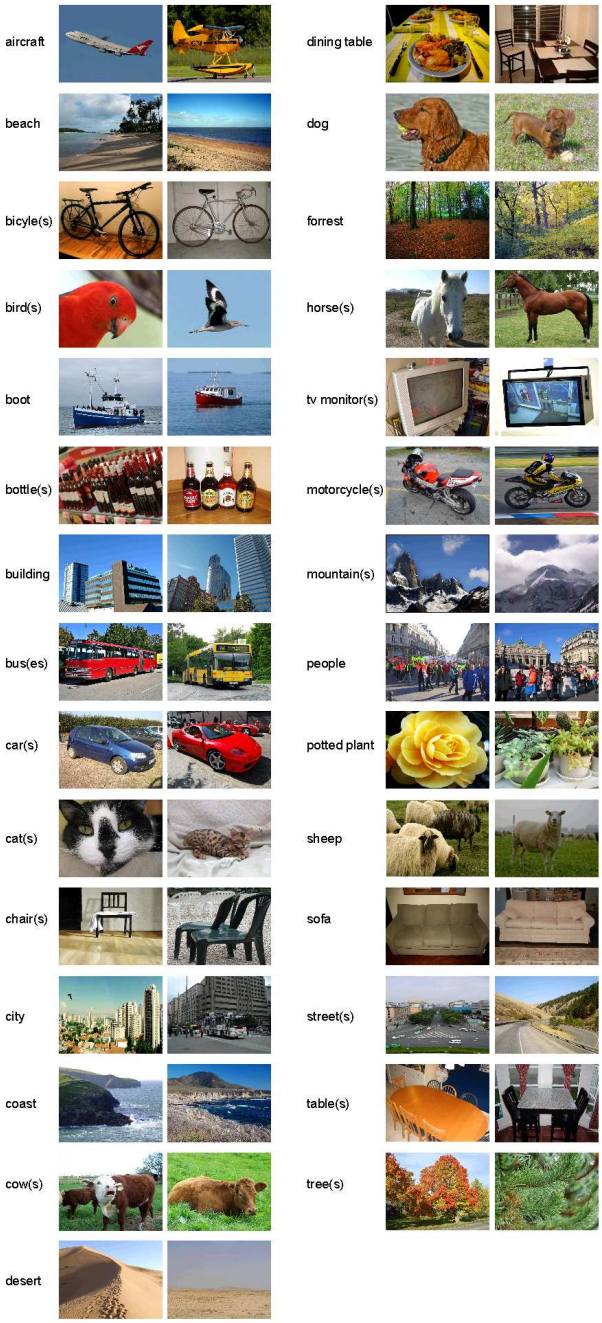


Figure A.2: Test images used for the experimental evaluation in Chapter 7.

B. Derivation of the EM Algorithm for the Continuous Vocabulary Models

B.1. pLSA with Shared Gaussian Words (SGW-pLSA)

The probability of a feature descriptor f_j in image d_i according to the SGW-pLSA is:

$$P(f_j, d_i) = \sum_{h=1}^{H} \sum_{k=1}^{K} P(z_j = h|d_i) \cdot P(g_j = k|z_j = h) \cdot P(f_j|g_j = k) \cdot P(d_i) \tag{B.1}$$

where

$$P(f_j|g_j = k) = N(f_j|\mu_k, \Sigma_k) \tag{B.2}$$

H denotes the number of topics, K is the number of shared Gaussians in the model and N denotes the Normal distribution.
The likelihood of the database (given all observations are independent) is then given by:

$$L = \prod_{i=1}^{M} \prod_{j=1}^{N_i} P(f_j, d_i) \tag{B.3}$$

N_i denotes the number of visual features detected in image d_i and M is the number of images in the database.
For the log-likelihood of the database we derive:

$$l = \sum_{i=1}^{M} \sum_{j=1}^{N_i} \log \left[\sum_{h=1}^{H} \sum_{k=1}^{K} [P(z_j = h|d_i) \cdot P(g_j = k|z_j = h) \cdot P(d_i) \cdot N(f_j|\mu_k, \Sigma_k)] \right] \tag{B.4}$$

Due to the sum inside the logarithm direct maximization is difficult. Thus we use the iterative Expectation Maximization (EM) algorithm.
We introduce following indicator variables:

$$\Delta c_{kh} = \begin{cases} 1 & \text{if the pair } (d_i, f_j) \text{ was generated by the } h\text{-th concept and Gaussian mixture } k \\ 0 & \text{otherwise} \end{cases} \tag{B.5}$$

for which we assume that they can be observed.
The data likelihood for the completely observable data is called the complete data likelihood L^{comp} and is given by:

$$L^{comp} = \prod_{i=1}^{M} \prod_{j=1}^{N_i} P(f_j, d_i, \Delta c) \tag{B.6}$$

with:

$$\Delta c = (\Delta c_{11}, ..., \Delta c_{1H}, ... \Delta c_{KH}) \tag{B.7}$$

$$P(f_j, d_i, \Delta c) = \prod_{h=1}^{H} \prod_{k=1}^{K} [P(z_j = h|d_i) \cdot P(g_j = k|z_j = h) \cdot P(d_i) \cdot N(f_j|\mu_k, \Sigma_k)]^{\Delta c_{kh}} \tag{B.8}$$

For the complete log data likelihood we obtain:

$$l_{comp} = \sum_{i=1}^{M} \sum_{j=1}^{N_i} \sum_{h=1}^{H} \sum_{k=1}^{K} \Delta c_{kh} \log [P(z_j = h|d_i) \cdot P(g_j = k|z_j = h) \cdot P(d_i) \cdot N(f_j|\mu_k, \Sigma_k)] \tag{B.9}$$

Denoting the h-th topic by z^h, i.e., z=h, and the k-th Gaussian component by g^k, the expectation of the indicator variables is derived as follows:

$$P(\Delta c|f_j, d_i) = \frac{P(f_j, d_i, \Delta c)}{P(f_j, d_i)} \tag{B.10}$$

$$= \frac{\prod_{h=1}^{H} \prod_{k=1}^{K} \left[P(z^h|d_i) \cdot P(d_i) \cdot P(g^k|z^h) \cdot N(f_j|\mu_k, \Sigma_k)\right]^{\Delta c_{kh}}}{\sum_{h=1}^{H} \sum_{k=1}^{K} P(z^h|d_i) \cdot P(d_i) \cdot P(g^k|z^h) \cdot N(f_j|\mu_k, \Sigma_k)} \tag{B.11}$$

$$E(\Delta c_{kh}|f_j, d_i) = P(\Delta c_{kh} = 1|f_j, d_i) \cdot 1 + P(\Delta c_{kh} = 0|f_j, d_i) \cdot 0 \tag{B.12}$$

$$= P(\Delta c_{kh} = 1|f_j, d_i) \tag{B.13}$$

$$= \frac{P(z^h|d_i) \cdot P(d_i) \cdot P(g^k|z^h) \cdot N(f_j|\mu_k, \Sigma_k)}{\sum_{h=1}^{H} \sum_{k=1}^{K} P(z^h|d_i) \cdot P(d_i) \cdot P(g^k|z^h) \cdot N(f_j|\mu_k, \Sigma_k)} \tag{B.14}$$

$$= \frac{P(z^h|d_i) \cdot P(g^k|z^h) \cdot N(f_j|\mu_k, \Sigma_k)}{\sum_{h=1}^{H} \sum_{k=1}^{K} P(z^h|d_i) \cdot P(g^k|z^h) \cdot N(f_j|\mu_k, \Sigma_k)} \tag{B.15}$$

$$= \beta_{kh}^{ij} \tag{B.16}$$

The calculation of β_{kh}^{ij} is the E-step of our EM algorithm.
To derive the M-step equations we need to maximize $E(l^{comp})$:

$$E(l^{comp}) = \sum_{i=1}^{M} \sum_{j=1}^{N_i} \sum_{h=1}^{H} \sum_{k=1}^{K} \beta_{kh}^{ij} \log [P(z_j = h|d_i) \cdot P(g_j = k|z_j = h) \cdot P(d_i) \cdot N(f_j|\mu_k, \Sigma_k)] \tag{B.17}$$

$$= \sum_{i=1}^{M} \sum_{j=1}^{N_i} \sum_{h=1}^{H} \sum_{k=1}^{K} \beta_{kh}^{ij} \log \left[P(z^h|d_i) \cdot P(g^k|z^h) \cdot P(d_i) \cdot N(f_j|\mu_k, \Sigma_k)\right] \tag{B.18}$$

$$\tag{B.19}$$

with respect to the constraints:

$$\sum_{i=1}^{M} P(d_i) = 1 \tag{B.20}$$

$$\sum_{i=1}^{H} P(z^h|d_i) = 1 \quad i = 1,...,M \tag{B.21}$$

$$\sum_{k=1}^{K} P(g^k|z^h) = 1 \quad h = 1,...,H \tag{B.22}$$

We introduce the Lagrange multiplier $\lambda, \tau_i, \alpha_h$:

$$\begin{aligned} F = &\sum_{i=1}^{M}\sum_{j=1}^{N_i}\sum_{h=1}^{H}\sum_{k=1}^{K} \beta_{kh}^{ij} \log\left[P(z^h|d_i)\cdot P(d_i)\cdot P(g^k|z^h)\cdot N(f_j|\mu_k,\Sigma_k)\right] \\ &+ \lambda(1-\sum_{i=1}^{M} P(d_i)) + \sum_{i=1}^{M}\tau_i(1-\sum_{h=1}^{H} P(z^h|d_i)) + \sum_{h=1}^{H}\alpha_h(1-\sum_{k=1}^{K} P(g^k|z^h)) \end{aligned} \tag{B.23}$$

Next we maximize F with respect to the following parameters:

$$\theta_{z^h|d_i} = P(z^h|d_i) \tag{B.24}$$

$$\theta_{d_i} = P(d_i) \tag{B.25}$$

$$\theta_{g^k|z^h} = P(g^k|z^h) \tag{B.26}$$

$$\theta_{\mu_k} = \mu_k \tag{B.27}$$

$$\theta_{\Sigma_k} = \Sigma_k \tag{B.28}$$

With those parameters we obtain:

$$\begin{aligned} F = &\sum_{i=1}^{M}\sum_{j=1}^{N_i}\sum_{h=1}^{H}\sum_{k=1}^{K} \beta_{kh}^{ij} \log\left[\theta_{z^h|d_i}\cdot\theta_{d_i}\cdot\theta_{g^k|z^h}\cdot N(f_j|\theta_{\mu_k},\theta_{\Sigma_k})\right] \\ &+ \lambda(1-\sum_{i=1}^{M}\theta_{d_i}) + \sum_{i=1}^{M}\tau_i(1-\sum_{h=1}^{H}\theta_{z^h|d_i}) + \sum_{h=1}^{H}\alpha_h(1-\sum_{k=1}^{K}\theta_{g^k|z^h}) \end{aligned} \tag{B.29}$$

Setting the partial derivatives to zero gives:

$$\mu_k^{new} = \frac{1}{p_k}\sum_{i=1}^{M}\sum_{j=1}^{N_i}\sum_{h=1}^{H}\beta_{kh}^{ij}\cdot f_j \quad h = 1,...,H;\ \ k = 1,...,K \tag{B.30}$$

$$\Sigma_k^{new} = \frac{1}{p_k}\sum_{i=1}^{M}\sum_{j=1}^{N_i}\sum_{h=1}^{H}\beta_{kh}^{ij}\cdot(f_j-\mu_k^{new})(f_j-\mu_k^{new})^T \tag{B.31}$$

$$= \left(\frac{1}{p_k}\sum_{i=1}^{M}\sum_{j=1}^{N_i}\sum_{h=1}^{H}\beta_{kh}^{ij}\cdot f_j^2\right) - (\mu_k^{new})^2 \quad h = 1,...,H;\ \ k = 1,...,K \tag{B.32}$$

where

$$p_k = \sum_{i=1}^{M} \sum_{j=1}^{N_i} \sum_{h=1}^{H} \beta_{kh}^{ij} \tag{B.33}$$

For the other parameters we obtain:

$$\frac{\partial F}{\partial \theta_{z^h|d_i}} = \frac{\sum_{j=1}^{N_i} \sum_{k=1}^{K} \beta_{kh}^{ij}}{\partial \theta_{z^h|d_i}} - \tau_i = 0 \quad h = 1,...,H; \;\; i = 1,...,M \tag{B.34}$$

$$\frac{\partial F}{\partial \theta_{d_i}} = \frac{\sum_{h=1}^{H} \sum_{j=1}^{N_i} \sum_{k=1}^{K} \beta_{kh}^{ij}}{\partial \theta_{d_i}} - \lambda = 0 \quad i = 1,...,M; \tag{B.35}$$

$$\frac{\partial F}{\partial \theta_{g^k|z^h}} = \frac{\sum_{i=1}^{M} \sum_{j=1}^{N_i} \beta_{kh}^{ij}}{\partial \theta_{g^k|z^h}} - \alpha_h = 0 \quad h = 1,...,H; \;\; k = 1,...,K \tag{B.36}$$

Solving for the desired parameters gives:

$$\theta_{z^h|d_i} = P(z^h|d_i) = \frac{\sum_{j=1}^{N_i} \sum_{k=1}^{K} \beta_{kh}^{ij}}{\tau_i} \quad h = 1,...,H; \;\; i = 1,...,M \tag{B.37}$$

$$\theta_{d_i} = P(d_i) = \frac{\sum_{h=1}^{H} \sum_{j=1}^{N_i} \sum_{k=1}^{K} \beta_{kh}^{ij}}{\lambda} \quad i = 1,...,M; \tag{B.38}$$

$$\theta_{g^k|z^h} = P(g^k|z^h) = \frac{\sum_{i=1}^{M} \sum_{j=1}^{N_i} \beta_{kh}^{ij}}{\alpha_h} \quad h = 1,...,H; \;\; k = 1,...,K \tag{B.39}$$

Taking the normalization constraints into account, we can solve for the Lagrange multiplier:

$$\tau_i = \sum_{h=1}^{H} \sum_{j=1}^{N_i} \sum_{k=1}^{K} \beta_{kh}^{ij} \tag{B.40}$$

$$\lambda = \sum_{h=1}^{H} \sum_{i=1}^{M} \sum_{j=1}^{N_i} \sum_{k=1}^{K} \beta_{kh}^{ij} \tag{B.41}$$

$$\alpha_h = \sum_{k=1}^{K} \sum_{i=1}^{M} \sum_{j=1}^{N_i} \beta_{kh}^{ij} \tag{B.42}$$

The remaining M-step equations are:

$$P(z^h|d_i)^{new} = \frac{\sum_{j=1}^{N_i} \sum_{k=1}^{K} \beta_{kh}^{ij}}{\sum_{h=1}^{H} \sum_{j=1}^{N_i} \sum_{k=1}^{K} \beta_{kh}^{ij}} \quad h = 1,...,H; \;\; i = 1,...,M \tag{B.43}$$

$$= \frac{\sum_{j=1}^{N_i} \sum_{k=1}^{K} \beta_{kh}^{ij}}{N_i} \quad h = 1,...,H; \;\; i = 1,...,M \tag{B.44}$$

$$P(d_i)^{new} = \frac{\sum_{h=1}^{H} \sum_{j=1}^{N_i} \sum_{k=1}^{K} \beta_{kh}^{ij}}{\sum_{h=1}^{H} \sum_{i=1}^{M} \sum_{j=1}^{N_i} \sum_{k=1}^{K} \beta_{kh}^{ij}} \quad i = 1,...,M; \tag{B.45}$$

$$= \frac{N_i}{\sum_i N_i} \quad i = 1,...,M; \tag{B.46}$$

$$P(g^k|z^h)^{new} = \frac{\sum_{i=1}^{M}\sum_{j=1}^{N_i}\beta_{kh}^{ij}}{\sum_{k=1}^{K}\sum_{i=1}^{M}\sum_{j=1}^{N_i}\beta_{kh}^{ij}} \quad h=1,...,H;\ \ k=1,...,K \tag{B.47}$$

B.2. pLSA with Gaussian Mixtures (GM-pLSA)

The probability of a word f_j in image d_i according to the GM-pLSA is:

$$P(f_j,d_i) = \sum_{h=1}^{H} P(z_j=h|d_i)\cdot P(f_j|z_j=h)\cdot P(d_i) \tag{B.48}$$

where

$$P(f_j|z_j=h) = \sum_{k=1}^{K} P(g_j^{z_j=h}=k|z_j=h)\cdot N(f_j|\mu_{kh},\Sigma_{kh}) \tag{B.49}$$

H denotes the number of topics in the model, K is the number of Gaussians used to model one topic and N denotes the Normal distribution.
The likelihood of the database (given all observations are independent) is then given by:

$$L = \prod_{i=1}^{M}\prod_{j=1}^{N_i} P(f_j,d_i) \tag{B.50}$$

Here N_i denotes the number of visual features detected in image i and M is the number of images in the database.
For the log likelihood of the database we derive:

$$l = \sum_{i=1}^{M}\sum_{j=1}^{N_i}\log\left[\sum_{h=1}^{H} P(z_j=h|d_i)\cdot P(d_i)\cdot\sum_{k=1}^{K} P(g_j^{z_j=h}=k|z_j=h)\cdot N(f_j|\mu_{kh},\Sigma_{kh})\right] \tag{B.51}$$

$$= \sum_{i=1}^{M}\sum_{j=1}^{N_i}\log\left[\sum_{h=1}^{H}\sum_{k=1}^{K} P(z_j=h|d_i)\cdot P(d_i)\cdot P(g_j^{z_j=h}=k|z_j=h)\cdot N(f_j|\mu_{kh},\Sigma_{kh})\right] \tag{B.52}$$

Due to the sum inside the logarithm direct maximization is difficult. Thus we use the iterative Expectation Maximization (EM) algorithm.
We introduce the following indicator variables:

$$\Delta c_{kh} = \begin{cases} 1 & \text{if the pair } (f_j,d_i) \text{ was generated by the } h\text{-th concept and its } k\text{-th Gaussian mixture} \\ 0 & \text{otherwise} \end{cases} \tag{B.53}$$

for which we assume that they can be observed.
The data likelihood for the completely observable data so called the complete data likeli-

hood L^{comp} and is given by:

$$L^{comp} = \prod_{i=1}^{M}\prod_{j=1}^{N_i} P(f_j, d_i, \Delta c_{kh}) \tag{B.54}$$

where

$$\Delta c = (\Delta c_{11}, ..., \Delta c_{1H}, ...\Delta c_{KH}) \tag{B.55}$$

$$P(f_j, d_i, \Delta c) = \prod_{h=1}^{H}\prod_{k=1}^{K}\left[P(z_j = h|d_i)\cdot P(d_i)\cdot P(g_j^{z_j=h} = k|z_j = h)\cdot N(f_j|\mu_{kh}, \Sigma_{kh})\right]^{\Delta_{kh}} \tag{B.56}$$

For the complete log data likelihood we obtain:

$$l^{comp} = \sum_{i=1}^{M}\sum_{j=1}^{N_i}\sum_{h=1}^{H}\sum_{k=1}^{K}\Delta_{kh}\log\left[P(z_j = h|d_i)\cdot P(d_i)\cdot P(g_j^{z_j=h} = k|z_j = h)\cdot N(f_j|\mu_{kh}, \Sigma_{kh})\right] \tag{B.57}$$

Denoting the h-th topic by z^h, i.e., $z = h$, and introducing the notation π_{kh} for the probability of the k-th Gaussian component associated with the topic h, i.e., $\pi_{kh} = P(g^{z=h} = k|z = h)$, the expectation of the indicator variables is derived as follows:

$$P(\Delta c|f_j, d_i) = \frac{P(f_j, d_i, \Delta c)}{P(f_j, d_i)} \tag{B.58}$$

$$= \frac{\prod_{h=1}^{H}\prod_{k=1}^{K}\left[P(z^h|d_i)\cdot P(d_i)\cdot \pi_{kh}\cdot N(f_j|\mu_{kh}, \Sigma_{kh})\right]^{\Delta c_{kh}}}{\sum_{h=1}^{H}\sum_{k=1}^{K}P(z^h|d_i)\cdot P(d_i)\cdot \pi_{kh}\cdot N(f_j|\mu_{kh}, \Sigma_{kh})} \tag{B.59}$$

$$E(\Delta c_{kh}|f_j, d_i) = P(\Delta c_{kh} = 1|f_j, d_i)\cdot 1 + P(\Delta c_{kh} = 0|f_j, d_i)\cdot 0 \tag{B.60}$$

$$= P(\Delta c_{kh} = 1|f_j, d_i) \tag{B.61}$$

$$= \frac{P(z^h|d_i)\cdot P(d_i)\cdot \pi_{kh}\cdot N(f_j|\mu_{kh}, \Sigma_{kh})}{\sum_{h=1}^{H}\sum_{k=1}^{K}P(z^h|d_i)\cdot P(d_i)\cdot \pi_{kh}\cdot N(f_j|\mu_{kh}, \Sigma_{kh})} \tag{B.62}$$

$$= \frac{P(z^h|d_i)\cdot \pi_{kh}\cdot N(f_j|\mu_{kh}, \Sigma_{kh})}{\sum_{h=1}^{H}\sum_{k=1}^{K}P(z^h|d_i)\cdot \pi_{kh}\cdot N(f_j|\mu_{kh}, \Sigma_{kh})} \tag{B.63}$$

$$= \beta_{kh}^{ij} \tag{B.64}$$

The calculation of β_{kh}^{ij} is the E-step of our EM algorithm.
To derive the M-step equations we need to maximize $E(l^{comp})$:

$$E(l^{comp}) = \sum_{i=1}^{M}\sum_{j=1}^{N_i}\sum_{h=1}^{H}\sum_{k=1}^{K}\beta_{kh}^{ij}\log\left[P(z_j = h|d_i)\cdot P(d_i)\cdot P(g_j^{z_j=h} = k|z_j = h)\cdot N(f_j|\mu_{kh}, \Sigma_{kh})\right] \tag{B.65}$$

$$= \sum_{i=1}^{M}\sum_{j=1}^{N_i}\sum_{h=1}^{H}\sum_{k=1}^{K}\beta_{kh}^{ij}\log\left[P(z^h|d_i)\cdot P(d_i)\cdot \pi_{kh}\cdot N(f_j|\mu_{kh}, \Sigma_{kh})\right] \tag{B.66}$$

with respect to the constraints:

$$\sum_{i=1}^{M} P(d_i) = 1 \tag{B.67}$$

$$\sum_{i=1}^{H} P(z^h|d_i) = 1 \quad i = 1,...,M \tag{B.68}$$

$$\sum_{k=1}^{K} \pi_{kh} = 1 \quad h = 1,...,H \tag{B.69}$$

We introduce the Lagrange multiplier $\lambda, \tau_i, \alpha_h$:

$$\begin{aligned} F = &\sum_{i=1}^{M}\sum_{j=1}^{N_i}\sum_{h=1}^{H}\sum_{k=1}^{K} \beta_{kh}^{ij} \log\left[P(z^h|d_i)\cdot P(d_i)\cdot \pi_{kh}\cdot N(f_j|\mu_{kh},\Sigma_{kh})\right] \\ &+ \lambda(1-\sum_{i=1}^{M} P(d_i)) + \sum_{i=1}^{M}\tau_i(1-\sum_{h=1}^{H} P(z^h|d_i)) + \sum_{h=1}^{H}\alpha_h(1-\sum_{k=1}^{K}\pi_{kh}) \end{aligned} \tag{B.70}$$

Next we maximize F with respect to the following parameters:

$$\theta_{z^h|d_i} = P(z^h|d_i) \tag{B.71}$$

$$\theta_{d_i} = P(d_i) \tag{B.72}$$

$$\theta_{\pi_{kh}} = \pi_{kh} \tag{B.73}$$

$$\theta_{\mu_{kh}} = \mu_{kh} \tag{B.74}$$

$$\theta_{\Sigma_{kh}} = \Sigma_{kh} \tag{B.75}$$

With align parameters we obtain:

$$\begin{aligned} F = &\sum_{i=1}^{M}\sum_{j=1}^{N_i}\sum_{h=1}^{H}\sum_{k=1}^{K} \beta_{kh}^{ij} \log\left[\theta_{z^h|d_i}\cdot \theta_{d_i}\cdot \theta_{\pi_{kh}}\cdot N(f_j|\theta_{\mu_{kh}},\theta_{\Sigma_{kh}})\right] \\ &+ \lambda(1-\sum_{i=1}^{M} \theta_{d_i}) + \sum_{i=1}^{M}\tau_i(1-\sum_{h=1}^{H} \theta_{zh|d_i}) + \sum_{h=1}^{H}\alpha_h(1-\sum_{k=1}^{K}\theta_{\pi_{kh}}) \end{aligned} \tag{B.76}$$

Setting the partial derivatives to zero gives:

$$\mu_{kh}^{new} = \frac{1}{p_{kh}}\sum_{i=1}^{M}\sum_{j=1}^{N_i}\beta_{kh}^{ij}\cdot f_j \quad h = 1,...,H;\ \ k = 1,...,K \tag{B.77}$$

$$\Sigma_{kh}^{new} = \frac{1}{p_{kh}}\sum_{i=1}^{M}\sum_{j=1}^{N_i}\beta_{kh}^{ij}\cdot (f_j-\mu_{kh}^{new})(f_j-\mu_{kh}^{new})^T \tag{B.78}$$

$$= \left(\frac{1}{p_{kh}}\sum_{i=1}^{M}\sum_{j=1}^{N_i}\beta_{kh}^{ij}\cdot f_j^2\right) - (\mu_{kh}^{new})^2 \quad h = 1,...,H;\ \ k = 1,...,K \tag{B.79}$$

where

$$p_{kh} = \sum_{i=1}^{M} \sum_{j=1}^{N_i} \beta_{kh}^{ij} \tag{B.80}$$

For the other parameters we obtain:

$$\frac{\partial F}{\partial \theta_{z^h|d_i}} = \frac{\sum_{j=1}^{N_i} \sum_{k=1}^{K} \beta_{kh}^{ij}}{\partial \theta_{z^h|d_i}} - \tau_i = 0 \quad h = 1,...,H; \;\; i = 1,...,M \tag{B.81}$$

$$\frac{\partial F}{\partial \theta_{d_i}} = \frac{\sum_{h=1}^{H} \sum_{j=1}^{N_i} \sum_{k=1}^{K} \beta_{kh}^{ij}}{\partial \theta_{d_i}} - \lambda = 0 \quad i = 1,...,M; \tag{B.82}$$

$$\frac{\partial F}{\partial \theta_{\pi_{kh}}} = \frac{\sum_{i=1}^{M} \sum_{j=1}^{N_i} \beta_{kh}^{ij}}{\partial \theta_{\pi_{kh}}} - \alpha_h = 0 \quad h = 1,...,H; \;\; k = 1,...,K \tag{B.83}$$

Solving for the desired parameters gives:

$$\theta_{z^h|d_i} = P(z^h|d_i) = \frac{\sum_{j=1}^{N_i} \sum_{k=1}^{K} \beta_{kh}^{ij}}{\tau_i} \quad h = 1,...,H; \;\; i = 1,...,M \tag{B.84}$$

$$\theta_{d_i} = P(d_i) = \frac{\sum_{h=1}^{H} \sum_{j=1}^{N_i} \sum_{k=1}^{K} \beta_{kh}^{ij}}{\lambda} \quad i = 1,...,M; \tag{B.85}$$

$$\theta_{\pi_{kh}} = \pi_{kh} = \frac{\sum_{i=1}^{M} \sum_{j=1}^{N_i} \beta_{kh}^{ij}}{\alpha_h} \quad h = 1,...,H; \;\; k = 1,...,K \tag{B.86}$$

Taking the normalization constraints into account, we can solve for the Lagrange multiplier:

$$\tau_i = \sum_{h=1}^{H} \sum_{j=1}^{N_i} \sum_{k=1}^{K} \beta_{kh}^{ij} \tag{B.87}$$

$$\lambda = \sum_{h=1}^{H} \sum_{i=1}^{M} \sum_{j=1}^{N_i} \sum_{k=1}^{K} \beta_{kh}^{ij} \tag{B.88}$$

$$\alpha_h = \sum_{k=1}^{K} \sum_{i=1}^{M} \sum_{j=1}^{N_i} \beta_{kh}^{ij} \tag{B.89}$$

The remaining M-step equations are:

$$P(z^h|d_i)^{new} = \frac{\sum_{j=1}^{N_i} \sum_{k=1}^{K} \beta_{kh}^{ij}}{\sum_{h=1}^{H} \sum_{j=1}^{N_i} \sum_{k=1}^{K} \beta_{kh}^{ij}} \quad h = 1,...,H; \;\; i = 1,...,M \tag{B.90}$$

$$= \frac{\sum_{j=1}^{N_i} \sum_{k=1}^{K} \beta_{kh}^{ij}}{N_i} \quad h = 1,...,H; \;\; i = 1,...,M \tag{B.91}$$

$$P(d_i)^{new} = \frac{\sum_{h=1}^{H} \sum_{j=1}^{N_i} \sum_{k=1}^{K} \beta_{kh}^{ij}}{\sum_{h=1}^{H} \sum_{i=1}^{M} \sum_{j=1}^{N_i} \sum_{k=1}^{K} \beta_{kh}^{ij}} \quad i = 1,...,M; \tag{B.92}$$

$$= \frac{N_i}{\sum_i N_i} \quad i = 1,...,M; \tag{B.93}$$

$$\pi_{kh}^{new} = \frac{\sum_{i=1}^{M}\sum_{j=1}^{N_i}\beta_{kh}^{ij}}{\sum_{k=1}^{K}\sum_{i=1}^{M}\sum_{j=1}^{N_i}\beta_{kh}^{ij}} \quad h = 1,...,H; \;\; k = 1,...,K \tag{B.94}$$

C. Derivation of the EM Algorithm for the Multi-Model PLSA Model

The probability of a visual words $w_{v,j}$ in image d_i according to the mm-pLSA is:

$$P(w_{v,j},d_i) = \sum_{l=1}^{L}\sum_{k=1}^{K} P(d_i)P(z_{top,j}=l|d_i)P(z_{v,j}=k|z_{top,j}=l)P(w_{v,j}|z_{v,j}=k) \tag{C.1}$$

where L is the total number of top-level concepts and K the total number of visual topics.
The probability of a tag $w_{t,j}$ associated with image d_i according to the mm-pLSA is:

$$P(w_{t,j},d_i) = \sum_{l=1}^{L}\sum_{p=1}^{P} P(d_i)P(z_{top,j}=l|d_i)P(z_{t,j}=p|z_{top,j}=l)P(w_{t,j}|z_{t,j}=p) \tag{C.2}$$

where P denotes the total number of tag topics.
The likelihood of the database consisting of observed pairs of both kinds (given all observations are independent) is then given by:

$$L = \prod_{i=1}^{M}\left[\prod_{m=1}^{N_i^v} P(w_{v,m},d_i)\prod_{n=1}^{N_i^t} P(w_{t,n},d_i)\right] \tag{C.3}$$

where N_i^v and N_i^t denote the number of visual words and tags respectively the image consists of, i.e., we assume that an image is d_i can be written as $\mathbf{w_i} = \{\mathbf{w^v w^t}\} = \{w_1^v,...w_{N_i^v}^v, w_1^t,...w_{N_i^t}^t\}$.
For the log-likelihood of the database we derive:

$$l = \sum_{i=1}^{M}\left[\sum_{m=1}^{N_i^v}\log P(w_{v,m},d_i + \sum_{n=1}^{N_i^t}\log P(w_{t,n},d_i)\right] \tag{C.4}$$

$$= \sum_{i=1}^{M}\left\{\sum_{m=1}^{N_i^v}\log\left[\sum_{l=1}^{L}\sum_{k=1}^{K} P(d_i)P(z_{top,j}=l|d_i)P(z_{v,j}=k|z_{top,j}=l)P(w_{v,m}|z_{v,j}=k)\right]\right.$$
$$\left. + \sum_{n=1}^{N_i^t}\log\left[\sum_{l=1}^{L}\sum_{p=1}^{P} P(d_i)P(z_{top,j}=l|d_i)P(z_{t,j}=p|z_{top,j}=l)P(w_{t,n}|z_{t,j}=p)\right]\right\} \tag{C.5}$$

Due to the sum inside the logarithm direct maximization is difficult. Thus we use the iterative Expectation Maximization (EM) algorithm.

C. Derivation of the EM Algorithm for the Multi-Model PLSA Model

We introduce the following indicator variables:

$$\Delta c_{lk} = \begin{cases} 1 & \text{if the pair } (w_{v,m}, d_i) \text{ was generated by the } l\text{-th top-level topic} \\ & \text{and the } k\text{-th visual topic} \\ 0 & \text{otherwise} \end{cases} \tag{C.6}$$

$$\Delta d_{lp} = \begin{cases} 1 & \text{if the pair } (w_{t,n}, d_i) \text{ was generated by the } l\text{-th top-level topic} \\ & \text{and the } p\text{-th tag topic} \\ 0 & \text{otherwise} \end{cases} \tag{C.7}$$

for which we assume that they can be observed.
The data likelihood for the completely observable data is called the complete data likelihood L^{compl} and is given by:

$$L^{compl} = \prod_{i=1}^{M} \left(\prod_{m=1}^{N_i^v} P(w_{v,m}, d_i, \Delta c) \prod_{n=1}^{N_i^t} P(w_{t,n}, d_i, \Delta d) \right) \tag{C.8}$$

with

$$\Delta c = (\Delta c_{11}, ..., \Delta c_{1K}, ..., \Delta c_{LK}) \tag{C.9}$$

$$\Delta d = (\Delta d_{11}, ..., \Delta d_{1K}, ..., \Delta d_{LP}) \tag{C.10}$$

$$P(w_{v,m}, d_i, \Delta c) = \prod_{l=1}^{L} \prod_{k=1}^{K} [P(d_i) P(z_{top,j} = l | d_i) P(z_{v,j} = k | z_{top,j} = l) P(w_{v,m} | z_{v,j} = k)]^{\Delta c_{lk}} \tag{C.11}$$

$$P(w_{t,n}, d_i, \Delta d) = \prod_{l=1}^{L} \prod_{p=1}^{P} [P(d_i) P(z_{top,j} = l | d_i) P(z_{t,j} = p | z_{top,j} = l) P(w_{t,n} | z_{t,j} = p)]^{\Delta d_{lp}} \tag{C.12}$$

For the complete log data likelihood we obtain:

$$l_{compl} = \sum_{i=1}^{M} \sum_{m=1}^{N_i^v} \log P(w_{v,m}, d_i, \Delta c) + \sum_{i=1}^{M} \sum_{n=1}^{N_i^t} \log P(w_{t,n}, d_i, \Delta d) \tag{C.13}$$

$$\begin{aligned} = & \sum_{i=1}^{M} \sum_{m=1}^{N_i^v} \sum_{l=1}^{L} \sum_{k=1}^{K} \Delta c_{lk} \log P(d_i) P(z_{top,j} = l | d_i) P(z_{v,j} = k | z_{top,j} = l) P(w_{v,m} | z_{v,j} = k) \\ & + \sum_{i=1}^{M} \sum_{n=1}^{N_i^t} \sum_{l=1}^{L} \sum_{p=1}^{P} \Delta d_{lp} \log P(d_i) P(z_{top,j} = l | d_i) P(z_{t,j} = p | z_{top,j} = l) P(w_{t,n} | z_{t,j} = p) \end{aligned} \tag{C.14}$$

Denoting the l-th top-level topic by z_{top}^l, i.e., $z_{top} = l$, the k-th visual topic by z_v^k, i.e., $z_v = k$, and the p-th tag topic by z_t^p, i.e., $z_t = p$, the expectation of the indicator variables is derived as follows:

Δc:

$$P(\Delta c|w_{v,m},d_i) = \frac{P(w_{v,m},d_i,\Delta c)}{P(w_{v,m},d_i)} \tag{C.15}$$

$$= \frac{\prod_{l=1}^{L}\prod_{k=1}^{K}\left[P(d_i)P(z_{top}^l|d_i)P(z_v^k|z_{top}^l)P(w_{v,m}|z_v^k)\right]^{\Delta c_{lk}}}{\sum_{l=1}^{L}\sum_{k=1}^{K}P(d_i)P(z_{top}^l|d_i)P(z_v^k|z_{top}^l)P(w_{v,m}|z_v^k)} \tag{C.16}$$

$$E(\Delta c_{lk}|w_{v,m},d_i) = P(\Delta c_{lk}=1|w_{v,m},d_i)\cdot 1 + P(\Delta c_{lk}=0|w_{v,m},d_i)\cdot 0 \tag{C.17}$$

$$= P(\Delta c_{lk}=1|w_{v,m},d_i)\cdot 1 \tag{C.18}$$

$$= \frac{P(d_i)P(z_{top}^l|d_i)P(z_v^k|z_{top}^l)P(w_{v,m}|z_v^k)}{\sum_{l=1}^{L}\sum_{k=1}^{K}P(d_i)P(z_{top}^l|d_i)P(z_v^k|z_{top}^l)P(w_{v,m}|z_v^k)} \tag{C.19}$$

$$\tag{C.20}$$

Δd:

$$P(\Delta d|w_{t,n},d_i) = \frac{P(w_{t,n},d_i,\Delta d)}{P(w_{t,n},d_i)} \tag{C.21}$$

$$E(\Delta d_{lp}|w_{t,n},d_i) = \frac{P(d_i)P(z_{top}^l|d_i)P(z_t^p|z_{top}^l)P(w_{t,n}|z_t^p)}{\sum_{l=1}^{L}\sum_{k=1}^{K}P(d_i)P(z_{top}^l|d_i)P(z_t^p|z_{top}^l)P(w_{t,n}|z_t^p)} \tag{C.22}$$

$$\tag{C.23}$$

Assuming we have N^v and different words in our visual vocabulary and denoting with w_v^r the r-th words of this vocabulary, we see that if $g \neq f$ if $w_{v,g} = w_v^r = w_{v,f}$ then $E(\Delta c_{lk}|w_{v,g},d_i) = E(\Delta c_{lk}|w_{v,f},d_i)$, i.e., if a word from the visual vocabulary appears twice (or more) in an image the associated expectations of the indicator variables $E(\Delta c_{lk}|w_{v,m},d_i)$ are equal. The same holds for $E(\Delta d_{lp}|w_{t,n},d_i)$. Thus we only need to calculate the expectation of the indicator variables once for each word in the respective vocabularies for each document d_i:

$$E(\Delta c_{lk}|w_v^m,d_i) = \frac{P(d_i)P(z_{top}^l|d_i)P(z_v^k|z_{top}^l)P(w_v^m|z_v^k)}{\sum_{l=1}^{L}\sum_{k=1}^{K}P(d_i)P(z_{top}^l|d_i)P(z_v^k|z_{top}^l)P(w_v^m|z_v^k)} \tag{C.24}$$

$$= c_{lk}^{im} \tag{C.25}$$

$$E(\Delta d_{lp}|w_t^n,d_i) = \frac{P(d_i)P(z_{top}^l|d_i)P(z_t^p|z_{top}^l)P(w_t^n|z_t^p)}{\sum_{l=1}^{L}\sum_{k=1}^{K}P(d_i)P(z_{top}^l|d_i)P(z_t^p|z_{top}^l)P(w_t^n|z_t^p)} \tag{C.26}$$

$$= d_{lp}^{in} \tag{C.27}$$

The calculation of c_{lk}^{im} and d_{lp}^{in} is the E-step of our EM algorithm.
To derive the M-step equations we need to maximize $E(l_{compl})$. Therefore we express the log-

likelihood l of the database in terms of the co-occurrence tables' entries:

$$l = \sum_{i=1}^{M} \left[\sum_{m=1}^{N^v} n(w_v^m, d_i) \log P(w_v^m, d_i) + \sum_{n=1}^{N^t} n(w_t^n, d_i) \log P(w_t^n, d_i) \right] \tag{C.28}$$

and derive:

$$\begin{aligned} E(l_{comp}) = & \sum_{i=1}^{M} \sum_{m=1}^{N^v} n(w_v^m, d_i) \sum_{l=1}^{L} \sum_{k=1}^{K} c_{lk}^{im} \log[P(d_i) P(z_{top}^l | d_i) P(z_v^k | z_{top}^l) P(w_v^m | z_v^k)] \\ & + \sum_{i=1}^{M} \sum_{n=1}^{N^t} n(w_t^n, d_i) \sum_{l=1}^{L} \sum_{p=1}^{P} d_{lp}^{in} \log[P(d_i) P(z_{top}^l | d_i) P(z_t^p | z_{top}^l) P(w_t^n | z_t^p)] \end{aligned} \tag{C.29}$$

$E(l_{compl})$ is maximized with respect to the following constraints:

$$\sum_{i=1}^{M} P(d_i) = 1 \tag{C.30}$$

$$\sum_{l=1}^{L} P(z_{top}^l | d_i) = 1 \quad i = 1, ..., M \tag{C.31}$$

$$\sum_{k=1}^{K} P(z_v^k | z_{top}^l) + \sum_{p=1}^{P} P(z_t^p | z_{top}^l) = 1 \quad l = 1, ..., L \tag{C.32}$$

$$\sum_{m=1}^{N^v} P(w_v^m | z_v^k) = 1 \quad k = 1, ..., K \tag{C.33}$$

$$\sum_{n=1}^{N^t} P(w_t^n | z_t^p) = 1 \quad p = 1, ..., P \tag{C.34}$$

We introduce the Lagrange multipliers: $\alpha, \lambda_i, \tau_k, \delta_p, \varepsilon_l$:

$$\begin{aligned} F = & \sum_{i=1}^{M} \sum_{m=1}^{N^v} \sum_{l=1}^{L} \sum_{k=1}^{K} n(w_v^j, d_i)\ c_{lk}^{im}\ log \left[P(d_i) P(z_{top}^l | d_i) P(z_v^k | z_{top}^l) P(w_v^m | z_v^k) \right] \\ & + \sum_{i=1}^{M} \sum_{n=1}^{N^t} \sum_{l=1}^{L} \sum_{p=1}^{P} n(w_t^j, d_i)\ d_{lp}^{in}\ log \left[P(d_i) P(z_{top}^l | d_i) P(z_t^p | z_{top}^l) P(w_t^n | z_t^p) \right] \\ & + \alpha (1 - \sum_{i=1}^{M} P(d_i)) \ + \ \sum_{i=1}^{M} \lambda_i (1 - \sum_{l=1}^{L} P(z_{top}^l | d_i)) \\ & + \sum_{l=1}^{L} \varepsilon_l (1 - \sum_{k=1}^{K} P(z_v^k | z_{top}^l) - \sum_{p=1}^{P} P(z_t^p | z_{top}^l)) \\ & + \sum_{k=1}^{K} \tau_k (1 - \sum_{m=1}^{N^v} P(w_v^m | z_v^k)) \ + \ \sum_{p=1}^{P} \delta_p (1 - \sum_{n=1}^{N^t} P(w_t^n | z_t^p)) \end{aligned} \tag{C.35}$$

Now we maximize F with respect to the parameters:

$$\vartheta_{d_i} = P(d_i) \tag{C.36}$$

$$\vartheta_{z^l_{top}|d_i} = P(z^l_{top}|d_i) \tag{C.37}$$

$$\vartheta_{z^k_v|z^l_{top}} = P(z^k_v|z^l_{top}) \tag{C.38}$$

$$\vartheta_{z^p_t|z^l_{top}} = P(z^p_t|z^l_{top}) \tag{C.39}$$

$$\vartheta_{w^m_v|z^k_v} = P(w^m_v|z^k_v) \tag{C.40}$$

$$\vartheta_{w^n_t|z^p_t} = P(w^n_t|z^p_t) \tag{C.41}$$

Setting the partial derivatives to zero gives:

$$\frac{\partial F}{\partial \vartheta_{d_i}} = \frac{\sum_{m=1}^{N^v}\sum_{l=1}^{L}\sum_{k=1}^{K} n(w^m_v, d_i)c^{im}_{lk}}{\vartheta_{d_i}} + \frac{\sum_{n=1}^{N^t}\sum_{l=1}^{L}\sum_{p=1}^{P} n(w^m_t, d_i)d^{in}_{lp}}{\vartheta_{d_i}} - \alpha = 0 \tag{C.42}$$

$$i = 1...M$$

$$\frac{\partial F}{\partial \vartheta_{z^l_{top}|d_i}} = \frac{\sum_{m=1}^{N^v}\sum_{k=1}^{K} n(w^m_v, d_i)c^{im}_{lk}}{\vartheta_{z^l_{top}|d_i}} + \frac{\sum_{n=1}^{N^t}\sum_{p=1}^{P} n(w^n_t, d_i)d^{in}_{lp}}{\vartheta_{z^l_{top}|d_i}} - \lambda_i = 0 \tag{C.43}$$

$$i = 1...M, l = 1...L$$

$$\frac{\partial F}{\partial \vartheta_{z^k_v|z^l_{top}}} = \frac{\sum_{i=1}^{M}\sum_{m=1}^{N^v} n(w^m_v, d_i)c^{im}_{lk}}{\vartheta_{z^k_v|z^l_{top}}} - \varepsilon_l = 0 \tag{C.44}$$

$$k = 1...K, l = 1...L$$

$$\frac{\partial F}{\partial \vartheta_{z^p_t|z^l_{top}}} = \frac{\sum_{i=1}^{M}\sum_{n=1}^{N^t} n(w^n_t, d_i)d^{in}_{lp}}{\vartheta_{z^p_t|z^l_{top}}} - \varepsilon_l = 0 \tag{C.45}$$

$$p = 1...P, l = 1...L$$

$$\frac{\partial F}{\partial \vartheta_{w^m_v|z^k_v}} = \frac{\sum_{i=1}^{M}\sum_{l=1}^{L} n(w^m_v, d_i)c^{im}_{lk}}{\vartheta_{w^m_v|z^k_v}} - \tau_k = 0 \tag{C.46}$$

$$m = 1...N^v, k = 1...K$$

$$\frac{\partial F}{\partial \vartheta_{w^n_t|z^p_t}} = \frac{\sum_{i=1}^{M}\sum_{l=1}^{L} n(w^n_t, d_i)d^{in}_{lk}}{\vartheta_{w^n_t|z^p_t}} - \delta_p = 0 \tag{C.47}$$

$$m = 1...N^t, p = 1...P$$

Solving for the desired parameter gives:

$$\vartheta_{d_i} = P(d_i) = \frac{\sum_{m=1}^{N^v}\sum_{l=1}^{L}\sum_{k=1}^{K} n(w^m_v, d_i)c^{im}_{lk} + \sum_{n=1}^{N^t}\sum_{l=1}^{L}\sum_{p=1}^{P} n(w^n_t, d_i)d^{in}_{lp}}{\alpha} \tag{C.48}$$

$$i = 1,...,M$$

$$\vartheta_{z^l_{top}|d_i} = P(z^{top}_l|d_i) = \frac{\sum_{m=1}^{N^v}\sum_{k=1}^{K} n(w^m_v,d_i)c^{im}_{lk} + \sum_{n=1}^{N^t}\sum_{p=1}^{P} n(w^n_t,d_i)d^{in}_{lp}}{\lambda_i} \tag{C.49}$$

$$i = 1,...,M \quad l = 1,...,L$$

$$\vartheta_{z^k_v|z^l_{top}} = P(z^v_k|z^{top}_l) = \frac{\sum_{i=1}^{M}\sum_{m=1}^{N^v} n(w^m_v,d_i)c^{im}_{lk}}{\varepsilon_l} \tag{C.50}$$

$$k = 1,...,K \quad l = 1,...,L$$

$$\vartheta_{z^p_t|z^l_{top}} = P(z^t_p|z^{top}_l) = \frac{\sum_{i=1}^{M}\sum_{n=1}^{N^t} n(w^n_t,d_i)d^{in}_{lp}}{\varepsilon_l} \tag{C.51}$$

$$p = 1,...,P \quad l = 1,...,L$$

$$\vartheta_{w^m_v|z^k_v} = P(w^v_m|z^v_k) = \frac{\sum_{i=1}^{M}\sum_{l=1}^{L} n(w^m_v,d_i)c^{im}_{lk}}{\tau_k} \tag{C.52}$$

$$m = 1,...,N^v \quad k = 1,...,K$$

$$\vartheta_{w^n_t|z^p_t} = P(w^t_n|z^t_p) = \frac{\sum_{i=1}^{M}\sum_{l=1}^{L} n(w^n_t,d_i)d^{in}_{lp}}{\delta_p} \tag{C.53}$$

$$m = 1,...,N^t \quad p = 1,...,P$$

Taking the normalization constraints into account we can solve for the Lagrange multiplier:

$$\alpha = \sum_{i=1}^{M}\left(\sum_{m=1}^{N^v}\sum_{l=1}^{L}\sum_{k=1}^{K} n(w^m_v,d_i)c^{im}_{lk} + \sum_{n=1}^{N^t}\sum_{l=1}^{L}\sum_{p=1}^{P} n(w^n_t,d_i)d^{in}_{lp}\right) \tag{C.54}$$

$$\lambda_i = \sum_{l=1}^{L}\left(\sum_{m=1}^{N^v}\sum_{k=1}^{K} n(w^m_v,d_i)c^{im}_{lk} + \sum_{n=1}^{N^t}\sum_{p=1}^{P} n(w^n_t,d_i)d^{in}_{lp}\right) \tag{C.55}$$

$$\tau_k = \sum_{m=1}^{N^v}\sum_{i=1}^{M}\sum_{l=1}^{L} n(w^m_v,d_i)c^{im}_{lk} \tag{C.56}$$

$$\delta_p = \sum_{n=1}^{N^t}\sum_{i=1}^{M}\sum_{l=1}^{L} n(w^n_t,d_i)d^{in}_{lp} \tag{C.57}$$

$$\varepsilon_l = \sum_{k=1}^{K}\sum_{i=1}^{M}\sum_{m=1}^{N^v} n(w^m_v,d_i)c^{im}_{lk} + \sum_{p=1}^{P}\sum_{i=1}^{M}\sum_{n=1}^{N^t} n(w^n_t,d_i)d^{in}_{lp} \tag{C.58}$$

Thus we derive the following M-step equations:

$$P(d_i)^{new} = \frac{\sum_{m=1}^{N^v}\sum_{l=1}^{L}\sum_{k=1}^{K} n(w_v^m, d_i) c_{lk}^{im} + \sum_{n=1}^{N^t}\sum_{l=1}^{L}\sum_{p=1}^{P} n(w_t^n, d_i) d_{lp}^{in}}{\sum_{i=1}^{M}\left(\sum_{m=1}^{N^v}\sum_{l=1}^{L}\sum_{k=1}^{K} n(w_v^m, d_i) c_{lk}^{im} + \sum_{n=1}^{N^t}\sum_{l=1}^{L}\sum_{p=1}^{P} n(w_t^n, d_i) d_{lp}^{in}\right)} \tag{C.59}$$

$$P(z_l^{top}|d_i)^{new} = \frac{\sum_{m=1}^{N^v}\sum_{k=1}^{K} n(w_v^m, d_i) c_{lk}^{im} + \sum_{n=1}^{N^t}\sum_{p=1}^{P} n(w_t^n, d_i) d_{lp}^{in}}{\sum_{l=1}^{L}\left(\sum_{m=1}^{N^v}\sum_{k=1}^{K} n(w_v^m, d_i) c_{lk}^{im} + \sum_{n=1}^{N^t}\sum_{p=1}^{P} n(w_t^n, d_i) d_{lp}^{in}\right)} \tag{C.60}$$

$$P(z_k^v|z_l^{top})^{new} = \frac{\sum_{i=1}^{M}\sum_{m=1}^{N^v} n(w_v^m, d_i) c_{lk}^{im}}{\sum_{k=1}^{K}\sum_{i=1}^{M}\sum_{m=1}^{N^v} n(w_v^m, d_i) c_{lk}^{im} + \sum_{p=1}^{P}\sum_{i=1}^{M}\sum_{n=1}^{N^t} n(w_t^n, d_i) d_{lp}^{in}} \tag{C.61}$$

$$P(z_p^t|z_l^{top})^{new} = \frac{\sum_{i=1}^{M}\sum_{n=1}^{N^t} n(w_t^n, d_i) d_{lp}^{in}}{\sum_{k=1}^{K}\sum_{i=1}^{M}\sum_{m=1}^{N^v} n(w_v^m, d_i) c_{lk}^{im} + \sum_{p=1}^{P}\sum_{i=1}^{M}\sum_{n=1}^{N^t} n(w_t^n, d_i) d_{lp}^{in}} \tag{C.62}$$

$$P(w_m^v|z_k^v)^{new} = \frac{\sum_{i=1}^{M}\sum_{l=1}^{L} n(w_v^m, d_i) c_{lk}^{im}}{\sum_{m=1}^{N^v}\sum_{i=1}^{M}\sum_{l=1}^{L} n(w_v^m, d_i) c_{lk}^{im}} \tag{C.63}$$

$$P(w_n^t|z_p^t)^{new} = \frac{\sum_{i=1}^{M}\sum_{l=1}^{L} n(w_t^n, d_i) d_{lp}^{in}}{\sum_{n=1}^{N^t}\sum_{i=1}^{M}\sum_{l=1}^{L} n(w_t^n, d_i) d_{lp}^{in}} \tag{C.64}$$

The expression for $P(d_i)$ in Eq. (C.59) can be further simplified:

$$\sum_{m=1}^{N^v}\sum_{l=1}^{L}\sum_{k=1}^{K} n(w_v^m, d_i) c_{lk}^{im} = \sum_{m=1}^{N^v}\sum_{l=1}^{L}\sum_{k=1}^{K} n(w_v^m, d_i) \frac{P(d_i)P(z_{top}^l|d_i)P(z_v^k|z_{top}^l)P(w_v^m|z_v^k)}{\sum_{l=1}^{L}\sum_{k=1}^{K} P(d_i)P(z_{top}^l|d_i)P(z_v^k|z_{top}^l)P(w_v^m|z_v^k)} \tag{C.65}$$

$$= \sum_{m=1}^{N^v} n(w_v^m, d_i) \frac{\sum_{l=1}^{L}\sum_{k=1}^{K} P(d_i)P(z_{top}^l|d_i)P(z_v^k|z_{top}^l)P(w_v^m|z_v^k)}{\sum_{l=1}^{L}\sum_{k=1}^{K} P(d_i)P(z_{top}^l|d_i)P(z_v^k|z_{top}^l)P(w_v^m|z_v^k)} \tag{C.66}$$

$$= \sum_{m=1}^{N^v} n(w_v^m, d_i) \tag{C.67}$$

$$\sum_{n=1}^{N^t}\sum_{l=1}^{L}\sum_{p=1}^{P} n(w_t^n) d_{lp}^{in} = \sum_{n=1}^{N^t} n(w_t^n) \tag{C.68}$$

Thus we get

$$P(d_i)^{new} = \frac{\sum_{m=1}^{N^v} n(w_v^m, d_i) + \sum_{n=1}^{N^t} n(w_t^n)}{\sum_{i=1}^{M} \left(\sum_{m=1}^{N^v} n(w_v^m, d_i) + \sum_{n=1}^{N^t} n(w_t^n) \right)} \tag{C.69}$$

$$= \frac{N_i^v + N_i^t}{\sum_{i=1}^{M} N_i^v + N_i^t} \tag{C.70}$$

List of Figures

List of Tables

Bibliography

[1] *http://www.flickr.com/*.

[2] *http://www.marketingvox.com/flickr-photo-count-hits-3-billion-041831/*.

[3] J. J. G. A. C. Murillo and C. Sagues. Surf features for efficient robot localization with omnidirectional images. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 3901–3907, 2007.

[4] S. Agarwal and D. Roth. Learning a sparse representation for object detection. In *ECCV '02: Proceedings of the 7th European Conference on Computer Vision-Part IV*, pages 113–130, 2002.

[5] P. Ahrendt, C. Goutte, and J. Larsen. Co-occurrence models in music genre classification. In *IEEE International Workshop on Machine Learning for Signal Processing*, pages 247–252, 2005.

[6] R. A. Baeza-Yates and B. Ribeiro-Neto. *Modern Information Retrieval*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1999.

[7] K. Barnard, P. Duygulu, D. Forsyth, N. de Freitas, D. M. Blei, and M. I. Jordan. Matching words and pictures. *Journal of Machine Learning Research*, 3, 2003.

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

[9] A. C. Berg, T. L. Berg, and J. Malik. Shape matching and object recognition using low distortion correspondences. In *IEEE Conference on Computer Vision and Pattern Recognition 2005 (CVPR'05)*, volume 1, pages 26–33 vol. 1, 2005.

[10] T. L. Berg and D. Forsyth. Automatic ranking of iconic images. Technical Report UCB/EECS-2007-13, EECS Department, University of California, Berkeley, 2007.

[11] D. Blei, T. L. Griffiths, M. I. Jordan, and J. B. Tenenbaum. Hierarchical topic models and the nested chinese restaurant process. In S. Thrun, L. Saul, and B. Schölkopf, editors, *Advances in Neural Information Processing Systems 16*. 2004.

[12] D. M. Blei and M. I. Jordan. Modeling annotated data. In *SIGIR '03: Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Informaion Retrieval*, pages 127–134, 2003.

[13] D. M. Blei and J. D. Lafferty. A correlated topic model of science. *The Annals of Applied Statistics*, 1(1):17–35, 2007.

[14] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, 2003.

[15] A. Bosch, A. Zisserman, and X. Munoz. Scene classification via pLSA. In *Proceedings of the European Conference on Computer Vision*, 2006.

[16] L. Cao and L. Fei-Fei. Spatially coherent latent topic model for concurrent object segmentation and classification. In *IEEE International Conference on Computer Vision (ICCV)*, 2007.

[17] C. Carson, S. Belongie, H. Greenspan, and J. Malik. Blobworld: image segmentation using expectation-maximization and its application to image querying. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(8):1026–1038, 2002.

[18] C. Chemudugunta, P. Smyth, and M. Steyvers. Modeling general and specific aspects of documents with a probabilistic topic model. In B. Schölkopf, J. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems 19*, pages 241–248. MIT Press, 2007.

[19] D. Crandall, L. Backstrom, D. Huttenlocher, and J. Kleinberg. Mapping the world's photos. In *Proc. 18th International World Wide Web Conference*, 2009.

[20] G. Csurka, C. R. Dance, L. Fan, J. Willamowski, and C. Bray. Visual categorization with bags of keypoints. In *Workshop on Statistical Learning in Computer Vision, ECCV*, pages 1–22, 2004.

[21] R. Datta, D. Joshi, J. Li, and J. Z. Wang. Image retrieval: Ideas, influences, and trends of the new age. *ACM Comput. Surv.*, 40(2):1–60, 2008.

[22] S. Deerwester, S. T. Dumais, G. W. Furnas, T. K. Landauer, and R. Harshman. Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, 41:391–407, 1990.

[23] A. Dempster, N. Laird, and D. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society*, 39(1):1–38, 1977.

[24] M. N. Do and M. Vetterli. Wavelet-based texture retrieval using generalized gaussian density and kullback-leibler distance. *IEEE Transactions on Image Processing*, 11(2):146–158, 2002.

[25] A. Dong and B. Bhanu. Active concept learning for image retrieval in dynamic databases. In *ICCV '03: Proceedings of the Ninth IEEE International Conference on Computer Vision*, page 90, 2003.

[26] 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.

[27] J. Fan, Y. Gao, H. Luo, and G. Xu. Automatic image annotation by using concept-sensitive salient objects for image content representation. In *SIGIR '04: Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 361–368, 2004.

[28] 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. *Comput. Vis. Image Underst.*, 106(1):59–70, 2007.

[29] P. F. Felzenszwalb and D. P. Huttenlocher. Pictorial structures for object recognition. *Int. J. Comput. Vision*, 61(1):55–79, 2005.

[30] R. Fergus, L. Fei-Fei, P. Perona, and A. Zisserman. Learning object categories from google's image search. In *ICCV '05: Proceedings of the Tenth IEEE International Conference on Computer Vision*, pages 1816–1823, 2005.

[31] R. Fergus, P. Perona, and A. Zisserman. Object class recognition by unsupervised scale-invariant learning. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 2, pages 264–271, 2003.

[32] M. A. Fischler and R. A. Elschlager. The representation and matching of pictorial structures. *IEEE Transactions on Computers*, 100(22):67–92, 1973.

[33] T. Gevers and A. W. M. Smeulders. Content-based image retrieval: an overview. In G. Medioni and S. B. Kang, editors, *Emerging Topics in Computer Vision*, pages 333 – 384. Prentice Hall, 2004.

[34] D. Grangier and S. Bengio. A discriminative kernel-based approach to rank images from text queries. *IEEE Trans. Pattern Anal. Mach. Intell.*, 30(8):1371–1384, 2008.

[35] G. Griffin, A. Holub, and P. Perona. Caltech-256 object category dataset. Technical Report 7694, California Institute of Technology, 2007.

[36] T. L. Griffiths and M. Steyvers. Finding scientific topics. *Proc. Natl. Acad. Sci. USA*, 101 Suppl 1:5228–5235, 2004.

[37] T. L. Griffiths, M. Steyvers, D. M. Blei, and J. B. Tenenbaum. Integrating topics and syntax. In L. K. Saul, Y. Weiss, and L. Bottou, editors, *Advances in Neural Information Processing Systems 17*, pages 537–544. MIT Press, 2005.

[38] G. E. Hinton. Training products of experts by minimizing contrastive divergence. *Neural Computation*, 14(8):1771–1800, 2002.

[39] G. E. Hinton and R. R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507, 2006.

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

[41] T. Hofmann, J. Puzicha, and M. I. Jordan. Learning from dyadic data. In *Proceedings of the 1998 Conference on Advances in Neural Information Processing Systems II*, pages 466–472, 1999.

[42] W. H. Hsu, L. S. Kennedy, and S.-F. Chang. Video search reranking via information bottleneck principle. In *MULTIMEDIA '06: Proceedings of the 14th Annual ACM International Conference on Multimedia*, pages 35–44, 2006.

[43] W. H. Hsu, L. S. Kennedy, and S.-F. Chang. Reranking methods for visual search. *IEEE MultiMedia*, 14(3):14–22, 2007.

[44] E. Hörster, M. Slaney, M. Ranzato, and K. Weinberger. Unsupervised image ranking. In *Workshop on Large-Scale Multimedia Retrieval and Mining*, 2009, to appear.

[45] A. K. Jain and F. Farrokhnia. Unsupervised texture segmentation using gabor filters. In *IEEE International Conference on Systems, Man and Cybernetics*, pages 14–19, 1990.

[46] A. K. Jain and A. Vailaya. Image retrieval using color and shape. *Pattern Recognition*, 29:1233–1244, 1996.

[47] Y. Jing and S. Baluja. Pagerank for product image search. In *WWW '08: Proceeding of the 17th International Conference on World Wide Web*, pages 307–316, 2008.

[48] T. Joachims. Optimizing search engines using clickthrough data. In *KDD '02: Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 133–142, 2002.

[49] K. Kavukcuoglu, M. Ranzato, and Y. LeCun. Fast inference in sparse coding algorithms with applications to object recognition. Technical report, Computational and Biological Learning Lab, Courant Institute, NYU, 2008. Tech Report CBLL-TR-2008-12-01.

[50] L. S. Kennedy and M. Naaman. Generating diverse and representative image search results for landmarks. In *WWW '08: Proceeding of the 17th International Conference on World Wide Web*, pages 297–306, 2008.

[51] B. Ko and H. Byun. Probabilistic neural networks supporting multi-class relevance feedback in region-based image retrieval. In *ICPR '02: Proceedings of the 16 th International Conference on Pattern Recognition (ICPR'02) Volume 4*, page 40138, 2002.

[52] D. Larlus and F. Jurie. Latent mixture vocabularies for object categorization. In *British Machine Vision Conference*, 2006.

[53] B. Leibe, A. Leonardis, and B. Schiele. Combined object categorization and segmentation with an implicit shape model. In *In ECCV Workshop on Statistical Learning in Computer Vision*, pages 17–32, 2004.

[54] F.-F. Li and P. Perona. A Bayesian hierarchical model for learning natural scene categories. In *CVPR '05: Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Volume 2*, pages 524–531, 2005.

[55] J. Li and J. Z. Wang. Real-time computerized annotation of pictures. *IEEE Trans. Pattern Anal. Mach. Intell.*, 30(6):985–1002, 2008.

[56] R. Lienhart and W. Effelsberg. Automatic text segmentation and text recognition for video indexing. *ACM/Springer Multimedia Systems*, 8:69–81, 2000.

[57] R. Lienhart and A. Hartmann. Classifying images on the web automatically. *Journal of Electronic Imaging*, 11(4):445–454, 2002.

[58] R. Lienhart, E. Kuranov, and V. Pisarevsky. Empirical analysis of detection cascades of boosted classifiers for rapid object detection. In *DAGM 25th Pattern Recognition Symposium*, pages 297–304, 2003.

[59] R. Lienhart and M. Slaney. Plsa on large scale image databases. In *ICASSP 2007: Proceedings of the 2007 IEEE International Conference on Acoustics, Speech and Signal Processing*, volume 4, pages IV–1217–IV–1220, 2007.

[60] R. Lienhart and A. Wernike. Localizing and segmenting text in images, videos and web pages. *IEEE Transactions on Circuits and Systems for Video Technology*, 12(4):256–268, 2002.

[61] D. Liu and T. Chen. Semantic-shift for unsupervised object detection. In *CVPRW '06: Proceedings of the 2006 Conference on Computer Vision and Pattern Recognition Workshop*, page 16, 2006.

[62] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.

[63] D. R. Martin, C. C. Fowlkes, and J. Malik. Learning to detect natural image boundaries using local brightness, color, and texture cues. *IEEE Trans. Pattern Anal. Mach. Intell.*, 26(5):530–549, 2004.

[64] J. Matas, O. Chum, M. Urban, and T. Pajdla. Robust wide baseline stereo from maximally stable extremal regions. In *British Machine Vision Conference*, pages 384–393, 2002.

[65] K. Mikolajczyk and C. Schmid. Scale & affine invariant interest point detectors. *International Journal of Computer Vision*, 60(1):63–86, 2004.

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

[67] D. Mimno, W. Li, and A. McCallum. Mixtures of hierarchical topics with pachinko allocation. In *ICML '07: Proceedings of the 24th International Conference on Machine Learning*, pages 633–640, 2007.

[68] F. Monay and D. Gatica-Perez. Plsa-based image auto-annotation: constraining the latent space. In *MULTIMEDIA '04: Proceedings of the 12th Annual ACM International Conference on Multimedia*, pages 348–351, 2004.

[69] D. Nister and H. Stewenius. 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, 2006.

[70] S. Obdrzálek and J. Matas. Object recognition using local affine frames on distinguished regions. In *British Machine Vision Conference*, 2002.

[71] A. Oliva and A. B. Torralba. Modeling the shape of the scene: A holistic representation of the spatial envelope. *International Journal of Computer Vision*, 42(3):145–175, 2001.

[72] B. A. Olshausen and D. J. Field. Sparse coding with an overcomplete basis set: a strategy employed by v1? *Vision Research*, 37:3311–3325, 1997.

[73] G. Pass, R. Zabih, and J. Miller. Comparing images using color coherence vectors. In *MULTIMEDIA '96: Proceedings of the Fourth ACM International Conference on Multimedia*, pages 65–73, 1996.

[74] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman. Object retrieval with large vocabularies and fast spatial matching. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2007.

[75] J. Philbin, J. Sivic, and A. Zisserman. Geometric lda: A generative model for particular object discovery. In *Proceedings of the British Machine Vision Conference*, 2008.

[76] P. Quelhas, F. Monay, J.-M. Odobez, D. Gatica-Perez, T. Tuytelaars, and L. V. Gool. Modeling scenes with local descriptors and latent aspects. In *ICCV '05: Proceedings of the Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1*, pages 883–890, 2005.

[77] P. Quelhas and J.-M. Odobez. Natural scene image modeling using color and texture visterms. In *International Conference on Image and Video Retrieval (CIVR)*, pages 411–421, 2006.

[78] M. Ranzato, F. J. Huang, Y.-L. Boureau, and Y. LeCun. Unsupervised learning of invariant feature hierarchies with applications to object recognition. *IEEE Conference on Computer Vision and Pattern Recognition 2007 (CVPR'07*, pages 1–8, 2007.

[79] M. Rosen-Zvi, T. Griffiths, M. Steyvers, and P. Smyth. The author-topic model for authors and documents. In *AUAI '04: Proceedings of the 20th Conference on Uncertainty in Artificial Intelligence*, pages 487–494, 2004.

[80] Y. Rui and T. S. Huang. *Relevance feedback techniques in image retrieval*. Springer-Verlag, London, UK, 2001.

[81] Y. Rui, T. S. Huang, and S.-F. Chang. Image retrieval: Current techniques, promising directions, and open issues. *Journal of Visual Communication and Image Representation*, 10(1):39–62, 1999.

[82] B. C. Russell, W. T. Freeman, A. A. Efros, J. Sivic, and A. Zisserman. Using multiple segmentations to discover objects and their extent in image collections. In *CVPR '06:*

Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pages 1605–1614, 2006.

[83] R. R. Salakhutdinov and G. E. Hinton. Semantic hashing. In *Proc. SIGIR Workshop on Information Retrieval and Applications of Graphical Models*, 2007.

[84] C. Schmid and R. Mohr. Local greyvalue invariants for image retrieval. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19:530–535, 1997.

[85] 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)*, 2007.

[86] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22:888–905, 2000.

[87] J. Sivic, B. C. Russell, A. A. Efros, A. Zisserman, and W. T. Freeman. Discovering objects and their location in images. In *International Conference on Computer Vision (ICCV 2005)*, 2005.

[88] J. Sivic, B. C. Russell, A. Zisserman, W. T. Freeman, and A. A. Efros. Unsupervised discovery of visual object class hierarchies. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2008.

[89] J. Sivic and A. Zisserman. Video google: A text retrieval approach to object matching in videos. In *Proceedings of the International Conference on Computer Vision*, volume 2, pages 1470–1477, 2003.

[90] 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 Trans. Pattern Anal. Mach. Intell.*, 22(12):1349–1380, 2000.

[91] M. A. Stricker and M. Orengo. Similarity of color images. In *Storage and Retrieval for Image and Video Databases (SPIE)*, pages 381–392, 1995.

[92] E. Sudderth, A. Torralba, W. Freeman, and A. Willsky. Describing visual scenes using transformed dirichlet processes. In Y. Weiss, B. Schölkopf, and J. Platt, editors, *Advances in Neural Information Processing Systems 18*, pages 1297–1304. MIT Press, 2006.

[93] M. J. Swain and D. H. Ballard. Color indexing. *Int. J. Comput. Vision*, 7(1):11–32, 1991.

[94] Y. W. Teh, M. I. Jordan, M. J. Beal, and D. M. Blei. Hierarchical dirichlet processes. *Journal of the American Statistical Association*, 101(476):1566–1581, 2006.

[95] S. Tong and E. Chang. Support vector machine active learning for image retrieval. In *MULTIMEDIA '01: Proceedings of the Ninth ACM International Conference on Multimedia*, pages 107–118, 2001.

[96] A. Torralba, R. Fergus, and Y. Weiss. Small codes and large databases for recognition. In *IEEE CVPR*, 2008.

[97] R. H. van Leuken, L. Garcia, X. Olivares, and R. van Zwol. Visual diversification of image search results. In *18th International World Wide Web Conference*, pages 341–341, 2009.

[98] J. Verbeek and B. Triggs. Region classification with markov field aspect models. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2007.

[99] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. *Proc. CVPR*, 1:511–518, 2001.

[100] J. Vogel and B. Schiele. Natural scene retrieval based on a semantic modeling step. In *CIVR*, pages 207–215, 2004.

[101] G. Wang, Y. Zhang, and L. Fei-Fei. Using dependent regions for object categorization in a generative framework. In *CVPR '06: Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 1597–1604, 2006.

[102] J. Z. Wang, J. Li, and G. Wiederhold. SIMPLIcity: semantics-sensitive integrated matching for picture libraries. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23:947–963, 2001.

[103] X. Wei and W. B. Croft. LDA-based document models for ad-hoc retrieval. In *SIGIR '06: Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 178–185, 2006.

[104] G. V. D. Wouwer, P. Scheunders, and D. V. Dyck. Statistical texture characterization from discrete wavelet representations. *IEEE Transactions on Image Processing*, 8:592–598, 1999.

[105] S. Young. A review of large-vocabulary continuous-speech recognition. *IEEE Signal Processing Magazine*, 13(5):45–57, 1996.

[106] C. Zhai and J. Lafferty. A study of smoothing methods for language models applied to ad hoc information retrieval. In *SIGIR '01: Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 334–342, 2001.

[107] D. Zhang, A. Wong, M. Indrawan, and G. Lu. Content-based image retrieval using gabor texture features. In *IEEE Transactions PAMI*, pages 13–15, 2000.

[108] S. Zhou, Y. Rui, and T. Huang. *Exploration of Visual Data*. Kluwer Academic Publishers, 2003.

Curriculum Vitae

Eva Hörster

Date of birth:	November 11, 1979
Place of birth:	Solingen, Germany
Citizenship:	German

Education:	2005 - 2009	Doctoral student at the Multimedia Computing Lab, University of Augsburg, Germany.
	1999 - 2004	Studies of Electrical Engineering, Electronics and Information Technology at the University of Erlangen-Nuremberg, Germany. Graduation with the degree *Dipl.-Ing. Univ.* in December 2004.
	2003	Exchange semester at INSA Rennes, France.
	31.05.1999	Abitur, Erzbischöfliche Marienschule Opladen, Germany.

Professions:	since 2005	Research and Teaching Assistant at the Multimedia Computing Lab, University of Augsburg, Germany.
	2008	Internship at Yahoo! Research, USA.
	2004	Internship and Diploma Thesis at Intel Research, USA.
	2002 - 2004	Student Assistant in different research groups, University of Erlangen-Nuremberg, Germany.
	2002	Internship at Instituto de Tecnologia de Software, Brazil.
	1999	Internship at VITS, Germany.