BAŞKENT ÜNİVERSİTESİ

FEN BİLİMLERİ ENSTİTÜSÜ

VIDEO CONCEPT CLASSIFICATION AND RETRIEVAL

HİLAL ERGÜN

YÜKSEK LİSANS TEZİ

2016

VIDEO CONCEPT CLASSIFICATION AND RETRIEVAL

VİDEO KAVRAM SINIFLANDIRMA VE GERİ ERİŞİMİ

HİLAL ERGÜN

Başkent Üniversitesi
Lisansüstü Eğitim Öğretim ve Sınav Yönetmeliğinin
BİLGİSAYAR Mühendisliği Anabilim Dalı İçin Öngördüğü
YÜKSEK LİSANS TEZİ
olarak hazırlanmıştır.

2016

“Video Kavram Sınıflandırma ve Geri Erişimi” başlıklı bu çalışma, jürimiz tarafından, 28/01/2016 tarihinde, **BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI 'nda YÜKSEK LİSANS TEZİ** olarak kabul edilmiştir.

Başkan	: Prof. Dr. Adnan Yazıcı
Üye (Danışman)	: Yrd. Doç. Dr. Mustafa Sert
Üye	: Yrd. Doç. Dr. Emre Sümer

ONAY

..../..../2016

Prof. Dr. Emin AKATA

Fen Bilimleri Enstitüsü Müdürü

ACKNOWLEDGMENTS

I would like to express my deep gratitude to my master thesis advisor, Asst. Prof. Dr. Mustafa Sert, for the continuous support, helpful advice, for his motivation and endless patience. His guidance helped me in all the time of research and writing of this thesis.

I would like to thank my father, Ayhan Ergün. He never lost his faith in me and always supported me throughout my life. I am grateful my mother Hanife Ergün, for always giving me spiritual support. I am also indebted my brother, Ercan Ergün, his friendship, unconditional love and support. And also I would like to thank to my father, Faruk Akyüz for encouraging me and his valuable advice.

Finally, I would like to deeply thank my husband, Yusuf Çağlar Akyüz. He is always encouraging me throughout all my studies. And also he never let me to pass sleepless night alone. Without his patience and sacrifice, I could not have completed this thesis. I will always be grateful forever for your love.

This thesis partially supported by 114R082 numbered TUBITAK project.

ABSTRACT

VIDEO CONCEPT CLASSIFICATION AND RETRIEVAL

HİLAL ERGÜN
Başkent University Institute of Science and Engineering
Computer Engineering Department

Search and retrieval in video content is a trending topic in computer vision. Difficulties of this research topic is two folds; extracting semantic information from structure of video images is not a simple task and demanding nature of video content requires efficient algorithms. Semantic information extraction is challenged by researchers for more than two decades, yet new improvements are still welcome by the community. Recent burst of efficient computer hardware architectures has exploited both accuracy and complexity of many algorithms adding a new dimension to the efficient algorithm selection. In this thesis, our goal is to classify visual concepts in video data for content-based search and retrieval applications. To this end, we introduce a complete visual concept classification and retrieval system. We use two state-of-the-art methods, namely “Bag-of-Words” (BoW) and “Convolutional Neural Network” (CNN) architecture for visual concept classification. The performance of the classifiers is further improved by optimizing the processing pipeline steps. For retrieval, we provide concept- and content-based querying of video data and perform evaluations on Oxford Buildings and Paris datasets. Results show that, a substantial performance gain is possible by optimizing processing pipelines of the classifiers and deep learning based methods outperform the BoW.

KEYWORDS: Video Concept Classification, Bag-of-Words (BoW), Convolutional Neural Networks (CNNs), Content-Based Retrieval, SVM, Deep Learning.

Supervisor: Asst. Prof. Dr. Mustafa SERT, Başkent University, Computer Engineering Department.

ÖZ

VİDEO KAVRAM SINIFLANDIRMA VE GERİ ERİŞİMİ

HİLAL ERGÜN
Başkent Üniversitesi Fen Bilimleri Enstitüsü
Bilgisayar Mühendisliği Anabilim Dalı

Video içerikleri içerisinde arama ve geri getirme bilgisayarlı görme alanında yükselen bir konudur. Bu alandaki zorluklar iki başlık altında toplanabilir; video imgeleri içerisindeki anlamsal bilginin çıkarımı kolay bir iş değildir ve video içeriklerini analiz edebilmek için yüksek verimlilikteki algoritmalara ihtiyaç duyulmaktadır. Bu alanda çalışan araştırmacılar anlamsal bilginin çıkarılması konusuna 20 yılı aşkın bir süredir eğilmektedir ve bu alandaki iyileştirmelere hala ihtiyaç duyulmaktadır. Son yıllarda bilgisayar mimarilerinin verimliliğinde yaşanan artışlar hem algoritmaların başarımlarını hem de karmaşıklıklarını artırmıştır ki bu da efektif algoritma seçimine yeni bir boyut kazandırmaktadır. Bu tez çalışmasında, amacımız video verileri içindeki görsel kavramların arama ve geri getirme uygulamalarına yönelik sınıflandırılmasıdır. Bu amaç doğrultusunda görsel kavram sınıflandırma ve geri getirme bazlı bir sistem öneriyoruz. Günümüzde çokça tercih edilen iki görsel sınıflandırma yaklaşımını sistemimize entegre ediyoruz; "Kelime Kümesi" yaklaşımı ve "Evrişimsel Sinir Ağları" yaklaşımı. Buna ek olarak, kelime kümesi temsili ve evrişimsel sinir ağları aşamalarında optimizasyonlar yaparak, öğrenme algoritmalarının başarımlarını artırıyoruz. Geri getirme için kavram ve örnek tabanlı sorgulama yöntemlerinin gösterimini yapıyoruz ve literatürde en çok tercih edilen Oxford Buildings ve Paris veri kümeleri üzerinde sonuçlarımızı görselliyoruz. Sonuçlar gösteriyor ki, kelime kümesi temsili ve evrişimsel sinir ağları aşamalarında yapılan optimizasyonlar yüksek performans artışlarını olası kılmaktadır ve derin öğrenme tabanlı metodlar kelime kümesi yaklaşımından daha iyi sonuçlar vermektedir.

ANAHTAR SÖZCÜKLER: Video Kavram Sınıflandırma, Kelime Kümesi (BoW), Evrişimsel Sinir Ağları (CNN), İçerik Tabanlı Geri Erişim, SVM, Derin Öğrenme

Danışman: Yrd. Doç. Dr. Mustafa SERT, Başkent Üniversitesi, Bilgisayar Mühendisliği Bölümü.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

1 INTRODUCTION

1.1 Video Concept Classification

Videos are composed of images, and related images compose shots of a video clip. Related shots of a video clip constitutes video scenes and every scene carries a semantic concept. Semantic concept of a video scene is important for many tasks including, but not limited to video summarization, indexing, retrieval and search. Since annotation of video scenes manually is an expensive task, researchers constantly looking for automatic methods of annotating video data. We believe video concept classification is highly correlated to problem of recognizing semantic category of an image. Concept classification may be on the level of object recognition or scene recognition [68].

1.2 Problem Description

Extracting semantic information from multimedia content has long been a challenging research area. Advances in the various fields of computer science in the last decade are offering us various solution alternatives for tackling semantic gap phenomenon. Developments in the human vision system understanding is letting computer researchers to design better algorithms for image understanding. Both hand-crafted features and deep feature learning architectures are developing with a constant pace and every new day computer scientists apply those improvements to previously un-solved image understanding problems [30] [84] [77] [23] [1]. With the advances in computer hardware it is easier to train bigger machine learning systems, especially easier accessibility of parallel architectures is the driving force of those systems under the hood. Thanks to vast availability of online multimedia content, we have more data to train and evaluate our algorithms which increases transferability of our solutions as well their application areas.

On the other hand, our understanding of developed systems lack implementations. At a first glance, this may sound strange but it is more of a fact, unfortunately. For instance, since their introduction to image understanding domain in 2012, convolutional neural networks achieved state-of-the-art results almost in every field of computer vision, however, rationale behind this success is not understood completely yet. Moreover, our understanding of older feature encoding algorithms is still developing. This led us to investigate different aspects of image understanding in a systematic manner in this thesis. Our aim is to understand strong and weak points in the various points of image processing pipelines and try to get rid of weak spots with proper merging of different pipelines. Finally, we apply our findings to the area of video scene classification.

1.3 Definitions

The general structure of a digital video is depicted in Figure 1.1. A digital video sequence consists of a sequence of still images taken at a certain rate. A video clip may contain many frames. Fortunately, video is normally made of a number of logical units or segments, namely video shots. Video scenes, on the other hand may contain a number of contigious video shots. In this study, we define video scene having high level semantics that are extracted from video data and also referred to as concept.

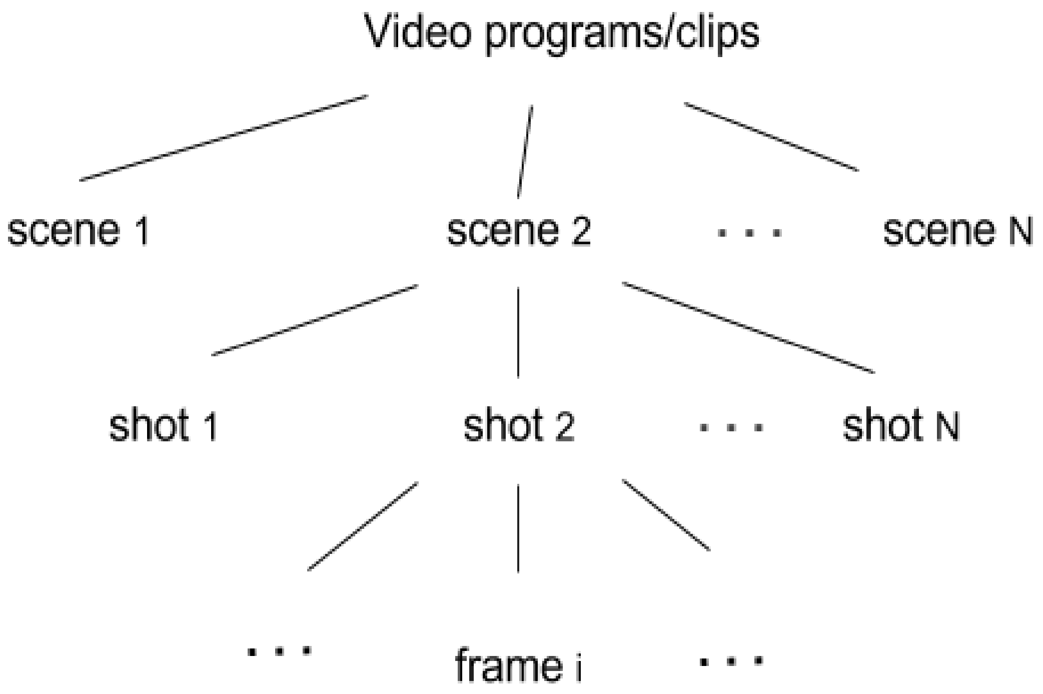


Figure 1.1 General structure of a video segment

1.4 Challenges

Automatic annotation of images is a challenging task because of “*semantic gap*” phenomenon which is the difference between understanding of visual concepts and image appearances. Overcoming semantic gap limitations is even harder in the presence of clutter, occlusion, camera shaking, viewpoint changes and various other noise sources inherent in images and videos [33]. Seperating a foreground object from its integrated background is much like a chicken-egg paradox, where classification of one implies identification of the other and vice versa. This further restrict us to use algorithms from vision areas other than background segmentation. In addition to visual challenges, heavy nature of image and video data causes demanding solutions in this line of field. Algorithms should be reaching optimum classification and recognition accuracy while keeping memory and computation resource usage at a minimum so that vasts amount of multimedia data can be better processed. With the introduction of big data into the picture, we believe requirements force use of more elegant solutions. For instance, hours of video content is uploaded to cloud services each day and even real-time performing applications may be classified as infeasible solutions.

1.5 Thesis Statement and Claims

In this thesis we assume that semantic content of a video scene can be anything; for instance it can be a very coarse classification like indoor and outdoor scenes, or it can be the determination of various human actions present in the videos, even it may be recognition of object categories. This forces us to discover relationships between low-level image features and high level semantics of a scene from a given dataset. Hence, we avoid crafting hand-made relationships or using rule-based approaches for the sake of generalizability.

In addition to mentioned assumption of concept classification, with this thesis we aim to show effectiveness of two state-of-the-art methods, namely the bag-of-words (BoW) based representations and deep learning for image and video concept classification and image retrieval. We show efficient vector quantization (VQ) encoding practices along recent deep learning techniques.

1.6 Organization

Organization of this thesis is as follows; Chapter 2 presents related work on our field of research. Chapter 3 presents details of our BoW framework as well as obtained empirical results. In Chapter 4, we describe our deep learning method and present its applications to visual concept classification. In Chapter 5, we investigate content-based image retrieval as well as concept-based retrieval. We conclude in the Chapter 6.

1.7 Original Contributions

In this thesis, we introduce a very efficient BoW processing pipeline and show how to optimize each different step. We also show how convolutional neural networks (CNNs) can be combined with BoW paradigm to further push classification accuracy. We present state-of-the-art results on various benchmark datasets. Specifically, the contributions of this thesis are three folds:

- We compare BoW and CNN architectures on common dataset and use identical processing pipelines.
- We analyze the effects of parameter selection to the overall classification accuracy.
- We also offer best practices for reaching state-of-the-art results on various benchmark datasets both in terms of computation efficiency and classification accuracy.

The work presented in this thesis has been published or submitted to following conferences:

- ❖ **Chapter 3** is published in Signal Processing and Communications Applications Conference in 2014.
- ❖ **Chapter 4** is submitted to The Second IEEE International Conference on Multimedia Big Data 2016.
- ❖ **Chapter 5** is published in Flexible Query Answering Systems Conference in 2015.

1.8 Software Libraries Used

Through out thesis study, we used various open-source libraries in our implementations. Our software packages use OpenCV [6] heavily, in many different places. We use Support Vector Machine (SVM) libraries provided by [7] and [18]. We use VLFeat both for feature mapping and extraction [73], as well Oxford VGG groups encoder evaluation software kit [8]. Our CNN implementations are using Caffe framework [26]. Last but not the least, we use Matlab more than often in our implementations [47].

2 LITERATURE REVIEW

2.1 Related Works on BoW Representation

Normally we would like to present a review of published studies under different sections for classification and retrieval as they are totally different problem domains. We still do this in the following sub-sections, however, we think it is more appropriate to start from common ancestors of both fields. Without such an introduction, we believe it is inevitable to loose some precise information about the history of these vision tasks. This is not only valid for classification and retrieval but also other fields of computer vision applications. For instance, both classification and retrieval algorithms find their early applications in texture recognition.

At this point we should emphasize that from a purely theoretical point of view, one can categorize vision applications into a wide selection of different topics with different requirements. There are many grifted categorizations present in the literature. Figure 2.1 shows results from our scan of existing literature and we believe this list is far from being complete. As it can be seen, many concepts are closely related to each other although solutions may be totally different. This grift relationship between different application areas forces us to treat these concepts as a single entity up to some point. We aggregate all these concept under the name of “Image understanding” wherever necessary throughout this study.

One of the very early object recognition approaches is to use direct pixel intensities or color at each pixel. This technique inherently assumes that images are cropped to region of interest and aligned in terms of pose and orientation which is not the case most of the time. Pixel intensity and color histograms can be viewed as global representations of an image [20]. Many object recognition studies employed parts-and-shapes models which uses spatial relationships between object components [85]. On the other hand, category wise object recognition or image classification enjoyed great success with the introduction of BoW paradigm which clearly become the choice of state-of-the-art methods in the literature.

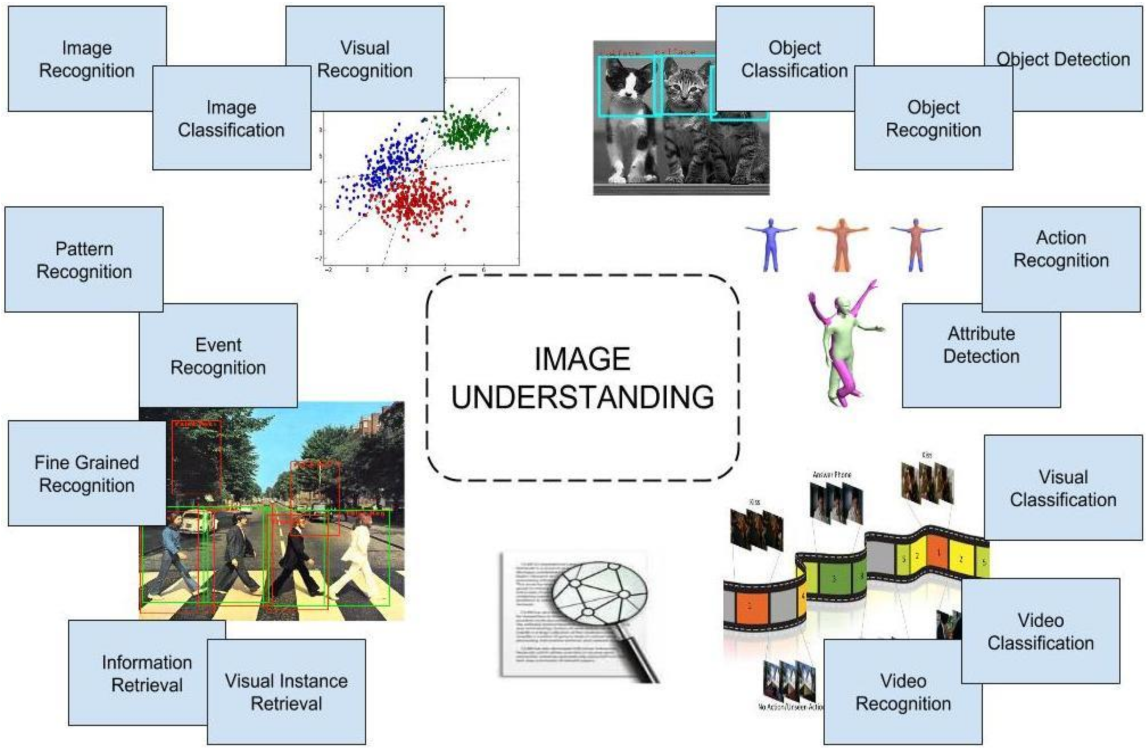


Figure 2.1 Image understanding

In 2004, CSurka et. al. first proposed BoW for the visual categorization or image classification [10]. However, this was not the first application of BoW paradigm to the image understanding. In 1999, Leung et. al. applied [38] BoW to texture recognition. They create a vocabulary of outputs of a filter bank using k-means clustering. In their study, they ignore spatial information and use histogram representation of image textons. In a later study, Cula et. al. used BoW descriptors again for texture recognition. Like Leung, they create texton libraries by k-means clustering a set of image features. In 2002, Varma and Zisserman improved Leung with the addition of rotationally invariant filters [72]. In 2002, Zhu et. al. [87] applied it to image retrieval. They create a dictionary using generalized Lloyd's algorithm which can be thought of a k-means variant. They use so called "keyblocks" of a given image instead of local image features. Naturally, their method is neither translation nor rotation invariant [10]. In 2003, Sivic et. al. applied BoW model to the video level object retrieval [63].

Following demonstration of success of BoW descriptors, many authors enhanced BoW representations for different image understanding tasks. Nister et. al. applied BoW representation to image retrieval from large selection of image dataset relatively using a higher number of dictionary dimension [49]. Laptev et. al. applied

BoW model to the video classification especially targeting human action categories [32]. Lazebnik et al. improved BoW classification by introducing spatial pyramids which incorporates lost spatial information into the BoW pipeline [33]. Yang et. al. replaced vector quantization step of BoW model with sparse coding and demonstrated state-of-the-art results on many image benchmark datasets [81]. In a similar study, Wang et. al. improved encoding performance of sparse coding using feature locality both in terms of performance and accuracy [76]. Van De Sande et. al. showed how color information can be inserted into BoW methodology [69]. Perronnin and Dance showed effectiveness of Fisher kernel (FK) encoding image category detection [51]. In order to improve quantization step of BoW, Philbin et al. introduced a method that uses soft assignment of image features to multiple visual codewords [54]. Gemert el. al. introduced a different way of soft assignment both addressing visual word plausibility and uncertainty [71] [70]. Jegou et al. introduced Hamming embedding for representing images with binary encodings [25].

2.2 Related Works on Deep Learning

In the recent years, CNNs have demonstrated excellent performance in plethora of computer vision applications. Most of this success is mainly attributed to two factors; recent availability of parallel processing architectures and larger image datasets [83] [62] [28]. Recent arrival of annotated and bigger sized datasets like ILSVRC [12], LabelMe [56] and MIT Places [86], made it possible to train deep convolutional networks for the task in hand without hitting the barrier of overfitting. Furthermore, accessibility of higher computational resources allowed researchers to design deeper networks with more parameters. For instance, top performer networks in ILSVRC challenge 2014 utilized more than 100 million parameters. Fusion of these two factors with better training algorithms [83] resulted in highly successful architectures performing state-of-the-art results in almost every field of computer vision.

One very important property of CNNs is their generalization ability. It is possible to apply a pre-trained network to a totally different dataset or application domain and achieve near state-of-the-art results. Moreover, with very little training effort they can beat most of other state-of-the-art approaches. This raises the question whether they can be applied to application domains where diversity of data is a challenge. Recent studies indeed show that this is perfectly possible [55] [13]. CNNs adapted to various image understanding tasks including but not limited to image classification, object detection, localization, human pose estimation, event detection, action recognition, face detection, traffic sign recognition, street number recognition and handwritten digit classification. They have gained much attention in recent years, albeit they were introduced in the late 80s [36]. Krizhevsky et. al. applied CNNs to the task of image classification and demonstrated excellent results [31]. Zeiler showed how CNN models can be further developed using visualization techniques [83]. In the following years deeper architectures surfaced further pushing classification accuracies. Winners of classification and localization tasks of ILSVRC 2014 challenge [12] employed deeper architectures [65] [62]. Sermanet et. al. showed that CNNs can be trained to fulfill multiple image tasks like classification and localization together [58]. Latest studies reveal that much deeper architectures are expected to surface in the very near future [23].

Availability of massively parallel GPU architectures is enabling researchers for trying plethora of configuration options. One of the most common techniques is the fusion of different approaches. Wang et al. demonstrated how multiple CNN architectures trained on different datasets can be employed for increasing event recognition accuracy in images. Guo and Gould applied fusion of different networks to the object detection [22]. Chatfield et al. showed ensemble of CNN architectures with Fisher vectors, albeit their gain was marginal.

Application of CNN architectures to video domain is also extensive. Karpathy et al. showed how CNN models can be enriched with temporal information present in videos using several fusion techniques [30]. Simonyan et al. improved general approach to video action recognition using two stream convolutional networks [61]. Ye and friends further analyzed two stream architectures for video classification [82]. Wang et al. outlined good practices for action recognition in videos [78]. Zha et al. applied CNNs along with shallow architectures to video classification [84].

3 VIDEO CONCEPT CLASSIFICATION USING BoW REPRESENTATION

3.1 Methodology

3.1.1 Overview of Bag of Words

Bag of words (BoW) model is originally a document classification methodology later applied to image classification in early 2000s and up to recently it has been the mostly established image classification method in the literature due to its ability to fill semantic gap phenomenon to some extent. In its very basic form, BoW creates histograms of local image features counting occurrence of image features in the given image. It is an orderless representation of those features which discards spatial information. While loss of inherent spatial information of image content seems like inappropriate, this gives BoW representation some sort of invariance to image layout.

For creating histograms of local patches or features, BoW model employs a family of dictionary coding algorithms. Extracted local features of image are encoded to a more suitable representation in this coding step for later operations. Standing on this descripton, BoW based image classification can be best described in three different distinct phases; dictionary creation, training and testing. We need a suitable visual dictionary for training and testing phases. For visual dictionary creation, variations of well-known k-means clustering algorithms are mostly employed. A corpus is generated from training features and clustered into desired number of visual codewords. We investigate this process in more detail in section 3.1.2.

After visual dictionary creation phase, we train a classifier for the task in hand. We begin by extracting local features of training images. Later we encode extracted features using vector quantization (VQ) using visual dictionary. After this step, we count appearance of each codeword in the training image which becomes our BoW descriptor for the given image. As a final step, we train a suitable classifier using training data and corresponding labels. In the testing phase, we almost follow the same steps of training phase. We first extract local features and encode extracted features. After creating image histograms and BoW descriptors, we predict final category of the given image using the previously generated classifier. All of the mentioned phases are constituted of some common processing steps which we name as the general BoW processing pipeline. This pipeline is composed of four operations; feature extraction, quantization, histogram generation and classification. These steps are visualized in Figure 3.1.

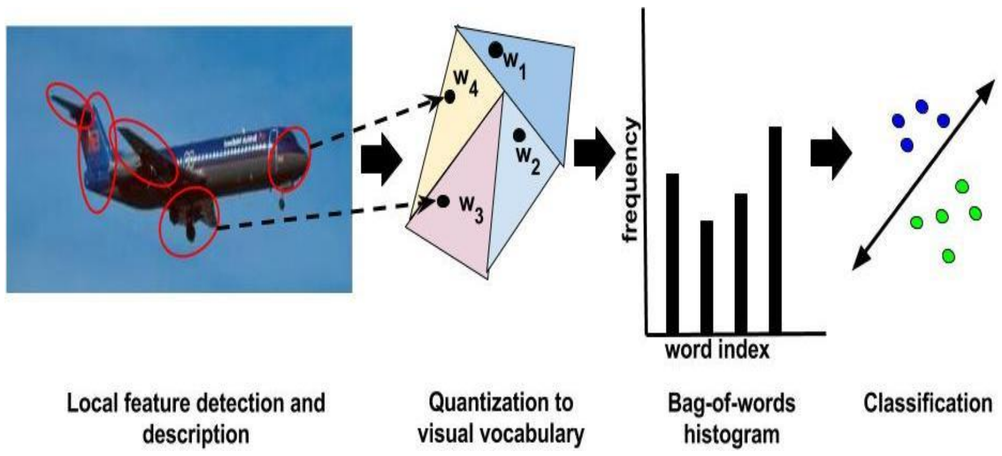

w1
w4
w2
w3
frequency
word index
Local feature detection and description
Quantization to visual vocabulary
Bag-of-words histogram
Classification

Figure 3.1 BoW pipeline

Several extensions exist to this paradigm in the literature. We investigate some of those extensions in the following sections but we need to mention some of them here for the sake of completeness. One of the possible extensions to classical BoW model is to use of spatial pyramids. In this approach, image is divided into smaller images and BoW model is independently applied to each smaller image. Later, these finer-grained image histograms are concatenated to create final image representation. Spatial pyramids provide a good cover for the spatial space and many state-of-the-art classification implementations use them directly or modified versions.

Some authors propose to overcome limitations VQ using better feature quantization techniques like sparse coding, Fisher encodings as well as employing soft assignment techniques instead of hard vector quantization. We further investigate various encoding schemes in section 3.1.4.

3.1.2 Visual dictionary creation

K-means clustering is by far the mostly employed technique for dictionary creation in the literature [75]. It is basically an unsupervised learning technique which finds desired number of cluster centers for a given data. It is an iterative algorithm, with each iteration new cluster centers are computed and each data point is assigned to their closest center. While it is possible to use many distance metrics, Euclidean (or L2 distances) is the most obvious choice.

Let’s assume we’re given set of ”N” data points where each sample is “D” dimensional:

$$S = \{x_1, x_2, x_3, x_4, ..., x_N\} \tag{3.1}$$

k-means aims to cluster these “N” data points into “K” clusters by minimizing the distance of each data point to their belonging cluster center:

$$J = \sum_{j}^{K} \sum_{i}^{N} ||x_i - c_j||^2 \tag{3.2}$$

After clustering operation, we export cluster centers for obtaining our so called visual dictionary composed of “K” visual codewords as modeled in Equation 3.3.

$$D(w) = (w_1, w_2, w_3, w_4, w_5 ... w_K) \tag{3.3}$$

Minimizing objection function of Equation 3.2 is NP hard in practical use cases [45] which is overcome by using heuristic approaches like Lloyd’s algorithm [43] or Elkan’s algorithm [14]. One of the shortcomings of standard k-means algorithm is that it has run-time complexity of $\mathcal{O}(NK)$. Thus, it becomes hard to reach stabilization when dictionary size and number of samples increases. One alternative is to use hierarchical k-means clustering (HKM) introduced by [49] which has a complexity of $O(NlogN)$ [75]. Another alternative is to use approximate k-means (AKM) algorithm of Philbin et. al. [53]. AKM has a run-time complexity of $O(MNlogN)$ where “M” is the number nearest clusters accessed. Use of more advanced algorithms is not the only way for creation of more efficient dictionary generation. One other technique is to sub-sample input feature samples. This operation not only reduces both memory constraints and running times for clustering, it also helps to eliminate outliers in the input data.

Although k-means is the most widely used method of dictionary generation [80], there are several alternatives have been investigated in the literature. Jurie et. al. explored a mean-shift algorithm to overcome k-means favoring of most frequent descriptors against rarely seen ones [29]. Some authors tried to come up with information theoretic semantic dictionaries using supervised learning methods [41] [34]. Tuytelaars et. al. divided feature space into regular grids instead clustering [67]. Wu et. al. [80] offered an alternative distance metric in place of Euclidean distance based on the histogram intersection kernel of Maji et. al [46]. Gaussian mixture models (GMMs) are also used to form dictionaries [50] [79].

3.1.3 Feature extraction

Feature extraction can be performed in two different approaches. One can use an interest point detector for detecting important points in image regions called keypoints. This scheme is called as sparse sampling as image features are extracted, relatively, from a set of sparse points. Various interest point detectors are present in the literature. Just to name a few we can say Laplacian of Gaussian, Difference of Gaussians, Harris detector, Hessian detector as well as their affine variant versions. Review of interest point detectors are beyond the scope of this study but additional information can be found in the literature [66] [57] [48]. Figure 3.2 visualizes a sparse sampling based feature detection.

Another alternative to interest point-based sampling is to dense sampling. In this scheme, features are extracted along a regular grid of points from the given image. Compared to sparse sampling, dense sampling produces much more image features. There is another alternative mentioned in the literature as random sampling where image features are extracted at randomly selected points from a given image. However, we believe this is only another form of dense sampling. Figure 3.3 visualizes a dense sampling based feature detection.

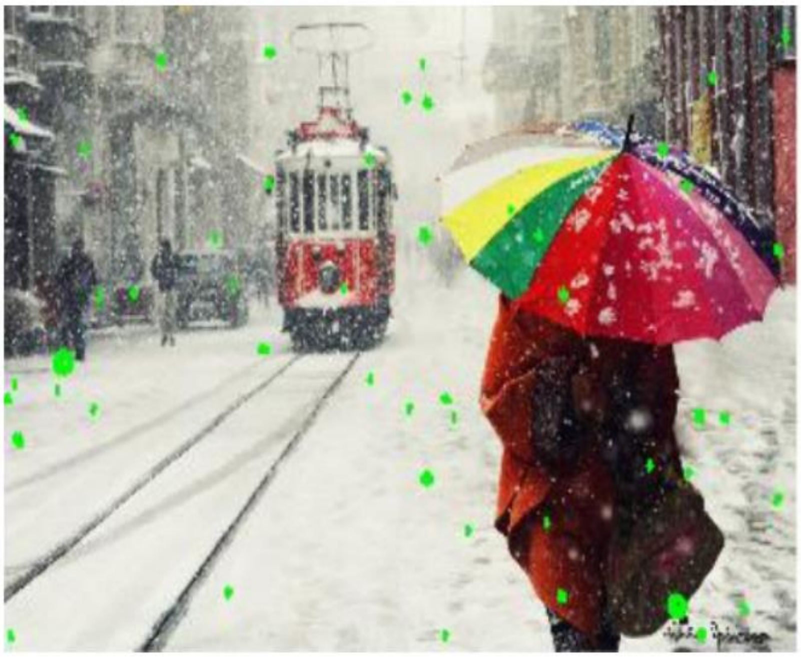


Figure 3.2 Feature extraction with Sparse Sampling

Many studies show the effectiveness of dense feature extraction over interest point based extraction on visual classification tasks [33], [27]. On the other hand, sparse sampling is employed in retrieval applications.

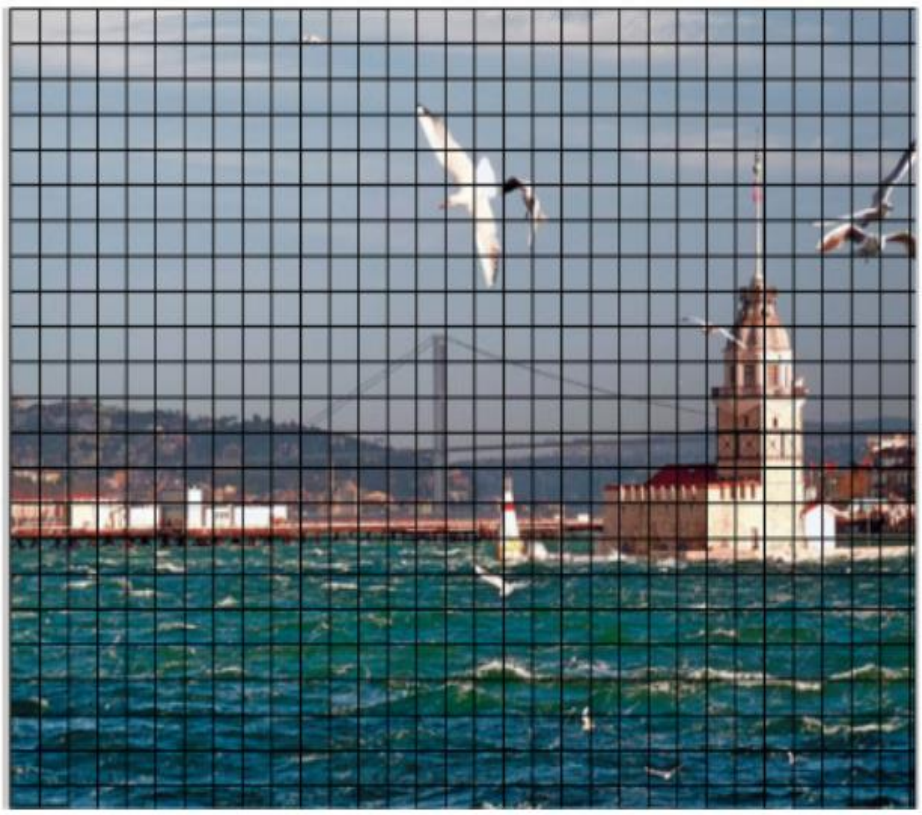


Figure 3.3 Feature extraction with Dense Sampling

Scale-Invariant Feature descriptor

Our local image feature of choice is scale-invariant feature transform (SIFT) of [44]. SIFT is an image descriptor for image-based matching and recognition developed by David Lowe (1999, 2004). SIFT algorithm contains two distinct operations, keypoint detection and descriptor extraction. SIFT searches a given image for keypoints in a multi-scale fashion. Each detected keypoint has various attributes like position in scale space and orientation. SIFT is a said to detect 'blobs' in an image in contrast to edges. It uses Difference of Gaussian's (DoG's) image detector under the hood. SIFT descriptor, on the other hand, is a 3-D spatial histogram of the image gradients around a given keypoint, which is visualized in Figure 3.4. Due to its construction principles, SIFT image descriptor is invariant to translations, rotations and scaling transformations in the image space. It is also invariant, up to some degree, to moderate perspective transformations and illumination variations. Many studies in the literature shows the excellence of SIFT feature for image matching, object recognition and retrieval tasks.

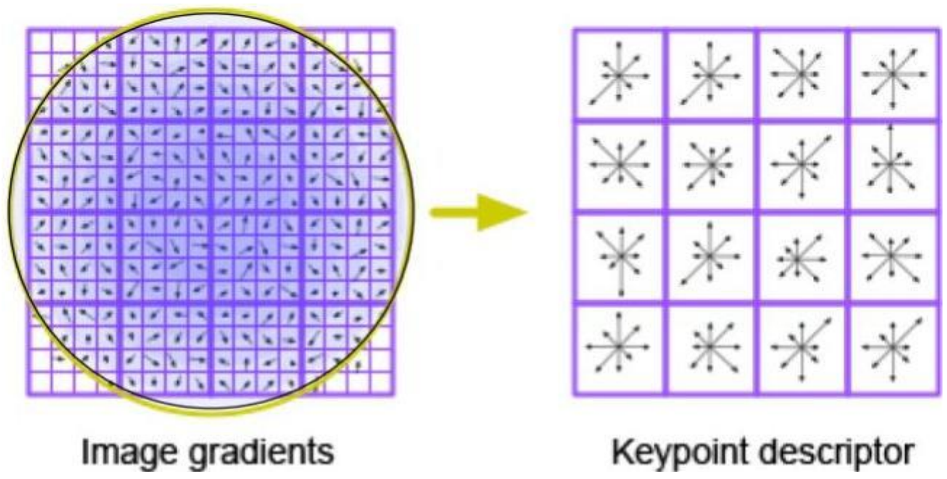


Figure 3.4 SIFT feature descriptor representation

3.1.4 Feature encoding

Feature encoding is the process of re-coding local features to a more suitable notation for classification. Encoding doesn't alter the local nature of the feature as it is pair-wise transform operation most of the time. There are many different encoding types present in the literature used with BoW. Hard vector quantization is one of the frequently used encodings which compares each feature with codewords in dictionary and assigns to the closest one. Soft assignment, improves this process by assigning to 'k' nearest neighbours instead of one. Sparse coding is based on the relaxation of sparsity condition in vector quantization. Fisher vectors and super-vector encodings uses Gaussian Mixture Models and try to capture high-level relations of local features.

Now assuming we have our visual dictionary modeled by Equation 3.2, we extract our local image features in a dense fashion to obtain our image features:

$$S(w) = (s_1, s_2, s_3, s_4, s_5 ... s_M) \tag{3.4}$$

where each s_i is a "D" dimensional feature vector. We note that "D" is 128 for SIFT descriptors. Standard BoW model assigns every feature to the closest entry in our codebook, or dictionary:

$$S'(w) = \frac{1}{M} \sum_{i}^{M} \begin{cases} 1 & \text{if w} = argmin(d(w,s)) \quad \forall w \in D(w) \\ 0 & otherwise \end{cases} \tag{3.5}$$

Here *d(.)* represents a similarity measure and we use Euclidean distance to measure closeness of two vectors. This operation often is called a quantization operation since it discretizes a feature vectors. Equation 3.5 is also known as vector quantization and one of the most frequently used feature encodings, sometimes it is called hard assignment. Careful readers should notice that *S'(ω)* is an extremely sparse vector, i.e. only one entry of it is different than zero. *1/M* term is present for normalization purposes, it cancels out the effect of different cardinalities of unevenly scaled images. Figure 3.5 visualizes this operation.

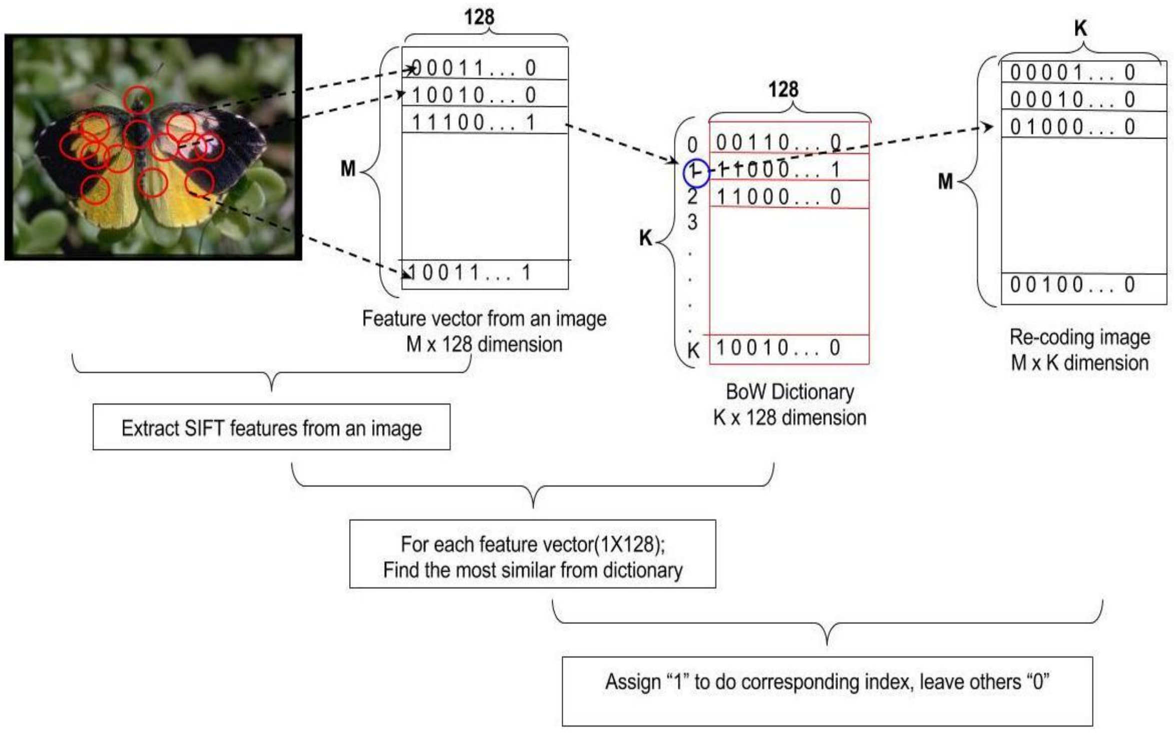


Figure 3.5 Vector coding visualization

Soft assignment is the process of quantizing local features in a soft manner, i.e. every feature contributes to more than one codeword in the dictionary. Soft assignment can modeled like:

$$\mathrm{UNC}(w) = \frac{1}{M} \sum_{i=1}^{n} \frac{K_\sigma\left(d\left(w, s_i\right)\right)}{\sum_{j=1}^{|V|} K_\sigma\left(d(w_j, s_i)\right)} \tag{3.6}$$

According to Gemert et. al. [71], this type soft assignment deals with codeword uncertainty which indicates that one local feature may distribute probability mass to more than one codeword. With increasing size of dictionary size, soft assignment converges to hard assignment and soft assignment may help to improve accuracy for relatively small dictionary sizes. However, soft assignments gains can be marginal depending on the pipeline parameters and may not worth the waste of additional computational power. We call this type of coding scheme as KCB following the original author's notation from now on.

Sparse coding is another alternative to vector quantization and poses some resemblance to soft assignment coding. It is based on the relaxation of sparsity condition in Equation 3.5. It encodes a local feature with a linear combination of a sparse set of basis vectors [42]. Sparse coding can be represented as in Equation 3.7:

$$u_i = \arg\min_{u \in R^n} \|x_i - \mathbf{B}u\|_2^2 + \lambda \|u\|_1. \quad (3.7)$$

Wang et. al. [76] introduced feature space locality into sparse coding and further improved its computational efficiency. Their implementation is known as locality constrained linear coding, or LLC, in the literature. In addition to better representation of local image features, sparse coding encodings works well with linear classifiers which we mention about their importance more in detail in section 3.1.6. According to Yang et. al., this success can be mainly attributed to the much less quantization errors of sparse coding compared to vector quantization [81].

Fisher vector encoding uses Gaussian Mixture Model's to learn higher order statistics of distributions of local features. Unlike BoW, it is not restricted by the occurrences of each visual word [52]. It can be tought of an extension of BoW as well as an soft encoding approach.

3.1.5 Spatial Pyramids

As we noted previously, BoW approach discards spatial layout of image features. Although this brings invariance to pose and geometric changes, to some degree, it also looses previous spatial information. Lazebnik et. al [33] introduced spatial pyramids method to take into account of rough scene geometry [52]. Spatial pyramids starts by dividing image into smaller grids which can either be regular or some irregular patterns. Then it calculates BoW descriptor for each sub-region of image. Grids can be created in multiple layers where going down the layers causing more fine-grained grid layouts. In the final representation, all histograms coming from all regions of all layers contribute to the final BoW descriptor, with some scheme of weight assignment taking in place. We re-depicted a well known visualization of spatial pyramids in Figure 3.6 where three different types of local features extracted from an image is shown. Level-0 refers to the whole image without any spatial partitioning. When we pass down to the Level-1, we create four different sub-regions of the image and calculate all corresponding histograms. We repeat same procedure with Level-2, but this time with a finer partitioning. Final image representation can be formed by concatenating all histograms into one feature vector. We should emphasize that Level-0 representation belongs to the BoW representation without any spatial pyramid extension.

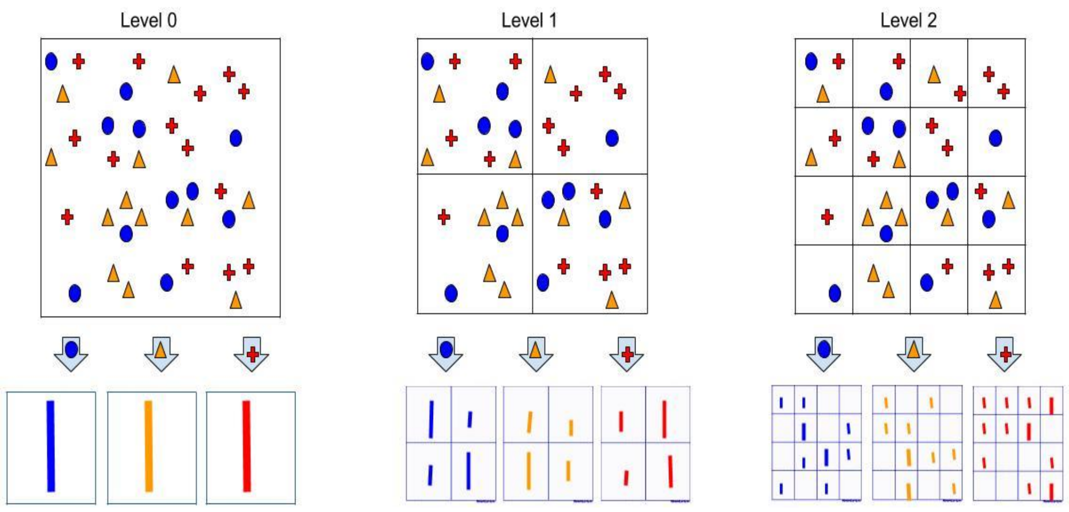


Figure 3.6 Spatial Pyramid visualization

One handicap of SPM extension is that it exploits feature dimensionality. Taking Level-2 into account, one ends up 21 (16 + 4 + 1) image regions. For Level-3, which is not depicted in Figure 3.6, this number rises to 85. Given that dictionary size is “K”, this results in final image feature size of 21 x K for Level-2. Although final descriptor size is bloated, SPM calculation is very efficient. Regardless of SPM level, nearest neighbours of each feature is searched only once due to the fact that location of a local feature in feature space is independent of its location in spatial space. With efficient programming practices, overhead of SPM generation is one of the fastest operations in whole processing pipeline.

We should mention that SPM layout depicted is here taken from original authors’ [33] papers. Later studies also investigated different layouts like horizontal and vertical binning. For different datasets where object layouts differ significantly it may be more efficient to adapt such layouts. For instance, instead of dividing images into 16 square regions one may employ 3x1 horizontal binning or 4x1 vertical binning. Figure 3.7 visualizes different binning strategies.

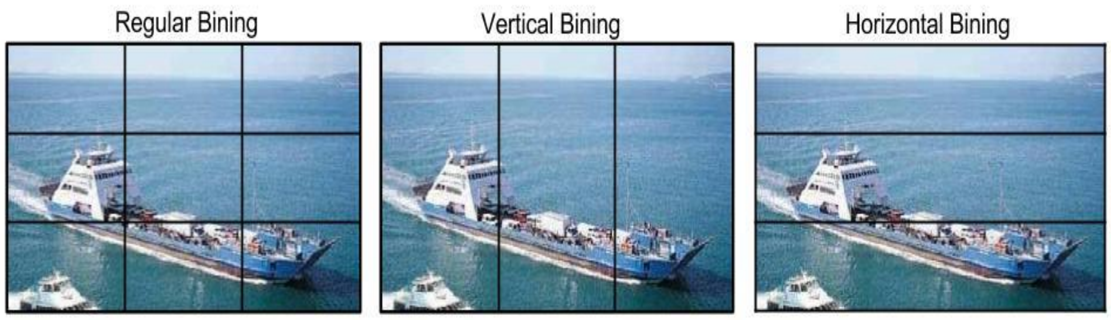


Figure 3.7 Spatial Pyramid binning visualization

3.1.6 Classifier design

Machine learning is the art of learning structure from data and classifiers are the key components for linking data to its structure. In this regard, Support Vector Machines (SVMs) are among the most successful implementations. They were introduced in early 90s [5] [9] and gained much attention since then. They are designed as binary classifiers, several extensions exist for multi-class classification. SVMs are linear classifiers but they can be extended to non-linear cases by the use of a so called "kernel trick".

Although non-linear SVMs are quite successful in many visual classification tasks, their performance comes at a high cost. Training of non-linear SVMs has a runtime complexity between $\mathcal{O}(N^2)$ and $\mathcal{O}(N^3)$ whereas linear SVMs have a complexity of $\mathcal{O}(N)$, where "N" is the number of training images [60] [27] [81]. This makes use of non-linear SVMs impractical in case we have more than tens of thousands training data. On the other hand, linear SVMs are inferior to non-linear ones on many image processing tasks [52]. To overcome this drawback, many researchers seek for better encoders suitable for large-scale image and video classification tasks. Sparse coding and Fisher Vector encodings are just fruits of two of the many such research directions. We head towards a different route though. We choose to apply kernel mappings to our data prior to classification using homogeneous kernel maps [74]. These algorithms scale linearly with respect to training data while providing the same accuracy as the original non-linear SVM classifiers [52]. They are implemented for histogram intersection and X^2 kernels which are the two most successful kernel types for our encoding operations [35] [59].

3.2 Datasets In Use

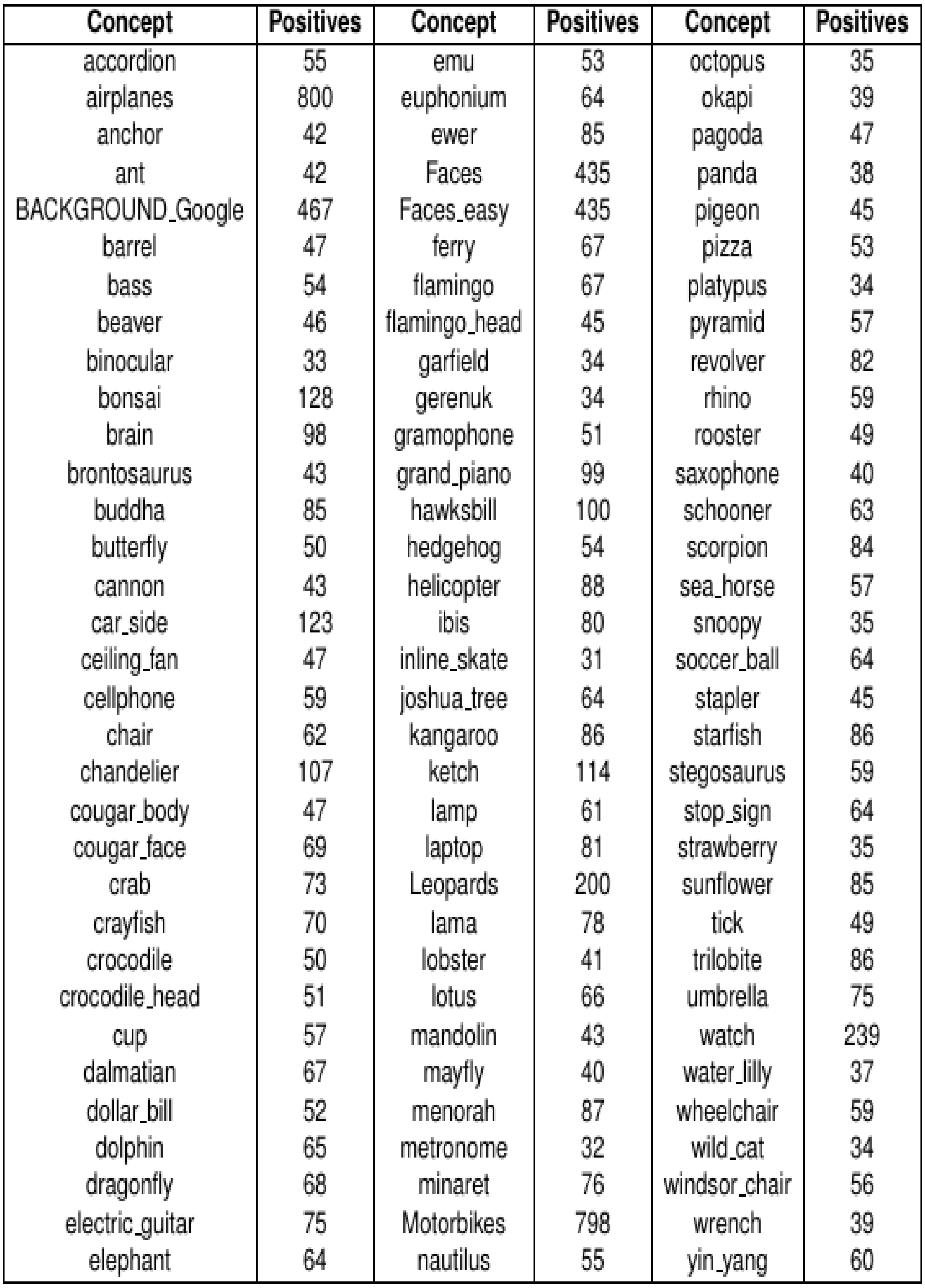
3.2.1 Caltech-101 dataset

Caltech-101 dataset [19] contains 101 categories of different objects and one background which we use as a normal category in our experiments. Caltech-101 contains a total of 9,144 images and has 31 to 800 images per category. Moreover, most of the categories have more than 50 images. Average image resolution is around 300x200 pixels and each image contains only a single object. Caltech-101 is one of the most widely adapted datasets by computer vision community, for object recognition and classification tasks; as of this writing it has more than 800 citations in the literature making it one of the most important benchmark datasets for object recognition. Figure 3.8 shows some samples images taken from this dataset and summary of concepts present in the dataset is listed in Table 3.1.

Figure 3.8 Some examples from Caltech-101 dataset

Caltech-101 doesn’t provide any image lists for train and test case splitting. Common practice used in the literature is to randomly split dataset into two and then using one for training and the other one for testing. To prevent favorization effects of random splitting, results are reported on at least three different splits. Cross validation is not utilized at all. Most frequently used split regime is to use 30 train samples and rest for testing. Since results are presented using class-wise mean accuracy, un-even image sizes selected in testing phase is cancelled out.

Table 3.1 Summarization of concepts for Caltech-101 dataset



Concept	Positives	Concept	Positives	Concept	Positives
accordion	55	emu	53	octopus	35
airplanes	800	euphonium	64	okapi	39
anchor	42	ewer	85	pagoda	47
ant	42	Faces	435	panda	38
BACKGROUND_Google	467	Faces_easy	435	pigeon	45
barrel	47	ferry	67	pizza	53
bass	54	flamingo	67	platypus	34
beaver	46	flamingo_head	45	pyramid	57
binocular	33	garfield	34	revolver	82
bonsai	128	gerenuk	34	rhino	59
brain	98	gramophone	51	rooster	49
brontosaurus	43	grand_piano	99	saxophone	40
buddha	85	hawksbill	100	schooner	63
butterfly	50	hedgehog	54	scorpion	84
cannon	43	helicopter	88	sea_horse	57
car_side	123	ibis	80	snoopy	35
ceiling_fan	47	inline_skate	31	soccer_ball	64
cellphone	59	joshua_tree	64	stapler	45
chair	62	kangaroo	86	starfish	86
chandelier	107	ketch	114	stegosaurus	59
cougar_body	47	lamp	61	stop_sign	64
cougar_face	69	laptop	81	strawberry	35
crab	73	Leopards	200	sunflower	85
crayfish	70	lama	78	tick	49
crocodile	50	lobster	41	trilobite	86
crocodile_head	51	lotus	66	umbrella	75
cup	57	mandolin	43	watch	239
dalmatian	67	mayfly	40	water_lilly	37
dollar_bill	52	menorah	87	wheelchair	59
dolphin	65	metronome	32	wild_cat	34
dragonfly	68	minaret	76	windsor_chair	56
electric_guitar	75	Motorbikes	798	wrench	39
elephant	64	nautilus	55	yin_yang	60

3.2.2 Caltech-256 dataset

Caltech-256 dataset [21] is an extension of Caltech-101 and it contains 256 object categories spanning total of 30607 images. Caltech-256 has the same structure as

Caltech-101 but with a better image distribution. When compared to Caltech-101, minimum number of images in any category is increased from 31 to 80. Like Caltech-101, Caltech-256 is also associated with object categorization and recognition and is cited by more than 1100 papers in the literature. Figure 3.9 shows some samples images taken from this dataset.

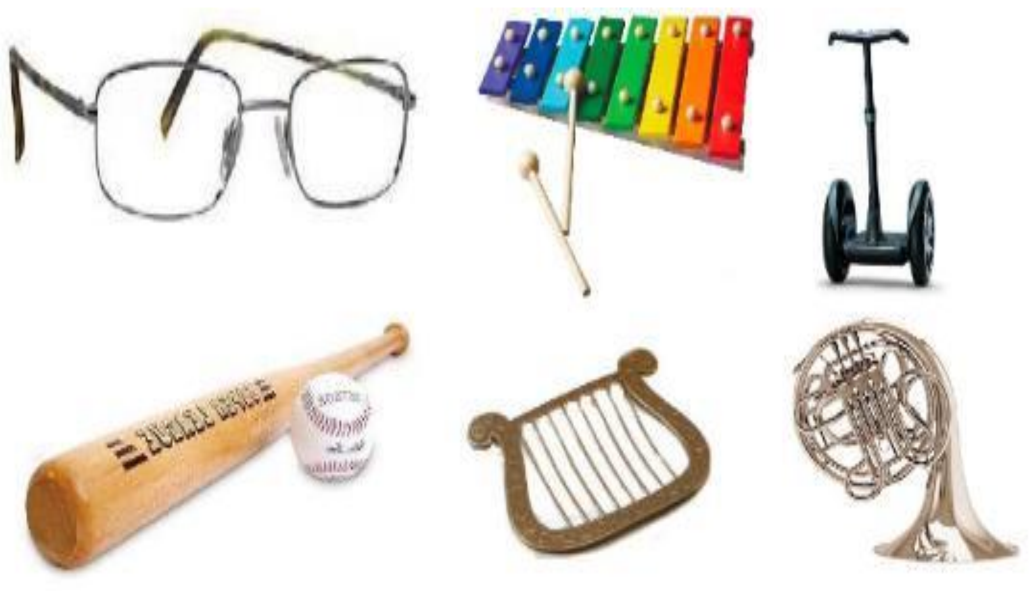


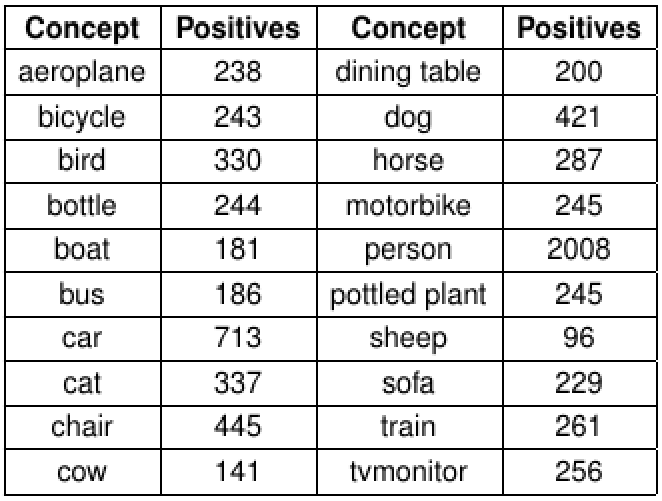
Figure 3.9 Some examples from Caltech-256 dataset

Validation procedure is much like Caltech101, only common choice for train split is to use 60 training images and rest for testing.

3.2.3 Pascal VOC 2007 dataset

The Pascal VOC 2007 [17] dataset is another benchmark dataset for image classification and localization. Pascal VOC 2007 contains 20 classes and a total of 9,963 images and 24,640 annotated objects. It is the predecessor of ILSVRC challenge. It has been cited more than 2800 papers in the literature. It is a harder dataset when compared to Caltech variations although has much less object classes. This is due to clutter and occulasion in the images as well as objects' low spatial spanning in the training images. Another difference from Caltech datasets is that Pascal VOC supplies a training list so everyone uses same training and testing data for presenting their results. Pascal VOC also uses mean average-precision (mAP) measure instead of a simple accuracy measure used in Caltech datasets. Once again we present dataset structure in Table 3.2.

Table 3.2 Summarization of concepts for Pascal VOC 2007 dataset



Concept	Positives	Concept	Positives
aeroplane	238	dining table	200
bicycle	243	dog	421
bird	330	horse	287
bottle	244	motorbike	245
boat	181	person	2008
bus	186	pottled plant	245
car	713	sheep	96
cat	337	sofa	229
chair	445	train	261
cow	141	tvmonitor	256

3.3 Results

In this section, we investigate various aspects of BoW methodology. We start with effects of dictionary size on classification performance. We compare performance of different dictionary creation techniques on Caltech-101 dataset mentioned in section 3.1.2. We evaluate hierarchical clustering (HKM) with a corpus size of six-hundered thousands and three millions and normal k-means clustering with six-hundered thousands and we choose the most simple encoding scheme, vector coding without any extensions. However, we choose x^2 SVM for best classification accuracy. Our results are depicted in Figure 3.10 which uses a logarithmic scale to better represent diverse cluster range. We see that HKM yields almost same results with full k-means clustering which only has in edge over HKM for very small cluster sizes. With the increase of target dictionary sizes, HKM starts grasping the representation of corpus as well as full k-means. This tells us to favor HKM in place of full k-means in our classification pipelines. We once again emphasize that HKM has a run-time complexity of $\mathcal{O}(NlogN)$ and when compared to full k-means with complexity $\mathcal{O}(NK)$, it is a more suitable choice for large datasets.

Our second observation regarding Figure 3.10 is that increasing dictionary size effectively increases classification accuracy. This is especially true with higher dictionary sizes which can be marked as debatable deduction given the different observations present in the literature. For instance, Lazebnik et. al. shows that there is a limiting effect of increasing dictionary size beyond a certain limit [33]. This is in par with our previous studies presented in SIU 2014 [59]. We should note that in both studies full k-means clustering is the choice dictionary generation algorithm. On the other hand, Chatfield et. al. [8] argues that increasing dictionary size always help to increase classification accuracy provided that feature encoding is not too lossy. We know that they use approximate k-means clustering (AKM) which is similar to HKM in terms of performance. These results suggests us to conclude that advanced clustering techniques like HKM and AKM are better at discriminating visual corpus compared to standard k-means algorithm even for big dictionary sizes. Given that they also have lower complexity than standard k-means algorithm they should be favored wherever possible.

Again [49] shows that using a larger corpus helps to cluster discriminatively better dictionaries when using large vocabularies. We observe same effect in our experiments. When using HKM clustering with more than three million feature corpus size, we observe higher classification accuracies than the previous six-hundered thousands case. Due to higher complexity of standard k-means algorithm, it is not feasible to increase corpus size for it above certain point. Given that HKM is both effective and discriminative with large a corpus, it is beneficial to increase corpus size up to a point where memory requirements is the limiting factor.

Next we investigate effects of different spatial pyramid level extensions on Caltech-101 dataset. We look into case of Spatial Pyramid (SPM) extension before investigating different feature encodings because SPM can be implemented regardless of the encoding choice. For SPM experiments we use VQ with HKM clustering for dictionary generation and same x^2 SVM as in the previous dictionary experiments. Figure 3.11 shows our results and Table 3.3 summarizes our numerical findings.

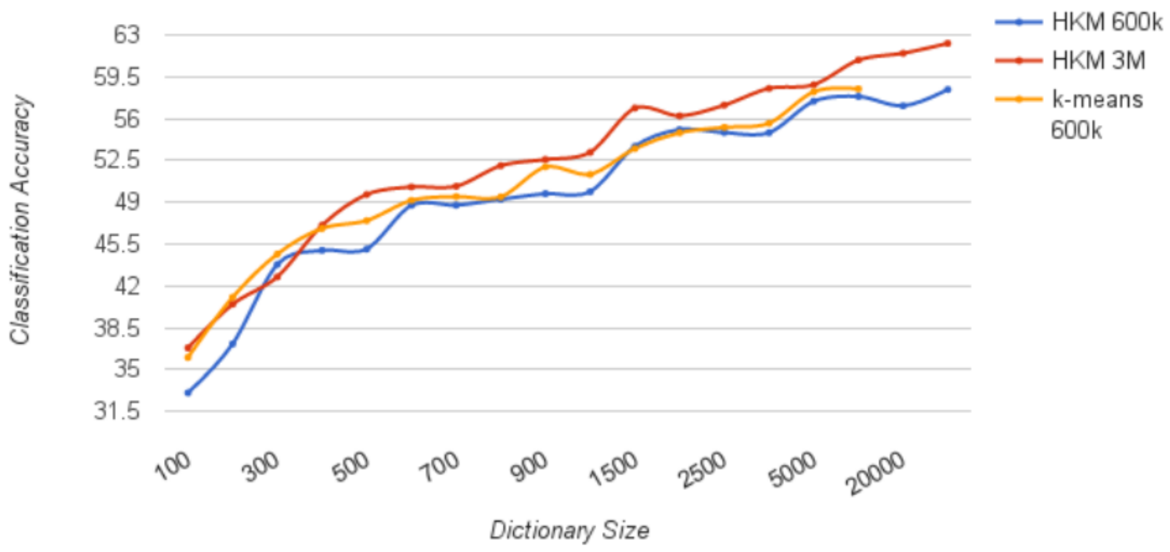


Figure 3.10 Classification accuracy with different dictionaries on Caltech-101

We observe that effects of SPMs is most dramatical with lower dictionary sizes. This implies that even if our visual dictionary is not discriminative enough to properly classify our dataset images, spatial information incorporated with SPMs adds up to classification accuracy. This adds some degree of robustness to our classification pipeline with respect to inadequate dictionary choices. Our second observation is that while using SPMs, increasing dictionary size beyond a certain limits results in almost no accuracy increase. For example taking two cases from Table 3.3, with L2 SPMs, 1000k dictionary size is only 1% worse than 5000k dictionary size.

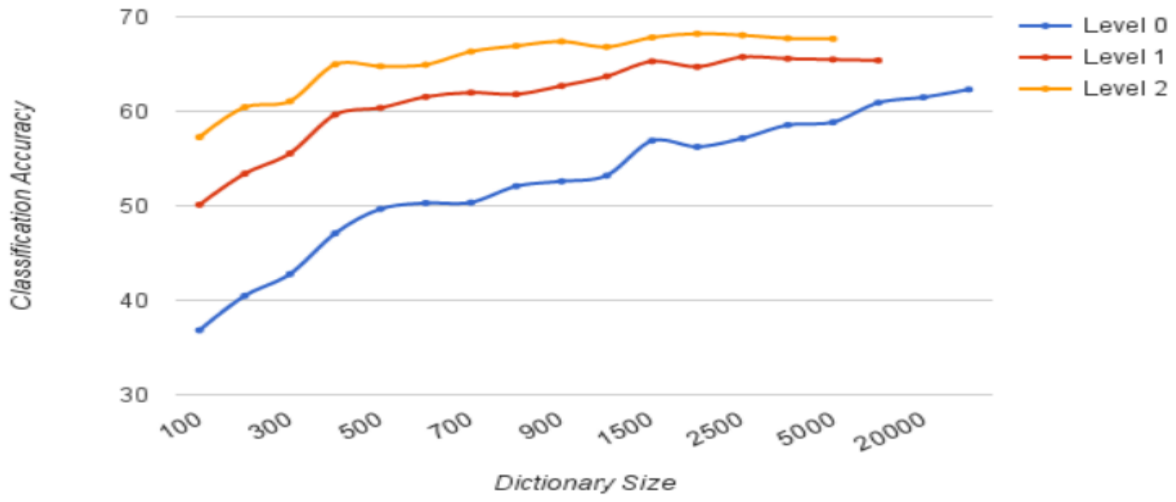
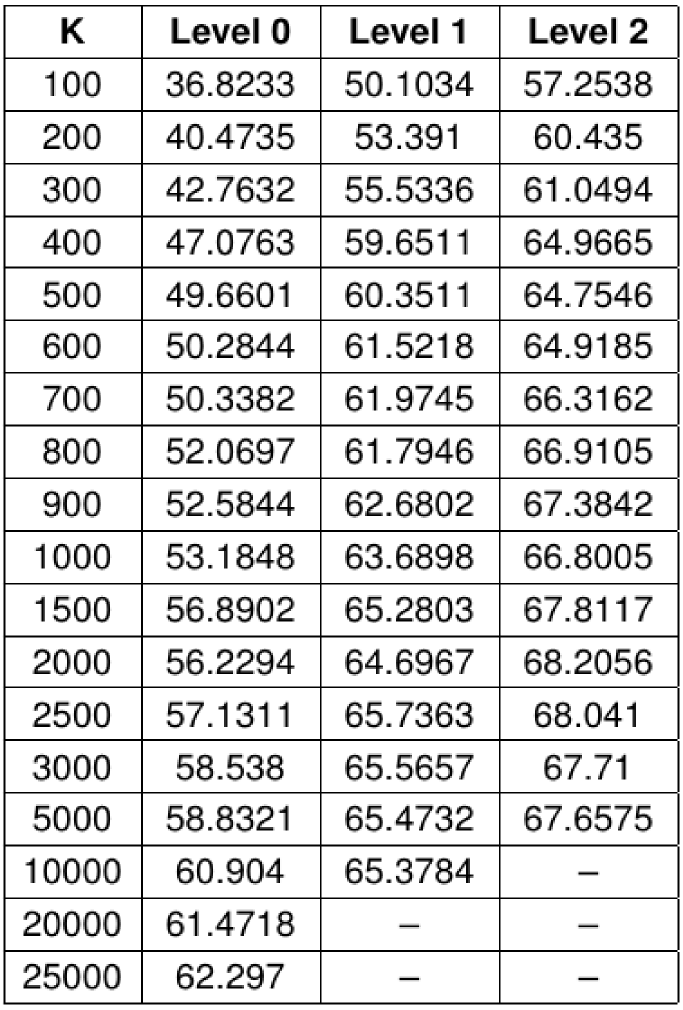


Figure 3.11 Classification accuracy with different dictionaries and pyramid levels on Caltech-101

Table 3.3 Effects of SPM with different dictionaries sizes on Caltech-101



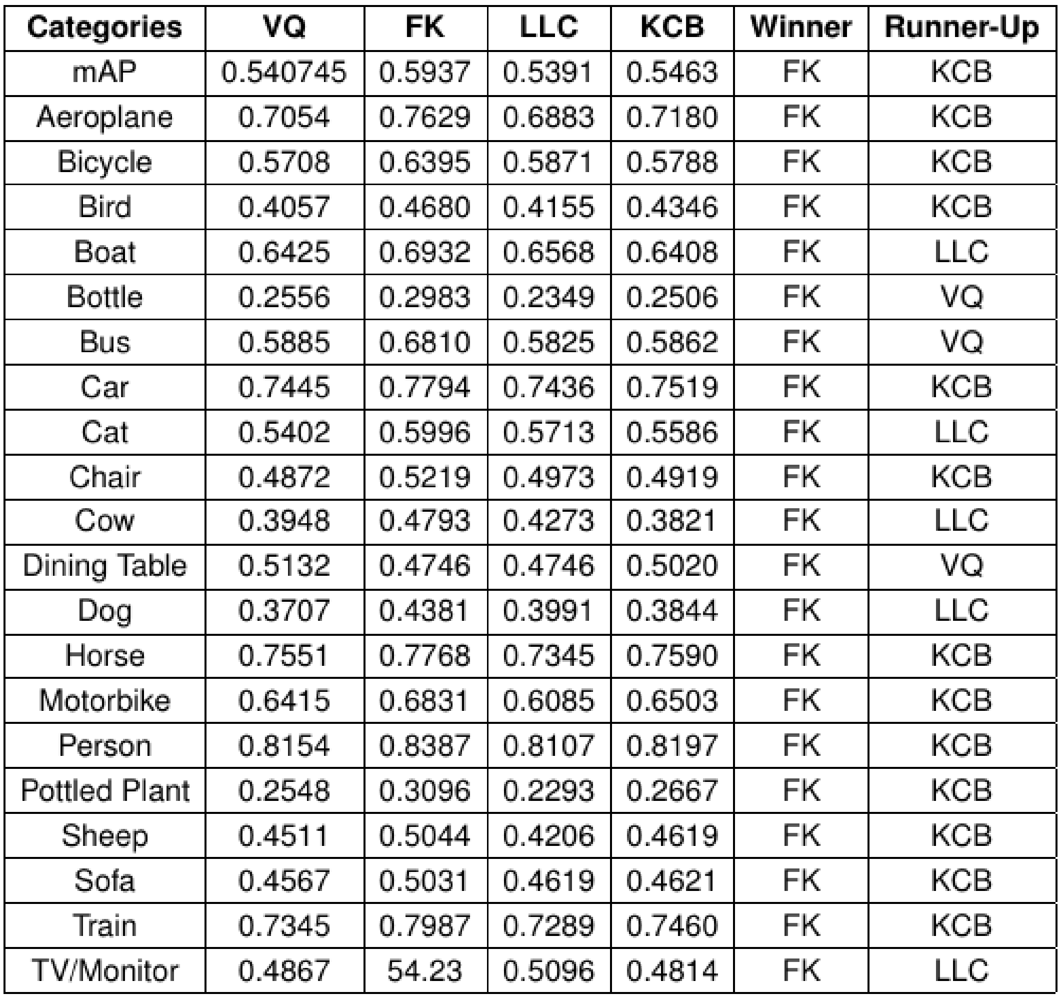
K	Level 0	Level 1	Level 2
100	36.8233	50.1034	57.2538
200	40.4735	53.391	60.435
300	42.7632	55.5336	61.0494
400	47.0763	59.6511	64.9665
500	49.6601	60.3511	64.7546
600	50.2844	61.5218	64.9185
700	50.3382	61.9745	66.3162
800	52.0697	61.7946	66.9105
900	52.5844	62.6802	67.3842
1000	53.1848	63.6898	66.8005
1500	56.8902	65.2803	67.8117
2000	56.2294	64.6967	68.2056
2500	57.1311	65.7363	68.041
3000	58.538	65.5657	67.71
5000	58.8321	65.4732	67.6575
10000	60.904	65.3784	–
20000	61.4718	–	–
25000	62.297	–	–

Next we investigate performances of different encoding types. This time we investigate feature encodings on more challenging Pascal VOC 2007 dataset. We fix dictionary size for all encodings as 4000k except for Fisher vector coding which we set-up to use 256 mixture models. For all encodings we fix feature extraction steps as pixel wide, both for horizontal and vertical resolutions. We extract SIFT features at multiple scales, also known as PHOW features [4]. We mix L1 spatial pyramids with three horizontal regions, totaling eight spatial regions. We use x^2 SVM for VQ and KCB encodings since they are reported to be work better with those SVM types [8]. We also verified this in our SVM experiments. For FK and LLC encodings we use linear SVMs as those encodings get along with linear SVMs and don't need non-linear kernels. For all our experiments, we select our SVM parameters either using grid search with 5-fold cross-validation in case of multiple parameters for non-linear SVM or using simple train-test validation for linear SVMs.

Table 3.4 lists our results. First thing striking us is the fact that VQ is reaching competitive results when compared to more advanced encodings, FK being the only exception. There are superior results for both LLC and KCB encodings when compared VQ encoding in the literature, however our results clearly states that this is not the case. Furthermore, there are other studies showing parallel results to our findings [8]. This can be attributed to various different phenomenon. First of all, unlike many studies, we follow a very aggressive feature extraction procedure. Both dense feature extraction step and images scales are used at limits. We believe this pushes performance of VQ further up whereas LLC and KCB is not taking advantage of denser feature extraction since they are already capable of coding image features at their optimum level. Another factor adding to this conclusion is that dictionary size can be increased further to make use of effectiveness of LLC and KCB encodings, as we believe they may further increase their performance with higher dictionary sizes. However, this increase is not be dramatic. For instance, [8] tested LLC with 25k dictionary sizes and they report 57.60% mAP on Pascal VOC 2007.

Our second observation is that none of these encodings are complementary. In Table 3.4 we listed winners and runners-up on a per-category basis. We see that FK encoding is the winner of all categories, and KCB is the runner-up in most of the categories. This makes us believe that success of any encoding is not dependant on any category, i.e. they get as discriminative as they can regardless of scene content. One other fact backing-up this theory is that categories with higher number of positive samples are always classified with higher accuracy. For instance, when we look into Table 3.2 we see that “Person” category has the highest number of positive samples and it is the category with highest AP in our results. This leads us to conclude that fusing different encodings types may not be a good candidate for multi-model image classification.

Table 3.4 Pascal VOC 2007 results for different encodings



Categories	VQ	FK	LLC	KCB	Winner	Runner-Up
mAP	0.540745	0.5937	0.5391	0.5463	FK	KCB
Aeroplane	0.7054	0.7629	0.6883	0.7180	FK	KCB
Bicycle	0.5708	0.6395	0.5871	0.5788	FK	KCB
Bird	0.4057	0.4680	0.4155	0.4346	FK	KCB
Boat	0.6425	0.6932	0.6568	0.6408	FK	LLC
Bottle	0.2556	0.2983	0.2349	0.2506	FK	VQ
Bus	0.5885	0.6810	0.5825	0.5862	FK	VQ
Car	0.7445	0.7794	0.7436	0.7519	FK	KCB
Cat	0.5402	0.5996	0.5713	0.5586	FK	LLC
Chair	0.4872	0.5219	0.4973	0.4919	FK	KCB
Cow	0.3948	0.4793	0.4273	0.3821	FK	LLC
Dining Table	0.5132	0.4746	0.4746	0.5020	FK	VQ
Dog	0.3707	0.4381	0.3991	0.3844	FK	LLC
Horse	0.7551	0.7768	0.7345	0.7590	FK	KCB
Motorbike	0.6415	0.6831	0.6085	0.6503	FK	KCB
Person	0.8154	0.8387	0.8107	0.8197	FK	KCB
Pottled Plant	0.2548	0.3096	0.2293	0.2667	FK	KCB
Sheep	0.4511	0.5044	0.4206	0.4619	FK	KCB
Sofa	0.4567	0.5031	0.4619	0.4621	FK	KCB
Train	0.7345	0.7987	0.7289	0.7460	FK	KCB
TV/Monitor	0.4867	54.23	0.5096	0.4814	FK	LLC

4 VIDEO CONCEPT CLASSIFICATION USING DEEP LEARNING

4.1 Methodology

4.1.1 Convolutional Neural Networks overview

Due to their recent success on many of the image processing tasks, we decide to import Convolutional Neural Networks (CNNs) into our processing pipelines. CNNs poses some unique properties for image understanding. First of all, they are highly transferable, i.e. one network trained on a given dataset may achieve state-of-the-art results on other datasets provided that overfitting is overcome by a suitable selection of training techniques. Secondly, they are highly parallel algorithms so vastly parallel architectures like graphical processing units (GPUs) can be utilized.

CNNs are biologically inspired feed-forward type artificial neural network architectures and they may be categorized as variants of multilayer perceptrons (MLP). Hubel and Wiesel [24] showed that biological visual cortex is composed of small cells sensitive to some part of visual field called receptive field. Receptive fields cover a portion of input image in a tiled arrangement of subregions which cover entire spatial correlations present in an image. There are two basic types revealed; simple cells responding maximally to edge like patterns in their receptive field and more complex cells, which have larger receptive fields, locally invariant to exact positions of input patterns. Early CNN architectures like LeNet [37] modelled this behaviour inherent in animal visual cortex. They found wide applications in image and video classification, natural language processing and many more computer vision tasks.

We investigate four different and popular CNN architectures in this part of our thesis. These CNN architectures are mostly winners or runners-up the famous image classification challenge ILSVRC [31] during their introduction time. CNN architectures are composed of a number of convolutional layers followed by subsampling layers. They may have additional fully-connected layers as well. CNN architectures all work with fixed input sizes but number of input color channels may vary from model to model. Given an image with resolution with size *W x H* and it is first resized to input resolution of the network, let's say *M x M*. Assuming we have an convolution layer at the input with kernel size “k”, stride “s” and padding “p”, this convolution layer produces “N” feature maps with size:

$$\text{Output size} = (M - k + 2 * p)/s + 1 \tag{4.1}$$

where “N” is fixed by network design. Each feature map then is subsampled using an average or max layers called pooling layers. After a stack of these multiple layers some fully-connected layers, which creates the vectoral outputs of a CNN, may follow. Figure 4.1 visualizes a generic CNN architecture layout.

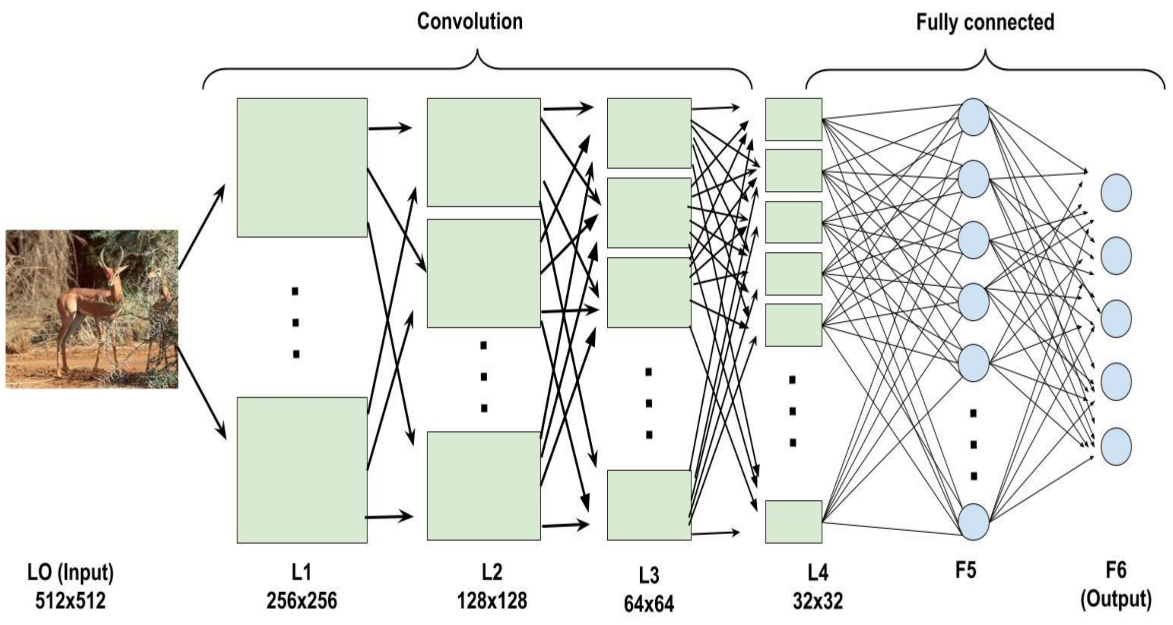
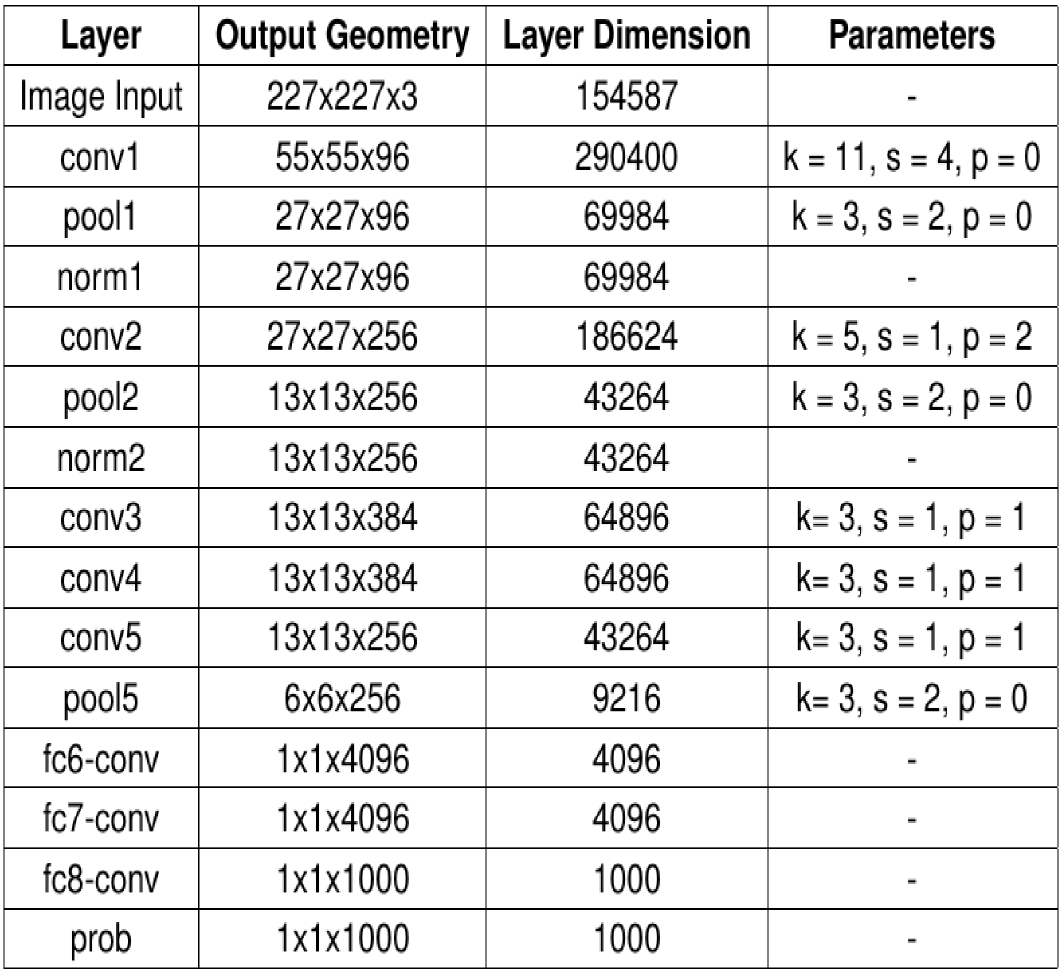


Figure 4.1 CNN architecture visualization

Our first CNN architecture is AlexNet, first introduced by Krizhevsky et.al in 2012 [31]. AlexNet consists of eight layers, five of them being convolutional layers and three layers are being fully-connected layers. They have a last softmax layer which is not included in this layer list. Softmax layer is tuned for semantic classification of 1000 ImageNet, or ILSVRC, classes. In addition to softmax layer, there are various other max-pooling layers present in the architecture which is not counted in the layer list as well. In addition to those, there are hidden rectified linear units (RELU) following each of the convolution and fully-connected layers. This architecture is important in many ways, for instance it is the first demonstration of success of CNNs at such large scale. Moreover, it is not only successful in image classification tasks, for instance state-of-the-art results are reported for real-time pedesterian detection using a slightly modified version of AlexNet, recently [1].

We show details of AlexNet architecture in Table 4.1. All output dimensions are calculated by using Equation 4.1

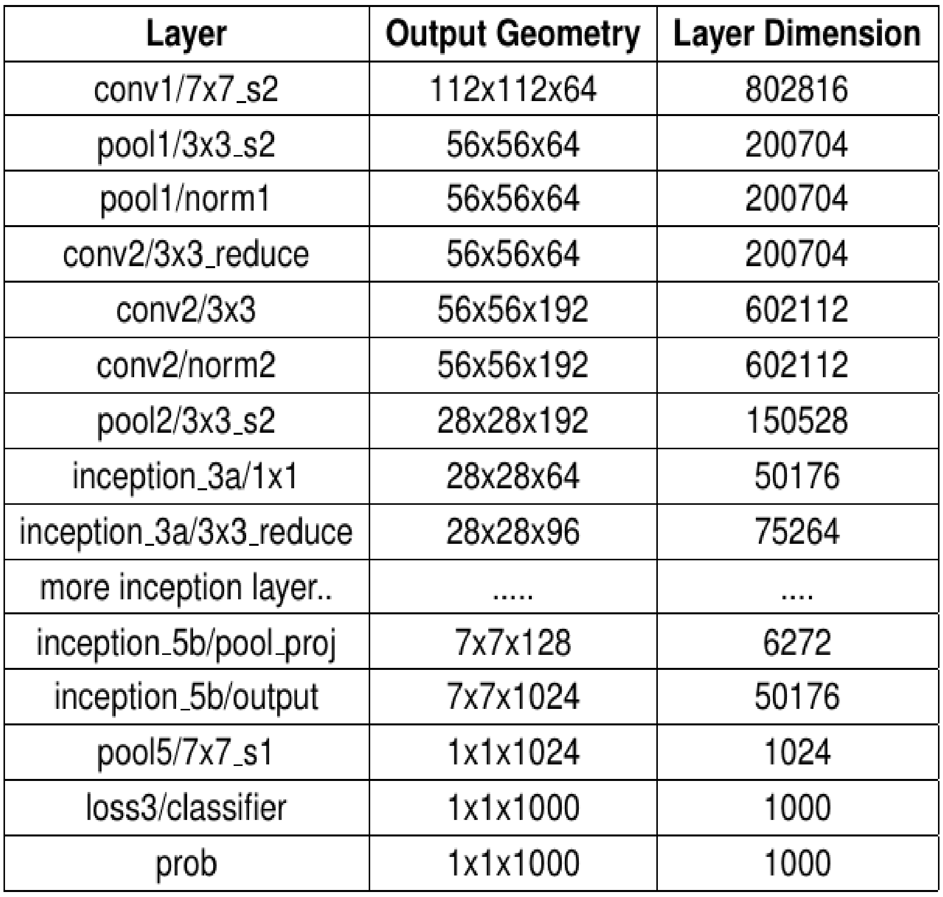
Table 4.1 AlexNet network architecture



Layer	Output Geometry	Layer Dimension	Parameters
Image Input	227x227x3	154587	-
conv1	55x55x96	290400	k = 11, s = 4, p = 0
pool1	27x27x96	69984	k = 3, s = 2, p = 0
norm1	27x27x96	69984	-
conv2	27x27x256	186624	k = 5, s = 1, p = 2
pool2	13x13x256	43264	k = 3, s = 2, p = 0
norm2	13x13x256	43264	-
conv3	13x13x384	64896	k= 3, s = 1, p = 1
conv4	13x13x384	64896	k= 3, s = 1, p = 1
conv5	13x13x256	43264	k= 3, s = 1, p = 1
pool5	6x6x256	9216	k= 3, s = 2, p = 0
fc6-conv	1x1x4096	4096	-
fc7-conv	1x1x4096	4096	-
fc8-conv	1x1x1000	1000	-
prob	1x1x1000	1000	-

Our second CNN model of choice is the winner of 2014 ILSVRC challenge by Szegedy et, al. [65]. This network derives from a so called Network-in-Network architecture of Lin et. al. [39] and called GoogleNet. GoogleNet is deeper network than AlexNet, consisting of twenty-two layers if pooling layers are not counted. If all the independent blocks are counted, their architecture is said to have approximately 100 layers. Although being four times deeper than AlexNet, GoogleNet is only 2x times less-efficient during inference time compared to AlexNet. They reach this efficient while utilizing larger networks by the help of filter-level sparsity concept first stated by Arora et. al. [3]. Table 4.2 summarizes the general layout of GoogleNet. Please note that whole GoogleNet architecture has almost 100 layers which we don’t show all in detail in this table.

Table 4.2 GoogleNet network architecture

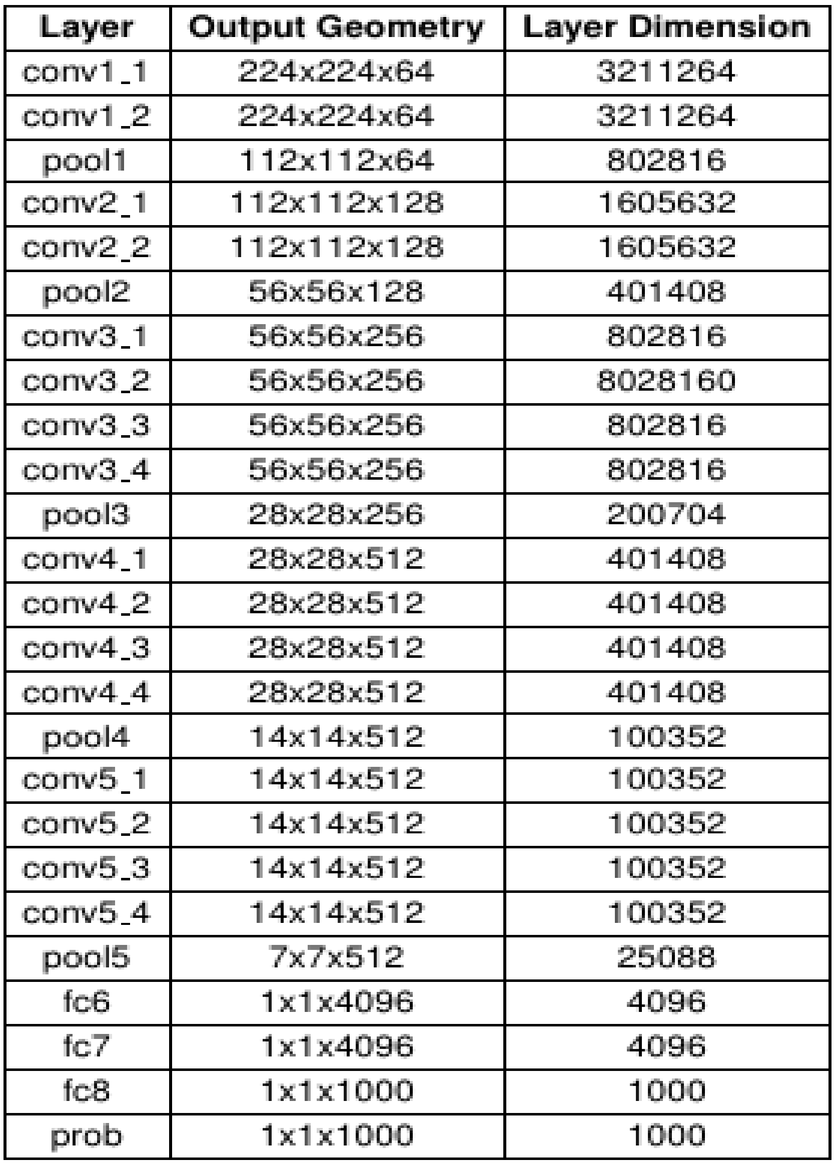


Layer	Output Geometry	Layer Dimension
conv1/7x7_s2	112x112x64	802816
pool1/3x3_s2	56x56x64	200704
pool1/norm1	56x56x64	200704
conv2/3x3_reduce	56x56x64	200704
conv2/3x3	56x56x192	602112
conv2/norm2	56x56x192	602112
pool2/3x3_s2	28x28x192	150528
inception_3a/1x1	28x28x64	50176
inception_3a/3x3_reduce	28x28x96	75264
more inception layer..	.....	....
inception_5b/pool_proj	7x7x128	6272
inception_5b/output	7x7x1024	50176
pool5/7x7_s1	1x1x1024	1024
loss3/classifier	1x1x1000	1000
prob	1x1x1000	1000

Our third and forth models are closely related to each other and introduced by Simonyan et. al. [62]. They secured second place of 2014 ILSVRC classification challenge, and won the same years localization challenge. They are code-named as VGG16 and VGG19 in the literature, latter being three layers deeper than the former. Their design is based on smaller convolutional filters in all layers of networks whereever possible. Smaller convolution steps permit them to increase network depth without too much increase in complexity. We show details of VGG networks in Table 4.3.

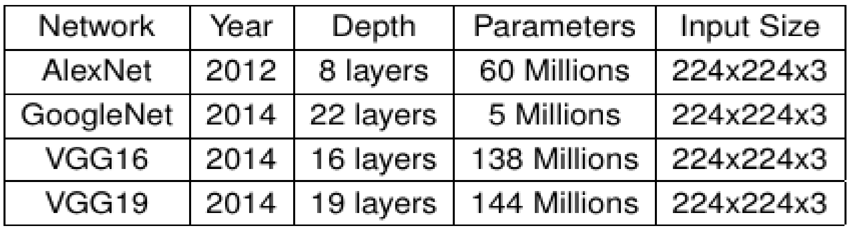
One of very important metrics for differentiating CNNs is their depth and number of parameters. Deeper networks with more parameters is harder to design and train, but often yield better results when compared shallower architectures. In Table 4.4 we provide various statistics about our CNN architectures.

Table 4.3 VGG19 network architecture



Layer	Output Geometry	Layer Dimension
conv1_1	224x224x64	3211264
conv1_2	224x224x64	3211264
pool1	112x112x64	802816
conv2_1	112x112x128	1605632
conv2_2	112x112x128	1605632
pool2	56x56x128	401408
conv3_1	56x56x256	802816
conv3_2	56x56x256	8028160
conv3_3	56x56x256	802816
conv3_4	56x56x256	802816
pool3	28x28x256	200704
conv4_1	28x28x512	401408
conv4_2	28x28x512	401408
conv4_3	28x28x512	401408
conv4_4	28x28x512	401408
pool4	14x14x512	100352
conv5_1	14x14x512	100352
conv5_2	14x14x512	100352
conv5_3	14x14x512	100352
conv5_4	14x14x512	100352
pool5	7x7x512	25088
fc6	1x1x4096	4096
fc7	1x1x4096	4096
fc8	1x1x1000	1000
prob	1x1x1000	1000

Table 4.4 Comparison of different network architectures



Network	Year	Depth	Parameters	Input Size
AlexNet	2012	8 layers	60 Millions	224x224x3
GoogleNet	2014	22 layers	5 Millions	224x224x3
VGG16	2014	16 layers	138 Millions	224x224x3
VGG19	2014	19 layers	144 Millions	224x224x3

4.1.2 Classification strategies

We have various classification strategies using CNNs:

1. We can train a CNN end-to-end on our target datasets.
2. We can fine-tune pre-trained CNN on our target datasets.

3. We can use lower layers of CNNs as feature extractors and develop further encoding schemes.
4. We can use higher layers of CNNs and combine them with simpler classifiers like SVMs.

First item is not suitable for our case, mainly due to two reasons. One is training end-to-end networks requires high-end computational resources which we don't have access through-out our thesis duration, secondly they require ultra-large scale datasets for achieving acceptable classification performance. For instance, GoogleNet was trained on a high-end distributed supercomputer of Google [11] and authors state that architecture can be trained on a few high-end GPUs in a week. VGG networks are not different than GoogleNet either in this regard, training one VGG network took three weeks using four top-notch GPUs in parallel [62].

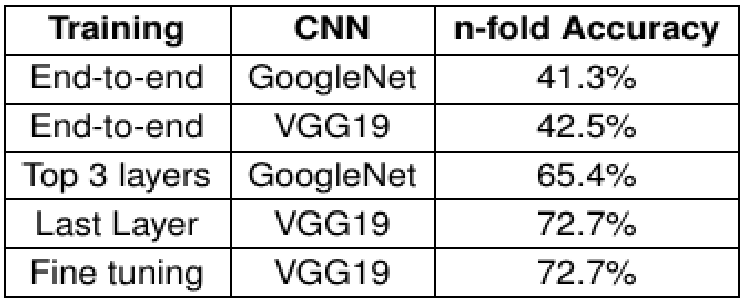
Training CNNs end-to-end may cause severe overfitting if they are not trained with enough data. We extract some facts from literature [61] [30] to show ineffectiveness of end-to-end training on relatively small datasets and list those in Table 4.5.

Our second option of using CNNs for image classification is to fine-tune a pre-trained CNN for our datasets in hand. This is in fact an efficient way of performing classification as seem in Table 4.5, however, we don't prefer this route as this step involves CNN training which is something we don't have too much experience about.

Our third option is a new research are in the literature and we investigate this option in our thesis. In this type of classification, we extract feature maps from lower layers network architectures and treat them as local image features. Later we run our BoW pipeline on those feature maps. We present our findings in the results section with our personal comments.

Our forth option is to combine CNNs with simple classifiers, SVMs in our case. Unlike third method, we use upper layers of networks for feature extraction without any encoding taking in between feature extraction and SVM classification. This type classification is examined by various studies in the literature as well.

Table 4.5 End-to-end CNN training on UCF-101 video dataset



Training	CNN	n-fold Accuracy
End-to-end	GoogleNet	41.3%
End-to-end	VGG19	42.5%
Top 3 layers	GoogleNet	65.4%
Last Layer	VGG19	72.7%
Fine tuning	VGG19	72.7%

4.2 Datasets In Use

We report our findings on Caltech datasets which we mention on sections 3.2.1 and 3.2.2. In addition to Caltech datasets, we make use of UCF101 dataset [64]. UCF-101 is an action recognition dataset of realistic action videos, collected from YouTube. It consists of 101 action classes, over 13k clips and 27 hours of video data. UCF-101 currently the largest dataset of human actions. As most of the available action recognition datasets are not realistic and are staged by actors, UCF-101 aims to encourage further research into action recognition by learning and exploring new realistic action categories.

4.3 Results

We first investigate effects using lower layers of CNN architectures as local features and apply BoW methodology for image classification purposes. We choose AlexNet as our CNN model. For each of the CNN layers, we first extract local image features using feature maps of receptive fields in the CNN layers. Then we create a visual vocabulary. Later using this vocabulary, we encode local image features. As a final step, we apply level two spatial pyramids and map features using homogenous kernel mapping of type x^2. We use linear classifiers for classification, with cost parameter of SVMs set to 10.

Figure 4.2 summarizes our findings. By looking at the numbers we see that convolution layers should be choice for feature extraction as outputs of pooling and normalization layers perform worse significantly. Second of all, as we go deeper in the network layers, accuracy consistently increases. This can be attributed to learning ability of CNNs. In the very first layers, learned features are weak and less invariant to changes in the image structure. As we go deeper, with the effects of more convolutions, learned features become stronger and with the effects of max-pooling layers they gain invariance to various imaging operations.

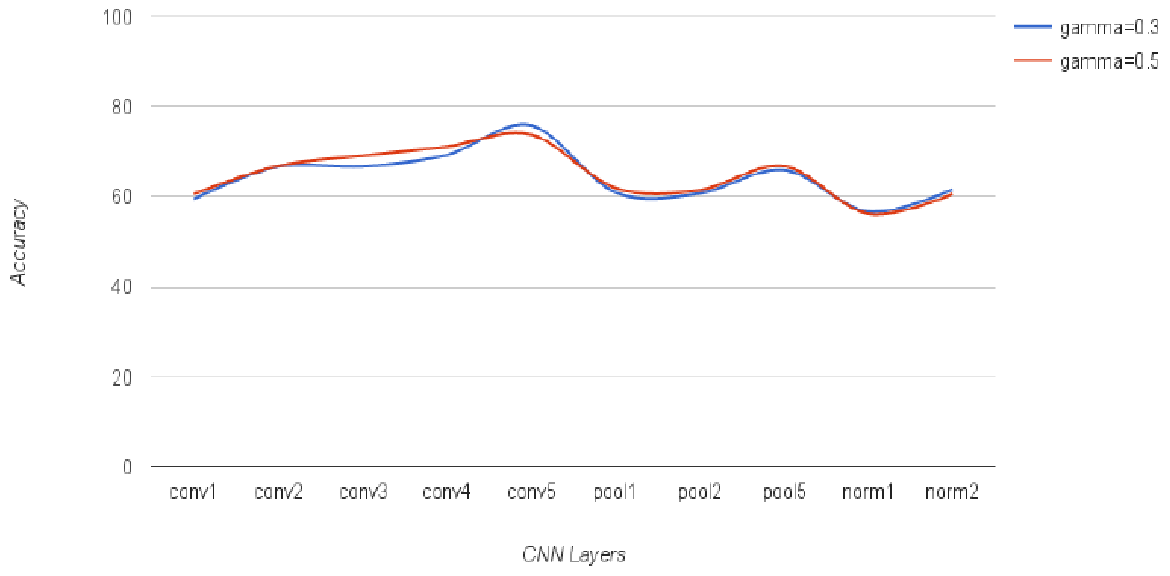


Figure 4.2 BoW methodology applied to CNN features on Caltech-101

Our most interesting finding is, though, features learned by network is almost better than hand-crafted features like SIFT for classification tasks. This leads to one another observation but we hold-on to that until we investigate other classification method of ours. In our second group of experiments, instead of extracting features from early layers and encoding them with BoW, we extract features from higher levels of our CNN architecture. We specifically choose final fully-connected layers for such a comparison. At a first glance, this is somehow expected to perform worse because we know that CNNs learn generic image features in the lower layers and more high-level semantics of images in the upper layers. Since our models are trained on an un-correlated and totally different dataset then we are using, we expect these semantics to be dataset specific. However, by looking our results shown in Table 4.6, we see this is not the case and CNNs achieve state-of-the-art results on our benchmark Caltech datasets. These unexpected results can be attributed to two distinct concepts. First, learned semantic features are more like mid-level image features rather than high-level image semantics. Secondly, CNNs are not only good at learning low-level image features, they are good at learning better encodings for those features as well. We must emphasize that CNN features perform equally well with linear SVMs.

Table 4.6 CNN classification performance on Caltech datasets

Network	Caltech101	Caltech256	Layer
AlexNet	88.77%	70.67%	fc6
VGG16	91.48%	-	fc6
VGG19	91.81%	80.49%	fc6
GoogleNet	91.48%	79.70%	inception_5b/output

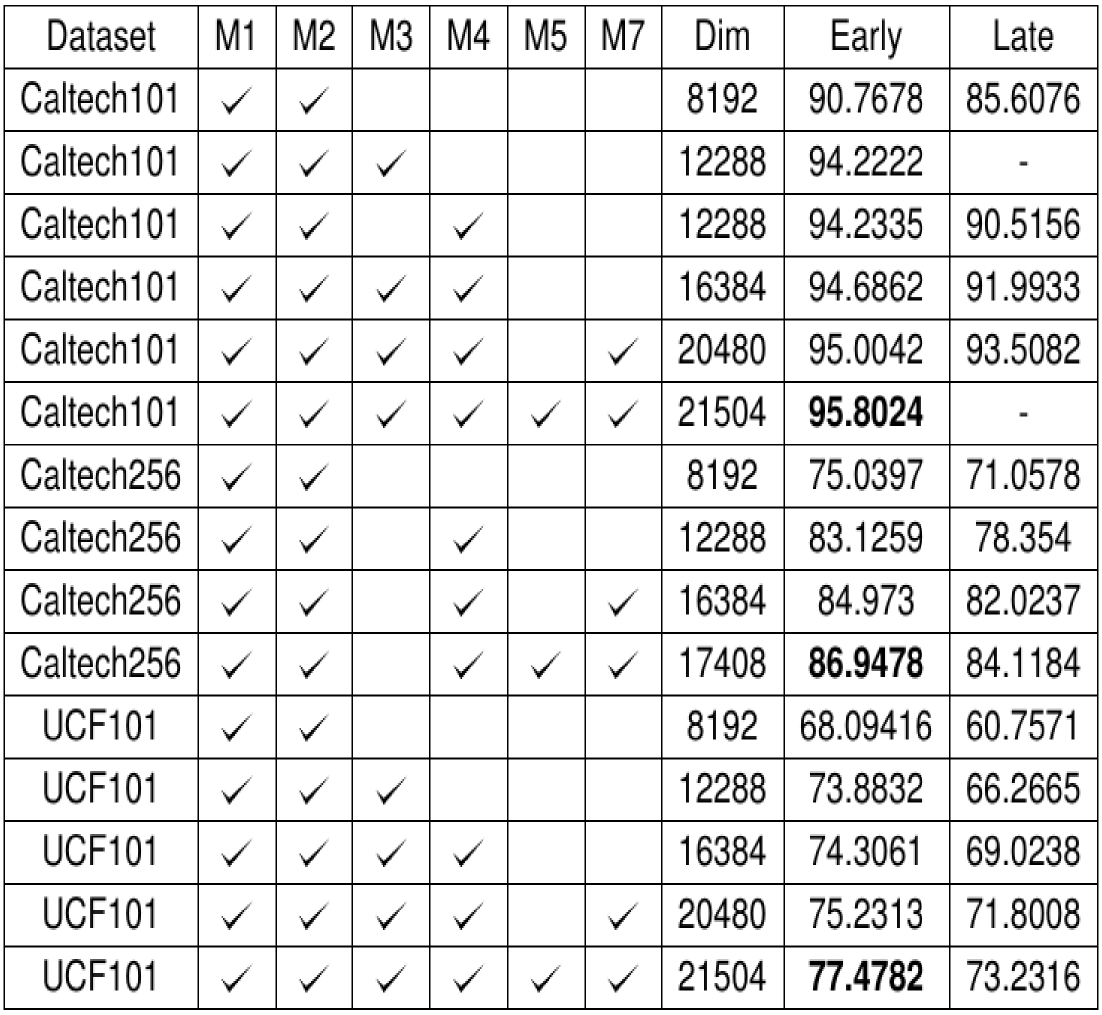
Finally we investigate effects model fusion of different CNN models. Given that CNNs are learning image features and related encodings, it is highly possible that different architectures may learn different representations which may be complementary to each other. We use seven different CNN models:

1. Model 1: AlexNet architecture trained on ImageNet dataset.
2. Model 2: Re-shaped AlexNet architecture for higher input resolutions.
3. Model 3: VGG16 architecture trained on ImageNet dataset.
4. Model 4: VGG19 architecture trained on ImageNet dataset.
5. Model 5: GoogleNet architecture trained on ImageNet dataset.
6. Model 6: AlexNet architecture trained on MIT places dataset [86].
7. Model 7: Re-shaped VGG16 architecture for higher input resolutions.

Our purpose for the selection of such a mix is to use insert diversity into our CNN selection. We operate on two CNNs, namely AlexNet and VGG16, and change their internal structure so that they can work with higher resolution images. We apply net surgery in a way that all the pre-trained parameters of the networks stay the same and we do not perform any model training. We also use a model pre-trained on a different dataset which we think may add some unique information to our final model. Fusing operation is performed on fully-connected layers of CNN models, i.e. we do not apply any re-coding. We fuse extracted features in two fashions, using early and late fusion strategies. Early fusion is the merging of features before presenting them to classifier. Late fusion is the merging of results of different SVMs using a proper voting scheme. Table 4.7 summarizes our findings, which we also shared in a submitted paper of ours for BigMM 2016 conference [15]. By looking at the results, we can say that early fusion consistently outperforms late fusion. For Caltech datasets, our results are better than current state-of-the-art results with a strong margin of 4-5%. For UCF101 dataset, our results still perform better than results demonstrated in literature using only static image features. Since we haven't employed any motion features, we are not compatiple with studies using motion features like STIP [32].

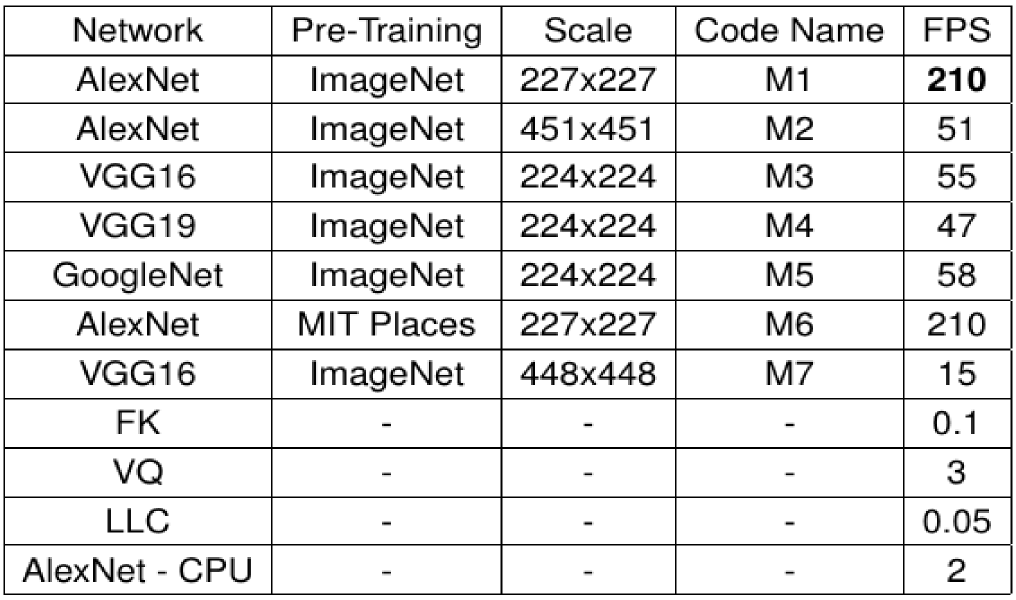
We show cardinality of each fusion in the "Dim" column of Table 4.7. Even fusion of seven different architectures generates smaller feature sizes than various BoW encodings. They are even much smaller than Fisher Vector encoding which is the most successful feature encoding presented in BoW results. Please note that FK dimensionality can be as high as 800k for a reasonable accuracy. Finally we compare run-time performance of various encodings in Table 4.8 which clearly states superiority CNN architectures. Please note that our CNN architectures are running on highly optimized GPU architectures, we show their CPU-only performance as well which is still far better than FK encoding.

Table 4.7 Fusion performance of different network architectures



Dataset	M1	M2	M3	M4	M5	M7	Dim	Early	Late
Caltech101	✓	✓					8192	90.7678	85.6076
Caltech101	✓	✓	✓				12288	94.2222	-
Caltech101	✓	✓		✓			12288	94.2335	90.5156
Caltech101	✓	✓	✓	✓			16384	94.6862	91.9933
Caltech101	✓	✓	✓	✓		✓	20480	95.0042	93.5082
Caltech101	✓	✓	✓	✓	✓	✓	21504	**95.8024**	-
Caltech256	✓	✓					8192	75.0397	71.0578
Caltech256	✓	✓		✓			12288	83.1259	78.354
Caltech256	✓	✓		✓		✓	16384	84.973	82.0237
Caltech256	✓	✓		✓	✓	✓	17408	**86.9478**	84.1184
UCF101	✓	✓					8192	68.09416	60.7571
UCF101	✓	✓	✓				12288	73.8832	66.2665
UCF101	✓	✓	✓	✓			16384	74.3061	69.0238
UCF101	✓	✓	✓	✓		✓	20480	75.2313	71.8008
UCF101	✓	✓	✓	✓	✓	✓	21504	**77.4782**	73.2316

Table 4.8 Running-Time performance of various architectures in use



Network	Pre-Training	Scale	Code Name	FPS
AlexNet	ImageNet	227x227	M1	**210**
AlexNet	ImageNet	451x451	M2	51
VGG16	ImageNet	224x224	M3	55
VGG19	ImageNet	224x224	M4	47
GoogleNet	ImageNet	224x224	M5	58
AlexNet	MIT Places	227x227	M6	210
VGG16	ImageNet	448x448	M7	15
FK	-	-	-	0.1
VQ	-	-	-	3
LLC	-	-	-	0.05
AlexNet - CPU	-	-	-	2

5 CONTENT AND CONCEPT BASED RETRIEVAL

Due to recent burst of multimedia data availability in 21th century, searching and retrieving aspects multimedia content is gaining more and more attention everyday. Tera-bytes of audio and video content is generated day-by-day and waiting to be indexed for further processing.

There can be many different query types for multimedia retrieval. On the other hand, image and video retrieval can classified down to two alternatives; text-based and content-based queries [40]. Content-based approaches or content-based image retrieval (CBIR) permits users to search their multimedia databases with the help of an example query item. In contrast, text-based approaches associate video shots or images with multiple semantic labels which once again can be queries by users.

As in image classification, BoW methodology is once again one of the most frequently assessed alternatives for retrieval applications. BoW offers most successful results in terms of retrieval accuracy and query running time. In this part of our study, we are investigating running efficient retrieval of image queries towards large scale video retrieval. We propose a processing pipeline similar to classification case with some modifications for running example based queries. Block diagram of our proposed pipeline architecture is depicted in Figure 5.1.

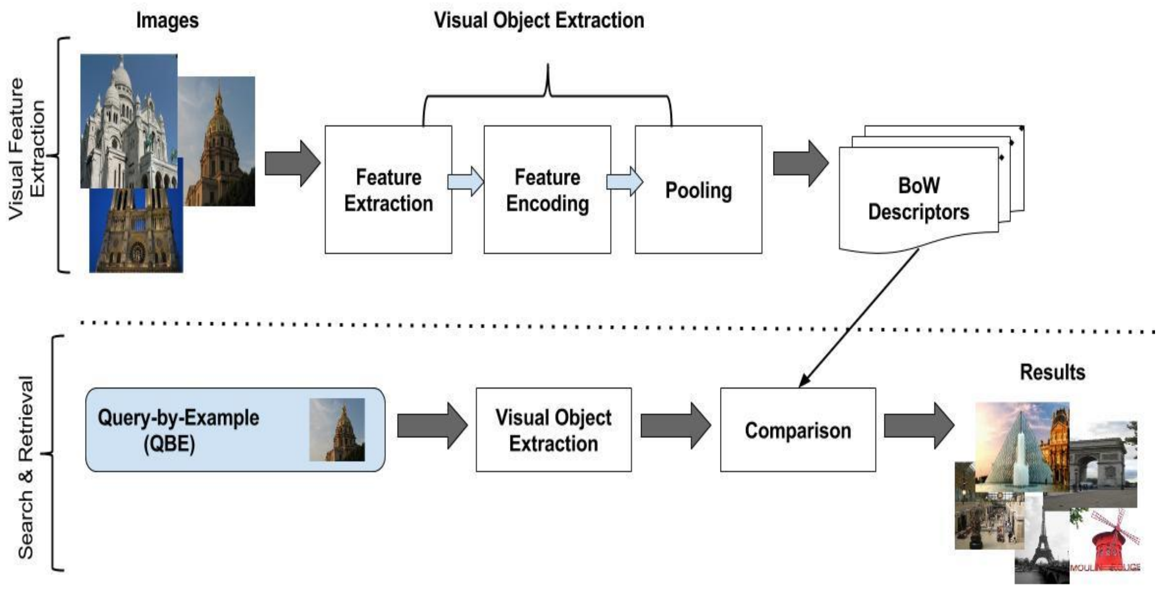


Figure 5.1 Overall view of the proposed search and retrieval system

5.1 Content-Based Retrieval

We use a very similar pipeline in retrieval as well. We first extract SIFT features of a query image. However, we use sparse sampling in place of dense sampling. Then we encode extracted features using VQ with a much higher dictionary codeword size compared to classification case. We apply spatial pyramids depending on the query type; content-based queries do not use SPM. In the final step, unlike image classification, simpler similarity measures are used in place of SVM classification for comparison of different image descriptors. We look into details of different distance metrics in the section 5.4.

5.2 Query Types

We implement two types of queries for demonstration purposes; query-by-example (QBE) and keyword queries.

QBE is the search of a given sample image in our local image database for matching pairs. This can be translated to a text query of “Find all images which are similar to this one”. QBE removes the difficulties rising from the description of image content using words.

Keyword queries, or semantic queries, permits user to query local image database with the use of high-level semantic keywords. User supplies a query keyword and retrieval systems scans tags of local images for matching descriptions. Here we assume that relevant tags are attached to images during indexing time.

5.3 Inverted Index

Inverted index is a very effective technique for reducing both memory requirements and time complexity of retrieval operations. An inverted index keeps a look-up table which contains associated codewords or dictionary entries for a corresponding image present in our database.

In order to reduce time complexity of underlying retrieval operations. We build inverted index in a way to benefit sparse structure of image descriptors. It keeps a look-up table of all images in database which contains specific codeword. i.e. dictionary entry. Figure 5.2 shows a graphical interpretation of our inverted index structure.

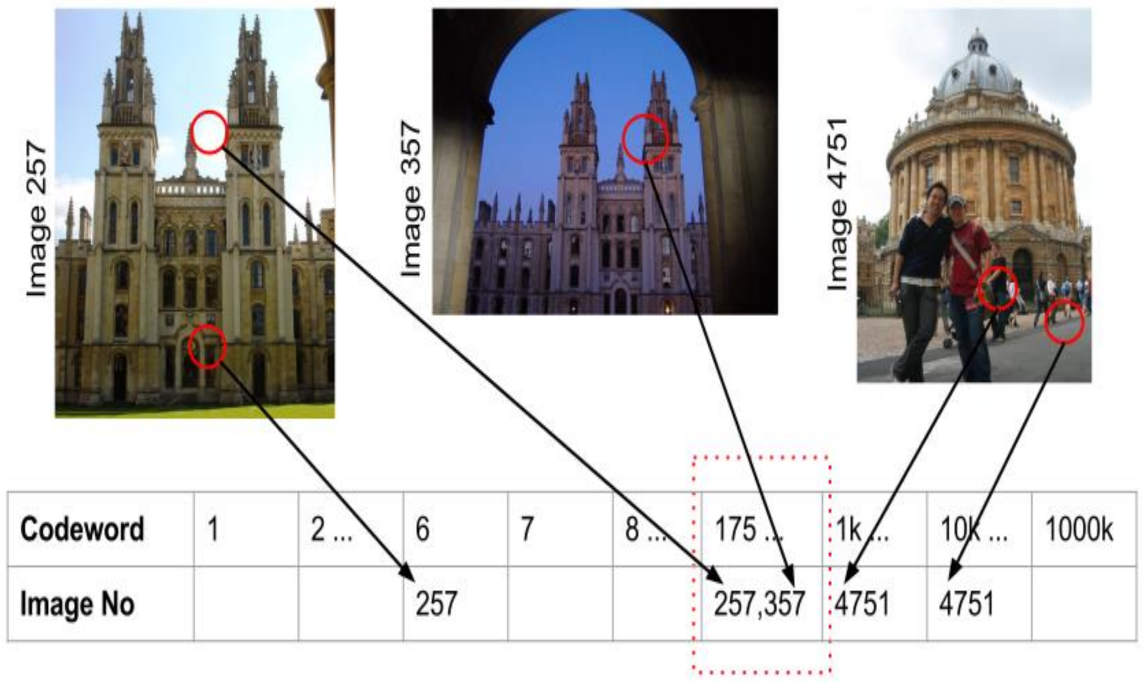


Figure 5.2 Overall view of the proposed search and retrieval system

This representation is effective in case of sparse image descriptors. In other words, inverted index exploits sparsity present in our image representations. Let's assume average number of features per image is M and suppose we have dictionary of size K. Recalling BoW representation give in previously, we can represent an image descriptor with a K-sized vector:

$$D(w) = (w_1, w_2, w_3, w_4, w_5 ... w_K) \tag{5.1}$$

For practical applications, M << K resulting in a very sparse descriptor D, i.e. many elements of D vector is zero. Since inverted index only keeps non-zero codewords coefficients of D in our database both memory requirements and similarity comparison time complexity is greatly reduced. This reduction is more dramatic in case where more than one image feature is quantized into the same dictionary codeword entry, which is a valid case for many practical applications. Keeping BoW descriptors as is results in linear increase both in dictionary size and number of images which is clearly avoided by the use of an inverted index.

To put in numbers, for Oxford Buildings database we have 9000 images, hence N = 9000. With a dictionary of size K = 1 million and empirical value of M = 3000, without use of inverted index we would have K x N = 9 billions of multiply-accumulate operations in case of L1 distance metric. Number of operations are much higher with the use of more advanced distance metrics. With the use of an inverted index, number of multiply-accumulate cycles reduces to M x N = 30 millions. This is a reduction factor of 0.3%.

Use of an inverted index with different distance metrics in not a trivial task due to layout of data structure in database. Since all elements of image descriptor vectors are not kept in database and are not accessible during query time, direct computation of similarity is not possible. However, it is possible to decompose most of the distance metrics, if not all, into two distinct summation terms. One of the terms only depends on the query vector, and the other term depends only on the database vectors. Then we can calculate both terms separately since only non-zero database elements contribute to final similarity measure. This permits us to reach almost constant time distance comparison, even for heavy similarity measures.

Our image classification pipeline steps employed in example based queries is in depicted in Figure 5.3. As a first step, we extract local SIFT features for every image using DoG detector. After local feature extraction, local features are hard vector quantized to create image feature histograms. Following vector quantization, we L1 normalize created histogram so that effect of unequal feature cardinalities in different images are neutralized. Spatial pyramid extension is not used for running queries due to use of sparse sampling. We rely on feature detectors so relevant spatial information is not discarded. Following this step, to suppress frequently used visual codewords we insert inverse document frequencies (idf) into extracted feature histograms. In the final step we compare our query image descriptor to the ones present in our database using different distance metrics.

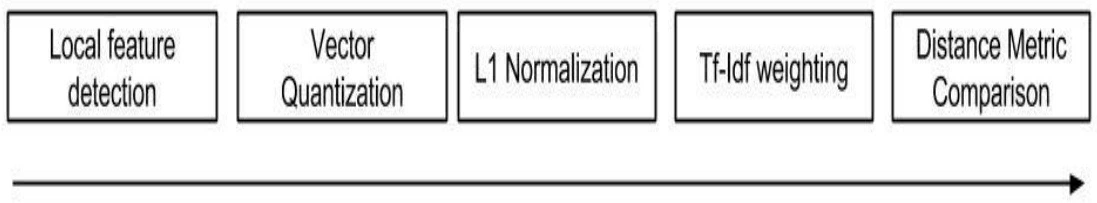


Figure 5.3 Retrieval pipeline overview

5.4 Distance Metrics

For comparing different image BoW descriptors, a suitable distance comparison should be employed. There are various alternatives, we evaluate performances of following metrics in our study.

5.4.1 L1 distance

L1 distance or norm, is defined in Equation 5.2.

$$L_1(x, y) = \sum_{i}^{m} |x_i - y_i| \tag{5.2}$$

5.4.2 L2 distance

L2 distance or norm, is defined in Equation 5.3.

$$L_2(x, y) = \sqrt{\sum_{i}^{m} (x_i - y_i)^2} \tag{5.3}$$

5.4.3 Histogram Intersection

Histogram intersection is a special metric used to compare histogram-like features. Histogram intersection can be formulated as in Equation 5.4.

$$H_{min}(x, y) = \sum_{i}^{m} min(x_i, y_i) \tag{5.4}$$

5.4.4 Hellinger distance

Hellinger distance, or Bhattacharyya’s distance, is used to compare similarity between two probability distributions. It is defined as in Equation 5.5 or 5.6 if vectors are L2 normalized.

$$H(x, y) = \frac{1}{\sqrt{2}} \left[\sum_{i}^{m} (\sqrt{x_i} - \sqrt{y_i})^2 \right]^{\frac{1}{2}} \tag{5.5}$$

$$H(x, y) = \left[1 - 1 \sum_{i}^{m} \sqrt{x_i y_i} \right]^{\frac{1}{2}} \tag{5.6}$$

5.4.5 Chi-Squared

Chi-Squared can be represented with Equation 5.7.

$$\chi^2(x,y) = \sum_{i}^{m} \frac{(x_i - y_i)^2}{x_i} \tag{5.7}$$

5.4.6 Cosine distance

Cosine distance can be seen in Equation 5.8.

$$D(x,y) = \frac{\sum x_i y_i}{\sqrt{\sum (x_i)^2}\sqrt{\sum (y_i)^2}} \tag{5.8}$$

5.5 Datasets

The Oxford Buildings dataset [53] consists of 5062 images extracted from Flickr by searching known Oxford landmarks. The collection had been manually annotated to generate a comprehensive ground truth for 11 different landmarks, each represented by 5 possible queries. This gives a set of 55 queries over which an object retrieval system can be evaluated. Figure 5.4 shows some examples from the dataset and Table 5.1 lists structure of the dataset.

Table 5.1 Summarization of concepts for Oxford dataset

Concept	Perfect Positives	Partial Positives	Total Images
All Souls	120	270	390
Ashmolean	60	65	125
Balliol	25	35	60
Bodleian	65	55	120
Christ Church	255	135	390
Cornmarket	25	20	45
Hertford	175	95	270
Keble	30	5	35
Magdalen	65	205	270
Pitt Rivers	15	15	30
Radcliffe	525	580	1105

The Paris Dataset [54] is composed of 6412 images collected from Flickr by searching for particular Paris landmarks. It includes 55 different queries on 12 different Paris buildings, much like Oxford buildings dataset. Figure 5.5 shows some examples from the dataset and Table 5.2 lists structure of the dataset.

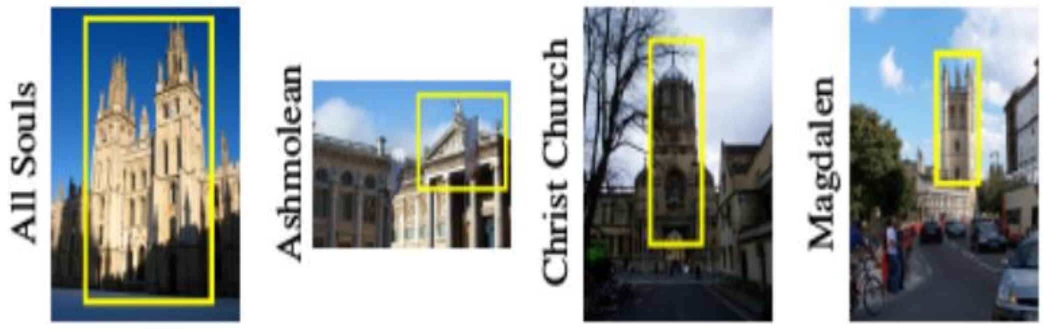


Figure 5.4 Some samples from Oxford dataset

Figure 5.5 Some samples from Paris dataset

In addition to Oxford and Paris datasets, we use Pascal VOC 2007 dataset for reporting our keyword query results. We mention details of Pascal VOC dataset in Section 3.2.3.

Table 5.2 Summarization of concepts for Paris dataset

Concept	Positives	Concept	Positives
Defense	585	Eiffel	1445
Invalides	990	Louvre	760
Moulin rouge	1158	Triomphe	1405
Musee dorsay	360	Notre Dame	595
Pantheon	630	Pompidou	255
SacreCoeur	745		

5.6 Results

We present our results from a recent study of our [16] in Figure 5.6 and Table 5.3. We investigate CBIR performance on Oxford Buildings and Paris datasets, then we conduct experiments with aformentioned comparison metrics on both datasets. We also investigate effects of L1 and L2 normalizations. Table 5.3 summarizes our results.

We also evaluate different metrics on Oxford Buildings dataset without IDF weighting scheme applied. Applying IDF weighting adds approximately 2 percent to average precision.

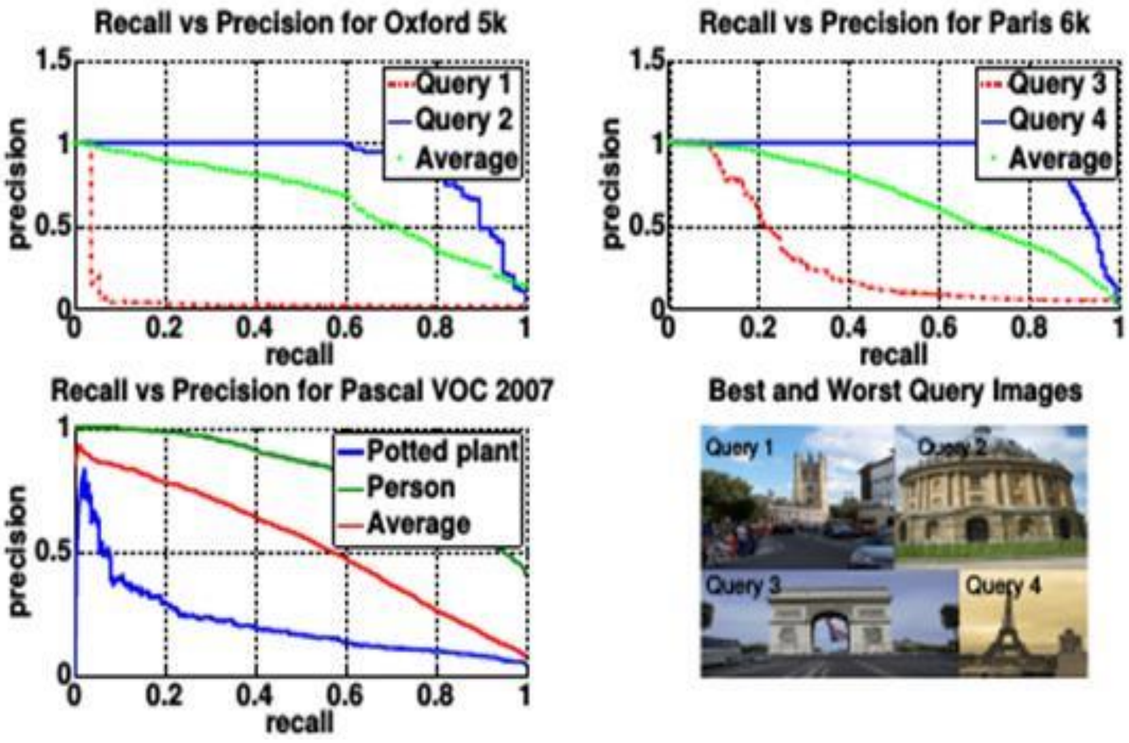


Figure 5.6 Precision and recall (PR) curves. The best, average, and worst queries/concepts are considered for each dataset using the proposed scheme: (a) Oxford Buildings dataset, (b) Paris dataset, (c) Pascal VOC 2007 dataset

Among all distances, Hellinger distance performs the best and this is in-line with results presented by [2]. Although Hellinger distance is heavier to compute compared to most of the other distances it was shown that it can be computed very efficiently with an additional normalization step to SIFT descriptor calculation

[2]. This normalization step doesn’t even need a modification in SIFT extraction routines, it is possible to apply normalization during quantization of image features.

Histogram intersection based distance comparison (min) is consistently second after Hellinger distance. This is en expected outcome for 2 reasons, one is that it has been shown to be superior for object classification tasks. [33] and, second, BoW descriptors are merge histograms. It should be noted that histogram intersection performs better with L1 on some datasets while it is performing better with L2 normalization on some other datasets with very slight changes so we can comment that it is normalization agnostic.

Table 5.3 Comparison of different distance metrics on Oxford Buildings and Paris datasets

Metric	Oxford5k	Paris6k	Oxford5k(No Idf)	Oxford5k	Paris6k	Oxford5k(No Idf)
	L1 Normalized			L2 Normalized		
L1	0.6176762	0.62068862	0.60776925	0.038044896	0.033984952	0.038047113
L2	0.075195491	0.043052912	0.075260796	0.61359107	0.60580742	0.59819621
Min	0.61762261	0.6501981	0.60809761	**0.62426698**	**0.64156407**	**0.61962712**
Cos	0.61361635	0.6333003	0.59808475	0.61361593	0.63330024	0.59807926
Hellinger	**0.6378966**	**0.65200478**	**0.6314373**	0.60204697	0.60970277	0.59328997
χ^2	0.59236705	0.62142205	0.58517522	0.55918133	0.5897671	0.55282593

6 CONCLUSIONS

6.1 Main Findings

In this thesis, we investigate many parameters and various extensions of BoW paradigm. We show the effectiveness of VQ versus different encodings. We show that it is fast, robust and scalable. Although it performs worse in accuracy when compared to FK, we believe extra overhead of FK should be avoided due to high memory requirements and relatively slower encoding timings. Repeating findings in Table 4.8, FK is more than twenty times slower than VQ in coding step. It also generates ten times larger feature vectors compared to VQ which increases inference time and memory usage. We believe deep learning algorithms should be used in place of FK, and VQ can be used jointly to further push classification accuracy.

In addition to feature encoding, we show that visual dictionary can be created using approximate k-means algorithms which make it possible to create more discriminative vocabulary with lower complexity of $\mathcal{O}(NlogN)$ vs $\mathcal{O}(NK)$.

Another important point is use of linear classifiers. Even our VQ encoding is performing better with nonlinear kernels, we show that kernel mapping can be applied prior to classification and linear SVMs can be used in place of nonlinear SVMs.

Finally, we evaluated our best selection of implementation parameters on two benchmark datasets of Caltech-101 and Caltech-256 showing state-of-the-art results among BoW implementations.

In addition to BoW methodology, we analyze deep learning in the context of visual concept classification. We investigate various CNN architectures show effectiveness of deeper architectures. We combine CNN architectures with BoW encoding, surprisingly, we achieve worse results when compared to CNN models alone but better or identical results using BoW encodings with hand-crafted features. This shows that CNN architectures are both able to learn effective image features and optimal encodings for learned image features, although proving this optimality is somehow impossible. But we can say that they are able to saturate results on benchmark datasets which gives us some clues about their optimality.

We also show that proposed BoW architectures perform well in retrieval applications. We show how Hellinger’s distance can be employed in L2 distance for better retrieval precision. We also show advanced distance metrics can be used with inverted index database structures.

6.2 Limitations and Future Work

Due to shortcomings in our computer hardware resources we haven’t investigated effects of CNN training in our thesis. Furthermore, we somehow lack proper understanding of CNN architectures’ advanced performances which prevents us from further pushing their classification accuracies with the integration of more advanced encodings. Another limitation of our thesis is that we discard valuable audio information present in the video which can definetely help us to improve classification accuracies. Being parallel to that, video specific motion features are discarded in our thesis. However, integrating audio and motion information into our classification pipeline would certainly boost our classification performance. We wish to address these shortcomings in our future studies.

REFERENCES

[1] AneliaAngelova, Alex Krizhevsky, Vincent Vanhoucke, AbhijitOgale, and DaveFerguson. Real-time pedestrian detection with deep networkcascades.

[2] SanjeevArora, AdityaBhaskara, RongGe, and Tengyu Ma. Provable boundsfor learning some deep representations. arXiv preprint arXiv:1310.6343, 2013.

[3] Anna Bosch, Andrew Zisserman, and Xavier Munoz. Image classification usingrandom forests and ferns. In Computer Vision, 2007.ICCV 2007. IEEE 11thInternational Conference on, pages 1–8. IEEE, 2007.

[4] Bernhard E Boser, Isabelle M Guyon, and Vladimir N Vapnik. A training algorithm for optimal margin classifiers. In Proceedings of the fifth annual workshopon Computational learning theory, pages 144–152. ACM, 1992.

[5] G. Bradski. Dr. Dobb's Journal of Software Tools, 2000.

[6] Chih-Chung Chang and Chih-Jen Lin. LIBSVM: A library for support vector machines. ACM Transactions on Intelligent Systems and Technology, 2:27:1–27:27, 2011.Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.

[7] Ken Chatfield, Victor S Lempitsky, Andrea Vedaldi, and Andrew Zisserman. The devil is in the details: an evaluation of recent feature encoding methods. InBMVC, volume 2, page 8, 2011.

[8] Corinna Cortes and Vladimir Vapnik.Support-vector networks. Machine learning, 20(3):273–297, 1995.

[9] Gabriella Csurka, Christopher Dance, Lixin Fan,JuttaWillamowski, and CédricBray.Visual categorization with bags of keypoints. In Workshop on statistical learning in computer vision, ECCV, volume 1, pages 1–2. Prague, 2004.

[10] Jeffrey Dean, Greg Corrado, RajatMonga, Kai Chen, Matthieu Devin, Mark Mao, Andrew Senior, Paul Tucker, Ke Yang, Quoc V Le, et al. Large scale distributed deep networks. In Advances in Neural Information Processing Systems, pages 1223–1231, 2012.

[11] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In Computer Vision and PatternRecognition, 2009.CVPR 2009.IEEE Conference on, pages 248–255.IEEE, 2009.

[12] Jeff Donahue, YangqingJia, OriolVinyals, Judy Hoffman, Ning Zhang, Eric Tzeng, and Trevor Darrell. Decaf: A deep convolutional activation feature forgeneric visual recognition. arXiv preprint arXiv:1310.1531, 2013.

[13] Charles Elkan. Using the triangle inequality to accelerate k-means. In ICML, volume 3, pages 147–153, 2003.

[14] Hilal Ergun and Mustafa Sert. Fusing deep convolutional networks for large scale visual concept classification.

[15] Hilal Ergun and Mustafa Sert. Efficient bag of words based concept extraction for visual object retrieval. In Flexible Query Answering Systems 2015, pages 389–402. Springer, 2016.

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

[17] Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. LIBLINEAR: A library for large linear classification. Journal of MachineLearning Research, 9:1871–1874, 2008.

[18] Li Fei-Fei, Rob Fergus, and PietroPerona. Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. Computer Vision and Image Understanding, 106(1):59–70, 2007.

[19] Kristen Grauman and Bastian Leibe. Visual object recognition.Number 11. Morgan & Claypool Publishers, 2010.

[20] Gregory Griffin, Alex Holub, and PietroPerona. Caltech-256 object category dataset. 2007.

[21] JianGuo and Stephen Gould. Deep CNN ensemble with data augmentation for object detection. CoRR, abs/1506.07224, 2015.

[22] Kaiming He, Xiangyu Zhang, ShaoqingRen, and Jian Sun. Deep residual learning for image recognition. arXiv preprint arXiv:1512.03385, 2015.

[23] David H Hubel and Torsten N Wiesel. Receptive fields, binocular interaction and functional architecture in the cat’s visual cortex. The Journal of physiology,160(1):106, 1962.

[24] HerveJegou, MatthijsDouze, and CordeliaSchmid.Hamming embedding and weak geometric consistency for large scale image search. In Computer Vision–ECCV 2008, pages 304–317. Springer, 2008.

[25] YangqingJia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell.Caffe: Convolutional architecture for fast feature embedding. arXiv preprint arXiv:1408.5093, 2014.

[26] Thorsten Joachims. Training linear svms in linear time. In Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining, pages 217–226. ACM, 2006.

[27] Frederic Jurie and Bill Triggs.Creating efficient codebooks for visual recognition.In Computer Vision, 2005.ICCV 2005. Tenth IEEE International Conference on, volume 1, pages 604–610. IEEE, 2005.

[28] Kaiming He, Xiangyu Zhang, ShaoqingRen, and Jian Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition.Pattern Analysis and Machine Intelligence, IEEE Transactions on,37(9):1904–1916, 2015

[29] Andrej Karpathy, George Toderici, SachinShetty, Tommy Leung, Rahul Sukthankar, and Li Fei-Fei. Large-scale video classification with convolutional neural networks.In Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on, pages 1725–1732.IEEE, 2014.

[30] Alex Krizhevsky, IlyaSutskever, and Geoffrey E. Hinton.Imagenet classification with deep convolutional neural networks. In F. Pereira, C.J.C.

Burges, L. Bottou, and K.Q. Weinberger, editors, Advances in Neural Information Processing Systems 25, pages 1097–1105. Curran Associates, Inc., 2012.

[31] Ivan Laptev, MarcinMarszałek, CordeliaSchmid, and Benjamin Rozenfeld. Learning realistic human actions from movies.In Computer Vision and Pattern Recognition, 2008.CVPR 2008.IEEE Conference on, pages 1–8.IEEE, 2008.

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

[33] Svetlana Lazebnik and Maxim Raginsky. Supervised learning of quantizer codebooks by information loss minimization. Pattern Analysis and Machine Intelligence, IEEE Transactions on, 31(7):1294–1309, 2009.

[34] Svetlana Lazebnik, CordeliaSchmid, and Jean Ponce.Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. InComputer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on, volume 2, pages 2169–2178. IEEE, 2006.

[35] Y LeCun, B Boser, J Denker, D Henderson, R Howard, W Hubbard, and L Jackel.Backpropagation applied to handwritten zip code recognition. NeuralComputation, 1(4):541–551, Dec 1989.

[36] YannLeCun, LéonBottou, YoshuaBengio, and Patrick Haffner. Gradient-based learning applied to document recognition. Proceedings of the IEEE, 86(11):2278–2324, 1998.

[37] Thomas Leung and Jitendra Malik.Recognizing surfaces using three-dimensional textons.In Computer Vision, 1999.The Proceedings of the Seventh IEEE International Conference on, volume 2, pages 1010–1017.IEEE, 1999.

[38] Min Lin,Qiang Chen, and Shuicheng Yan. Network in network.CoRR, abs/1312.4400, 2013.

[39] Jialu Liu. Image retrieval based on bag-of-words model. CoRR, abs/1304.5168, 2013.

[40] Jingen Liu and Mubarak Shah.Scene modeling using co-clustering.In Computer Vision, 2007.ICCV 2007. IEEE 11th International Conference on, pages 1–7. IEEE, 2007.

[41] Lingqiao Liu, Lei Wang, and Xinwang Liu.In defense of soft-assignment coding.In Computer Vision (ICCV), 2011 IEEE International Conference on, pages 2486–2493, Nov 2011.

[42] Stuart P Lloyd. Least squares quantization in pcm. Information Theory, IEEE Transactions on, 28(2):129–137, 1982.

[43] David G Lowe. Object recognition from local scale-invariant features. In Computer vision, 1999.The proceedings of the seventh IEEE international conference on, volume 2, pages 1150–1157.IEEE, 1999.

[44] MeenaMahajan, PrajaktaNimbhorkar, and KasturiVaradarajan. The planar k-means problem is np-hard. In WALCOM: Algorithms and Computation, pages 274–285. Springer, 2009.

[45] SubhransuMaji, Alexander C Berg, and Jitendra Malik.Classification using intersection kernel support vector machines is efficient. In Computer Vision and Pattern Recognition, 2008. CVPR 2008.IEEE Conference on, pages 1–8.IEEE, 2008.

[46] MATLAB. version 7.10.0 (R2010a). The MathWorks Inc., Natick, Massachusetts, 2010.

[47] KrystianMikolajczyk and CordeliaSchmid. Scale & affine invariant interest point detectors. International journal of computer vision, 60(1):63–86, 2004.

[48] David Nister and HenrikStewenius.Scalable recognition with a vocabulary tree.In Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on, volume 2, pages 2161–2168. IEEE, 2006.

[49] FlorentPerronnin. Universal and adapted vocabularies for generic visual categorization. Pattern Analysis and Machine Intelligence, IEEE Transactions on, 30(7):1243–1256, 2008.

[50] FlorentPerronnin and Christopher Dance.Fisher kernels on visual vocabularies for image categorization.In Computer Vision and Pattern Recognition, 2007. CVPR’07. IEEE Conference on, pages 1–8.IEEE, 2007.

[51] FlorentPerronnin, Jorge Sánchez, and Thomas Mensink.Improving the fisher kernel for large-scale image classification. In Computer Vision–ECCV 2010, pages 143–156. Springer, 2010.

[52] James Philbin, Ondřej Chum, Michael Isard, Josef Sivic, and Andrew Zisserman. Object retrieval with large vocabularies and fast spatial matching. In Computer Vision and Pattern Recognition, 2007.CVPR’07.IEEE Conference on, pages 1–8.IEEE, 2007.

[53] James Philbin, Ondřrej Chum, Michael Isard, Josef Sivic, and Andrew Zisserman.Lost in quantization: Improving particular object retrieval in large scale image databases.In Computer Vision and Pattern Recognition, 2008.CVPR 2008.IEEE Conference on, pages 1–8.IEEE, 2008.

[54] Ali S Razavian, HosseinAzizpour, Josephine Sullivan, and Stefan Carlsson. Cnn features off-the-shelf: an astounding baseline for recognition. In ComputerVision and Pattern Recognition Workshops (CVPRW), 2014 IEEE Conference on, pages 512–519. IEEE, 2014.

[55] Bryan C Russell, Antonio Torralba, Kevin P Murphy, and William T Freeman. Labelme: a database and web-based tool for image annotation. International journal of computer vision, 77(1-3):157–173, 2008.

[56] CordeliaSchmid, Roger Mohr, and Christian Bauckhage.Evaluation of interest point detectors. International Journal of computer vision, 37(2):151–172, 2000.

[57] Pierre Sermanet, David Eigen, Xiang Zhang, Michaël Mathieu, Rob Fergus, andYannLeCun. Overfeat: Integrated recognition, localization and detectionusing convolutional networks. arXiv preprint arXiv:1312.6229, 2013.

[58] M. Sert and H. Ergun. Video scene classification using spatial pyramid based features. In Signal Processing and Communications Applications Conference (SIU), 2014 22nd, pages 1946–1949, April 2014.

[59] ShaiShalev-Shwartz, Yoram Singer, Nathan Srebro, and Andrew Cotter.Pegasos: Primal estimated sub-gradient solver forsvm. Mathematical programming, 127(1):3–30, 2011.

[60] Karen Simonyan and Andrew Zisserman.Two-stream convolutional networks for action recognition in videos. In Advances in Neural Information Processing Systems, pages 568–576, 2014.

[61] Karen Simonyan and Andrew Zisserman.Very deep convolutional networks for large-scale image recognition.arXiv preprint arXiv:1409.1556, 2014.

[62] J. Sivic and A. Zisserman. Video google: a text retrieval approach to object matching in videos. In Computer Vision, 2003.Proceedings. Ninth IEEE International Conference on, pages 1470–1477 vol.2, Oct 2003.

[63] KhurramSoomro, Amir RoshanZamir, and Mubarak Shah.Ucf101: A dataset of 101 human actions classes from videos in the wild. arXiv preprint arXiv:1212.0402, 2012.

[64] Christian Szegedy, Wei Liu, YangqingJia, Pierre Sermanet, Scott Reed, DragomirAnguelov, DumitruErhan, Vincent Vanhoucke, and Andrew Rabi-novich. Going deeper with convolutions.CoRR, abs/1409.4842, 2014.

[65] TinneTuytelaars and KrystianMikolajczyk. Local invariant feature detectors: Asurvey. Found. Trends.Comput.Graph. Vis., 3(3):177–280, July 2008.

[66] TinneTuytelaars and CordeliaSchmid.Vector quantizing feature space with a regular lattice.In Computer Vision, 2007.ICCV 2007. IEEE 11th International Conference on, pages 1–8. IEEE, 2007.

[67] Koen EA Van de Sande, Theo Gevers, and Cees GM Snoek. A comparison of color features for visual concept classification. In Proceedings of the 2008international conference on Content-based image and video retrieval, pages 141–150. ACM, 2008.

[68] Koen EA Van De Sande, Theo Gevers, and Cees GM Snoek.Evaluating color descriptors for object and scene recognition. Pattern Analysis and Machine Intelligence, IEEE Transactions on, 32(9):1582–1596, 2010.

[69] Jan C van Gemert, Jan-Mark Geusebroek, Cor J Veenman, and Arnold WM Smeulders.Kernel codebooks for scene categorization. In Computer Vision–ECCV 2008, pages 696–709. Springer, 2008.

[70] Jan C Van Gemert, Cor J Veenman, Arnold WM Smeulders, and Jan-Mark Geusebroek.Visual word ambiguity. Pattern Analysis and Machine Intelligence,IEEE Transactions on, 32(7):1271–1283, 2010.

[71] ManikVarma and Andrew Zisserman.Classifying images of materials: Achieving viewpoint and illumination independence. In Computer Vision—ECCV 2002, pages 255–271. Springer, 2002.

[72] A. Vedaldi and B. Fulkerson.VLFeat: An open and portable library of computer vision algorithms. 2008.

[73] A. Vedaldi and A. Zisserman.Efficient additive kernels via explicit feature maps.Pattern Analysis and Machine Intelligence, IEEE Transactions on, 34(3):480–492, March 2012.

[74] Jingdong Wang, Jing Wang, QifaKe, Gang Zeng, and Shipeng Li. Fast approximate k-means via cluster closures. In Multimedia Data Mining and Analytics, pages 373–395. Springer, 2015.

[75] Jinjun Wang, Jianchao Yang, Kai Yu, FengjunLv, T. Huang, and Yihong Gong.Locality-constrained linear coding for image classification. In Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on, pages 3360–3367, June 2010.

[76] Limin Wang, Zhe Wang, Wenbin Du, and Yu Qiao. Object-scene convolutional neural networks for event recognition in images.arXiv preprint arXiv:1505.00296, 2015.

[77] Limin Wang, YuanjunXiong, Zhe Wang, and Yu Qiao. Towards good practices for very deep two-stream convnets. arXiv preprint arXiv:1507.02159, 2015.

[78] John Winn, Antonio Criminisi, and Thomas Minka. Object categorization by learned universal visual dictionary. In Computer Vision, 2005.ICCV 2005. Tenth IEEE International Conference on, volume 2, pages 1800–1807. IEEE,2005.

[79] Jianxin Wu and James M Rehg. Beyond the euclidean distance: Creating effective visual codebooks using the histogram intersection kernel. In ComputerVision, 2009 IEEE 12th International Conference on, pages 630–637. IEEE, 2009.

[80] Jianchao Yang, Kai Yu, Yihong Gong, and T. Huang. Linear spatial pyramid matching using sparse coding for image classification. In Computer Vision andPattern Recognition, 2009.CVPR 2009.IEEE Conference on, pages 1794–1801, June 2009.

[81] Hao Ye, Zuxuan Wu, Rui-Wei Zhao, Xi Wang, Yu-Gang Jiang, and XiangyangXue. Evaluating two-stream cnn for video classification.arXiv preprint arXiv:1504.01920, 2015.

[82] Matthew D. Zeiler and Rob Fergus. Visualizing and understanding convolutional networks.CoRR, abs/1311.2901, 2013.

[83] ShengxinZha, Florian Luisier, Walter Andrews, NitishSrivastava, and RuslanSalakhutdinov.Exploiting image-trained cnnarchitectures for unconstrained video classification.arXiv preprint arXiv:1503.04144, 2015.

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

[85] Bolei Zhou, AgataLapedriza, Jianxiong Xiao, Antonio Torralba, and AudeOliva.Learning deep features for scene recognition using places database. In Advances in Neural Information Processing Systems, pages 487–495, 2014.

[86] Lei Zhu, Al Bing Rao, and Aldong Zhang. Theory of keyblock-based image retrieval. ACM Transactions on Information Systems (TOIS), 20(2):224–257,2002.