ParisTech
INSTITUT DES SCIENCES ET TECHNOLOGIES
PARIS INSTITUTE OF TECHNOLOGY

TELECOM ParisTech

2014-ENST-0040

EDITE - ED 130

Doctorat ParisTech

T H È S E

pour obtenir le grade de docteur délivré par

TELECOM ParisTech

Spécialité « Signal et Images »

présentée et soutenue publiquement par

Usman Farrokh NIAZ

le 08 juillet 2014

Cutting the Visual World into Bigger Slices for Improved Video Concept Detection

Amélioration de la détection des concepts dans les vidéos par de plus grandes tranches du Monde Visuel

Directeur de thèse : **Bernard Mérialdo**

Jury

M. Philippe-Henri Gosselin, Professeur, INRIA	Rapporteur
M. Georges Quénot, Directeur de recherche CNRS, LIG	Rapporteur
M. Georges Linares, Professeur, LIA	Examinateur
M. François Brémond, Professeur, INRIA	Examinateur
M. Bernard Mérialdo, Professeur, EURECOM	Encadrant

TELECOM ParisTech
école de l'Institut Télécom - membre de ParisTech

T H È S E

Abstract

Visual material comprising images and videos that surrounds us is growing ever so rapidly over the internet and in our personal collections. This necessitates automatic understanding of the visual content to conceive intelligent methods for correctly indexing, searching and retrieving images and videos. We have entered an age where the reliance of humanity on computing has never been felt so strongly before. However we still are not able to make our machines see and understand the world around us like we humans do. Initial successes in Computer Vision after the conception of the field some 50 years ago encouraged the then scientist to predict the successful resolution of the problem of automatic understanding of visual material with in some years. 5 decades on and our computers still achieve partial success in truly detecting the content depicted in an images, let alone a video.

The difficult task of automatic indexing and retrieval of videos based on their content is abundantly addressed by academic and industrial exploration. Decades of research on text based automatic retrieval and then content based methods to index videos has led to the development of ingenious systems. These complex systems mainly utilize image analysis and statistical learning tools and more or less comprise of several intricate steps to make the machine able to recognize the content. However unlike text based search, automatic understanding of the video content is far from solved with our machines today using the current methods. A vast amount of information is extracted from videos some of which is used efficiently and cleverly to achieve the target being mindful to the scalability and complexity of the system.

Understanding the importance of content based indexing and retrieval and after doing extensive research of the state of the art in the field we feel that the automatic system can be improved in many ways. We predominately target the information that is *not* used when building such a system, whilst its already there. This thesis aims at improving the automatic detection of concepts in the internet videos by exploring all the available information and putting the most beneficial out of it to good use. We look at it in the sense of cutting thicker slices from the vast world of visual information to build our categorization system.

Our contributions target various levels of the concept detection system in videos and propose methods to improve the global performance of the inter disciplinary system. The standard system extracts low level information from the videos and givens a decision about the presence of an entity in the video frame after a series of computing steps. Our contributions start with improving the video representation model employing some existing information from the Visual World. We then devise methods to incorporate knowledge from similar and dissimilar entities to build improved recognition models. Lastly we contrive certain semi-supervised learning methods to get the best of the substantial amount of unlabeled data.

Acknowledgements

During the latter half of 2010, owing to the unfolding of a series of unfortunate events, I had become quite clueless and disheartened. Fortunately enough, Professor Bernard Mérialdo hired me to do PhD research under his supervision, I was rescued. With computer vision being my research passion, finding a PhD thesis in the topic with an excellent professor was like a dream come true. I would like to start these acknowledgments by expressing my profound gratitude to professor Mérialdo for his continuous support, his careful guidance, his positive attitude, his advices, his patience and for some of the morale boosting sessions we had.

I would also like to thank Professor Lionel Prévost (UPMC, UAG) as he was always a source of inspiration for me for the very short period of time we worked together and played a very important role in getting me the PhD position at Eurecom.

I felt at home in Eurecom right from the start. With a wonderful mix of people from beautiful and fabulous countries, having amazing and friendly personalities I felt welcomed since my first day. In the old campus we were five in a big room, including a Turkish, a Tunisian, an Italian and then a Cameroonian, an Argentinean and me, a Pakistani. The seniors were always ready to help and offered helpful advices. Then there was always someone bringing special treats from their respective countries, to share with everyone. Birthdays, pot de thèses and 'MMtalks' were another medium of getting everyone together. I wish to thank Claudiu and Miriam for their guidance and specially Miriam for her reassurances of, 'don't worry, Its going to be fine'. I can never forget the discussions I used to have with Claudiu about 'where we stand' (and where should we stand). I would like to thank Simon, Hajer, Neslihan, Huda, Federico, Gislain, Ravi, Giovanni, Leela, Christelle, José and of course my Pakistani friends in Eurecom Sabir and Imran. The role of all the secretaries, HR and the guys at IT help-desk, with whom we PhD students had frequent interaction, can not go unacknowledged. It is their excellent service, helpful attitude and proper guidance that makes them stand apart, making your life wonderful at Eurecom. Finally, life bound to be wonderful if you are working and living in one of the most enviable places in the world, specially Europe, the French Riviera. In the south of France enjoying around 300 days of sunlight a year gives one enough time and opportunities to relax, stay healthy and come to work with a fresh mind.

During my PhD at Eurecom I worked on the ANR's SUMACC project in collaboration with other expert partners. During the project meetings I had the chance to meet brilliant people and learn from them including professor Georges, Christophe, Claude, Clément, Mohamed and Richard.

Words aren't enough to encompass the immense gratitude i owe to my wonderful parents. All I can do is thank them for always treating my worries as their top priorities. They've not only been my mentors but my friends and have always been a source of motivation,

inspiration and guidance for me. I attribute much of the success in my life to my mother's prayers and her constant guidance. Not a single conversation with her ever ends without me learning something new, something useful. My father on the other hand see's things on the lighter side and encourages us to do the same as well. I believe it is the perfect parental combination one could hope for and it has worked perfectly for me (and my siblings) even though I was away from home for a long time, completing my higher studies.

And finally I cannot conclude without thanking the most important person who was there with me during a good part of my PhD, my wife Bushra. Since I've gotten to know her she's been a constant support in every aspect of my life. She's been with me through all the ups and downs, that come your way specially in pursuit of a PhD, far away from home, supporting me and proudly standing by my side, and I know that she'll always be there for me. She has always believed in me and cherished me with her amazing and supportive words and her beautiful smile.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

They say a picture is worth a thousand words but this is a mere understatement in computer vision. In fact, for vision scientists, a picture may contain millions of *visual* words. These visual words are built using various visual features capturing important measurements from images (color, texture, spatial arrangements, local patches, global statistics, etc...). All this multitude of information is used to help computers capture the understanding of the content of images. It is an attempt to make machines see and understand the world like we do. However this multi-megabytes of information still achieves partial success and the recognition of content in the images continues to elude our intelligent machines.

Video analysis is a scaled-up version of the same problem where a series of images come into play bringing along the relation between frames and temporal dependencies. The core, however, still constitutes of analysis at frame level. Automatic image/video analysis consists of tasks like categorization, retrieval, copy detection, event detection etc., at the helm of which lies the ability to recognize the visual content based on image semantics. Manipulating images and video documents for automatic analysis is among the most difficult challenges faced in computer vision [1]. However the need for automatic recognition of visual content has never been felt more strongly before with the exponential increase in the amount of visual information spreading over the internet. Roughly 350 million new photos are uploaded to Facebook every day [2], and the number for Flickr is between 20 to 40 million per day [3]. Moreover Facebook boasts more than quarter of a trillion photos uploaded to their web site. YouTube has an astounding 100 hours of new video uploaded every minute. The amount of users using these services is growing by day and this online medium of communication is now matching the importance of broadcast Television. This staggering amount of new information and its importance to users calls for reliable and efficient methods to analyze the visual content in order to develop methods to automatically search, index and browse these large databases.

In this introduction we start with explaining the pursued Video Concept Detection task which is also referred to as Semantic Indexing. We follow with an overview of the major

steps involved in achieving concept detection. Further we analyze the opportunities for improvements in the detection pipeline and present our motivations to work in the selected directions. The introduction lists, at the end, our contributions to the video concept detection pipeline.

1.1 Video Concept Detection (Semantic Indexing)

For years scientific research in image and video retrieval and indexing has been dominated by text or concept based approaches where the *metadata* or associated text based descriptions like title, an article or narrative and tags etc. are used to retrieve multimedia content from the internet. Practically this *is* the popular method these days to get images from the popular search engines and look for videos on video sharing services like YouTube and Dailymotion. Since the dawn of the century though research focus has been shifted to directly understanding the content of the multimedia documents and using the extracted information to build retrieval models. Textual descriptions associated with or surrounding the multimedia content on the internet are not always reliable. These are usually subjective to the uploader, personalized, misleading, incomplete and sometimes do not exist at all. Moreover annotating a large video database from scratch with manual human effort is laborious and could suffer from the similar incompleteness and bias. The content on the other hand is always right and we can safely rely on its authenticity, provided its correct use.

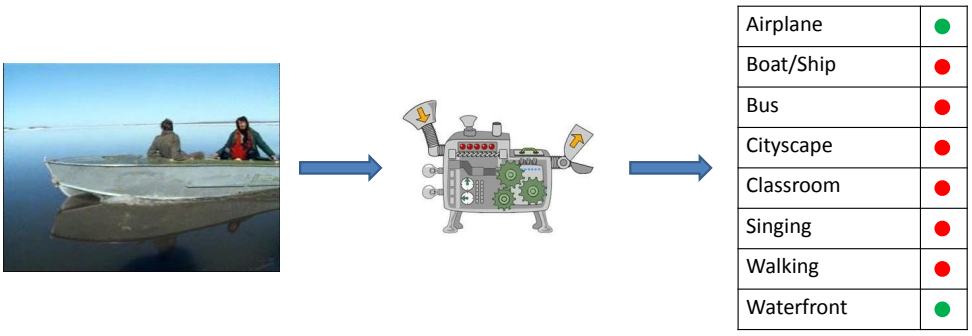


FIGURE 1.1: Video Concept Detection

Video Concept Detection or video categorization aims at automatically describing a video with semantic concepts that correspond to the content of the video, figure 1.1. These **semantic** concepts are high level descriptions of the video that directly depict the key information present in the content. The semantic concept can consist simply of labels like objects or people, can be depicted by a scene or can comprise of a situation with a complex interaction of different entities. The solution assigns a probability of presence to a concept or a label in the video frame (e.g. the video contains a "Bus" for sure and is less likely to contain a "Helicopter hovering"). The probability is assigned by a classifier which is trained for that concept. This classification model is built for each concept separately. The classification is

not done for each frame of the video, rather a set of *keyframes* are extracted for each video [4, 5]. These **keyframes** are *representative* of the video content. Video is first segmented into *shots* where a new shot location can be given with the video metadata or can be automatically detected [5]. Usually a single keyframe is then extracted from each video shot which is the most informative of the shot frames. Since analyzing videos with keyframes offer a convenient and effective alternative to analyzing video as a whole [4–6] we opt to work with keyframes throughout this thesis.

A semantic concept does not have a fixed description most of the times and there exists plenty of within class variations. The detection framework has to capture this intra-class variability when building classification models for concepts. The video concept detection system which starts to work with the raw pixel data from images and assigns probabilities to test images is very intricate and is composed of quite a few processing steps. These steps are detailed in the next section.

1.2 Video Concept Detection Pipeline

A video categorization system comprises a complex structure of more or less sequential elements which span over a number of scientific disciplines including image and signal processing, statistical analysis, mining and machine learning to name a few. Figure 1.2 presents the major components of the framework where we start with images (keyframes) assuming that video was already segmented into shots. Building different concept detection models and using them employs largely all the steps of the framework and only differ in the functionality of some of the stages.

The system extracts a number of different visual features from the provided images at the stage 1 of the framework. These raw features are usually not used directly and are rather transformed to an aggregated image specific representation in the stage 2 of the pipeline for further processing. A number of annotated images is usually available to learn the model which classifies the concept images from the rest. This makes the 3rd stage where the classification model is used to find predictions on the test images which depicts the prospects of existence of the concept in the corresponding video. The fourth stage is an additional but very useful step where predictions from different models are combined to unite the powers of the individual learners. We briefly describe each of the stages of the pipeline below before looking into the possible rooms for improvement in the components of the pipeline.

Stage 1: Feature Extraction and Description

An image in a typical video is composed of tens of thousands of pixels that are mere numerals but an enormous amount of information can be extracted from a single image. This step is the foundation of the detection or retrieval framework as features are expected to capture the essence of what the image represents. Humans can, most of the times, figure out the concept depicted in the image by just taking a look. The features are expected to encapsulate this

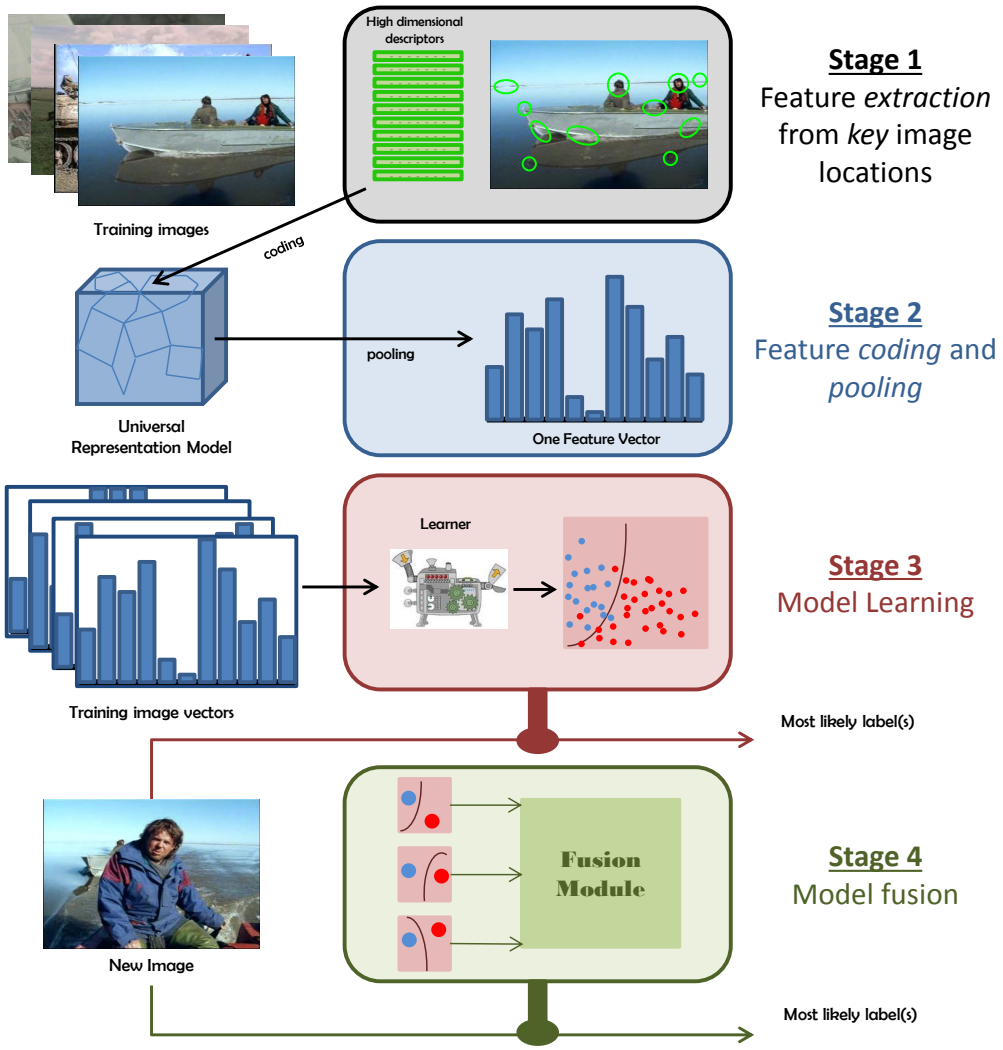


FIGURE 1.2: Video Concept Detection Pipeline.

essence of the image for machines. They capture the semantics of the image to make the machine understand the similarity or dissimilarity between the images.

A video frame can be described globally or can be bisected into regions before building descriptions for each region. These descriptors are called the *visual* features of the image. While **global** features capture the qualities or properties of the image on the whole the **local** features work on a local neighborhood of the image to gather information about the targeted pixel. Global features can be computed for the whole image or parts of image by first segmenting the image into predefined parts and then building the feature for each segment. They describe global properties of the image (segment) like color, gist or some sort of spacial information. Local features may be described for predefined image locations and are sometimes only extracted for specific image points. These points are expected to contain rich information compared to the rest of the image. Methods are employed to find these critical points that are based on some maximization criterion. Statistics from the local

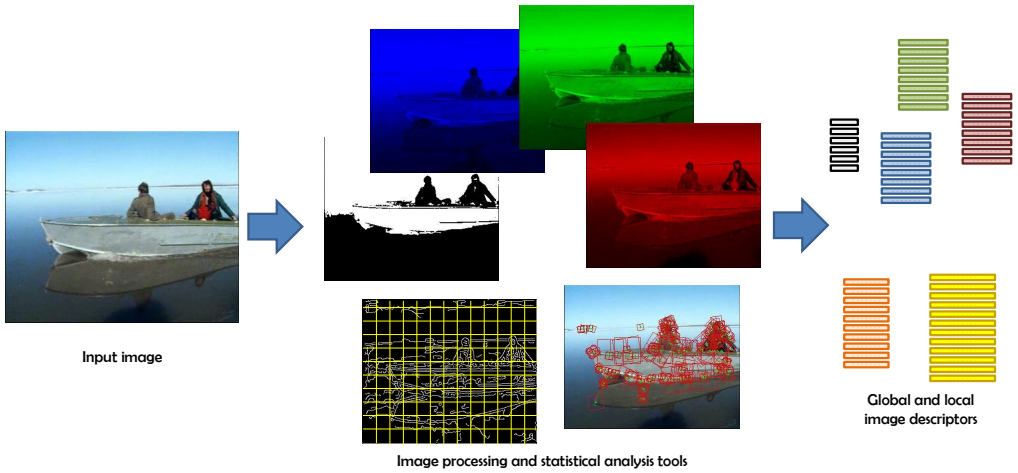


FIGURE 1.3: Global and local feature extraction example.

neighborhood or *patch* are calculated to describe that part of the image. The number of features thus may vary from image to image if this kind of patch extraction is done. Local statistics include extracting properties like information about shape and geometry, the local structure, texture etc.. Figure 1.3 depicts the idea of extracting global and local features from an example image. Note that size and number of features extracted are different for different feature types.

The features extracted should be relevant to the specific task of concept detection and should be robust. So if the appearance of e.g. an object changes in the image the features should still be able to capture the relevant and necessary information. They should also resist change to the accidental distortion of images from illumination variation, camera movement and other forms of distortions. A good knowledge of image processing is required to build such features.

Usually the features of a specific type are used to build the rest of the system but sometimes different types of features can be merged pre-hand to perform the task. More discussion on the use of the features is to follow.

Stage 2: Feature Coding and pooling

The different kinds of features extracted in the previous stage can be directly used to represent an image but their use is prohibitive due to their sheer size and huge number. This is more often the case for local image descriptors that are extracted at numerous image locations. Furthermore the number of these features extracted from the images could be different as well which prohibits a unified representation for all the images. That is why the features are usually *summarized* into a single high dimensional feature vector of fixed length, figure 1.4.

To construct the unified representation first the features from the image are transformed or **coded** into a set of values. This projection of the feature onto the new representation has certain properties: two features that are close are encoded into the same representation, the representations are compact and most of them are zero for an image [7]. Some examples

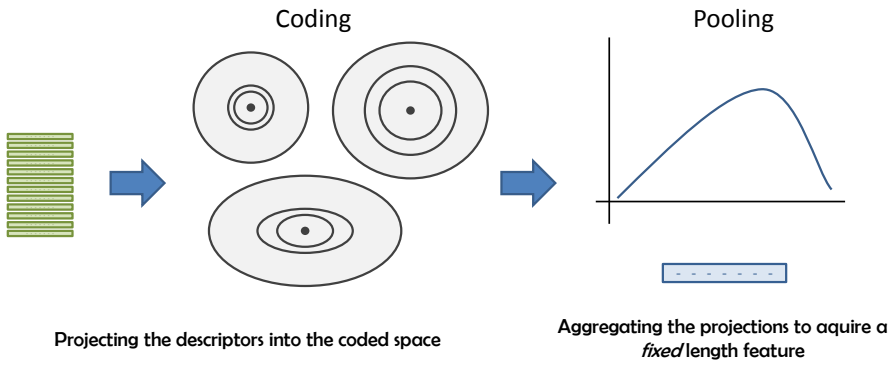


FIGURE 1.4: Coding and pooling to generate the summarized feature for an image.

of coding include vector quantization [8], soft assignment [9, 10] and probability estimation for GMMs to generate Fisher vectors [11]. The new representation or the codebook is constructed using features from some training images. After each image feature is coded, **pooling** aggregates the coded features into a single value for each of the codes. This results in a vector of values with length equal to the number of coded representations. Pooling can vary from averaging and taking the maximum value to parameter adaptation in the case of Fisher vectors [11, 12].

The benefit of these fixed length vectors is that they can be directly fed into a discriminative classifier and a model for the concept can be built. Moreover the reduced size helps in fast learning and efficient prediction of the test examples. On the other side of the picture the true discriminative power of the features that capture the image dynamics from specific locations is somewhat lost as the local features are aggregated. Extreme values may be diminished due to averaging or maximizing and geometrical structure can be no longer used. Refinements can be done to include some useful information that hints at some of the lost dynamics of the original image. The size of the coded representation can be increased to better distinguish the image features at the expense of increased training and prediction time and the risk of overfitting the parameters to the training images.

Stage 3: Model Learning

This stage is the brain of the pipeline where machine learns to correctly identify examples of a concept. This necessitates the presence of some annotated images that must be used for learning the model for a semantic concept. These images are treated as the positive examples and are used to train the classification model to distinguish them from the rest of the world. To be able to achieve this intelligent behavior a great deal of number crunching is required and usually this step takes the most of the time in the pipeline. This time increases linearly with the number of concepts as in practice a separate model is learned for each concept.

State of the art in learning such kind of a model is mainly divided into two parallels: discriminative learning and generative learning, figure 1.5. **Discriminative** classifiers learn

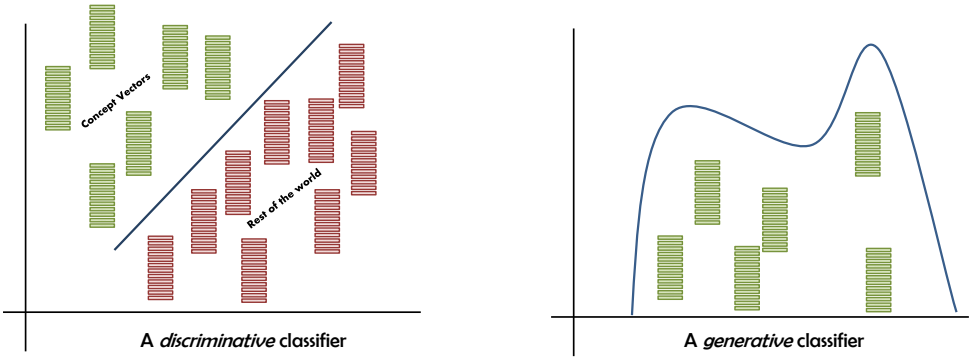


FIGURE 1.5: (a) A discriminative classifier separating positive and negative examples. (b) A generative classifier adapted to the positive concept examples.

to map the input examples x to the class labels y [13]. This is achieved by modeling the conditional probability $P(y|x)$ directly from the input examples x and their respective labels y. On the other hand the **Generative** classifiers learn the joint probability distribution $P(x, y)$. Bayes Rule is used to find the conditional probability $P(y|x)$ to predict the most likely label. So to resume a generative model tries to learn how the data was generated, and can be used e.g. to generate more samples of the data, while the discriminative model does not care about the data generation and learns to categorize directly.

The quality of the classifier learned depends not only on the type of learning method used and its parameters but also on the kind of feature representation selected and the quality of positive and negative instances acquired for training.

We will give some examples of popular generative and discriminative models used in the concept detection and retrieval frameworks in the next chapter where we discuss state of the art but in this thesis we stick to discriminative form of classification. **Stage 4: Fusion** Although the concept detection pipeline pretty much completes at the previous step, combining a variety of learners on different descriptors is always a fruitful option. Fusion allows learners to be built for different types of features independently and only then be combined when generating the final output concept probability. The different types of features can include other modalities like audio and textual information along with visual features. This type of fusion is called decision or late fusion as it is done at the decision level as shown in the figure 1.6. A feature level fusion can also be envisaged and pops up in the literature now and then but is less popular due to the heterogeneous combination and the increased size of the combined feature. Chapter 5 highlights some important contributions from the field of multimodal fusion.

Finally a feedback loop is employed sometimes to refine the set of training examples and add some new useful examples to build a stronger classification model.

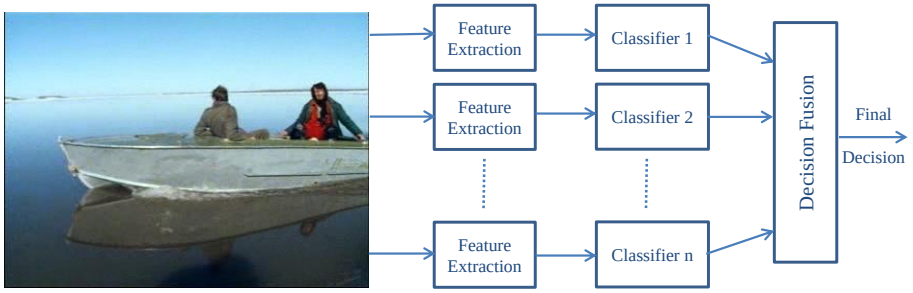


FIGURE 1.6: Classifier fusion at the decision level.

1.3 Motivation

The information extracted from video frames at pixel level are low level descriptions of the image and this does not translate directly to the semantic meaning of the image. There exists thus a *Semantic Gap* [14] between the low level image description that the computer vision system uses, to learn about the higher level semantics that the image represents. There are various reasons for the existence of this gap. Foremost is the difference in the human perception of a concept and the representation suitable for our computers to understand. Then there is subjectivity [15, 16], i.e. when different users perceive similar content differently. Lastly the semantic gap is widened by the within class diversity for many concepts. To highlight this problem consider an image of a chair placed inside an office and another image of a chair outside in a lawn. Though both images contain the concept *Chair*, the low level pixels would not agree for the most part of the two images. Research in automatic video content understanding has been focused to reduce this semantic gap.

The performance of a detection system depends on all the factors that make up the detection pipeline. For each of these steps we list below the main opportunities of improvement.

- **Features extracted at pixel level**
  As the basis of the detection pipeline this phase receives considerable research attention in order to capture the most useful information with a manageable size from the images.

- **Selecting and building the most informative mid-level representation**
  Summarizing the visual features is the most sensitive step in the pipeline as most of the raw low level information is lost here. Research in this area focuses on building comprehensive mid-level representations that successfully capture the image semantics.

- **Refining the representation**
  Research has also been done to refine the mid-level representation to try to bridge the semantic gap by adding e.g. contextual, spatial or label specific information.

- **Quality of annotations/ selecting the best examples**
  The tedious work of extracting and summarizing features is of no use if representative examples are not used for concept model learning. Selecting the best examples for training and refining them with feedback is important to the performance of the system. Moreover there always exist great amount of unlabeled examples to be discovered. New instances should be added to add diversity while increasing performance.

- **Finding the best decision boundary**
  In the end it all comes to the classifier which distinguishes the positive instances from the negatives. Research is evolving in this area as well to build classifiers that generalize well on test examples and have lower complexity.

- **Fusing the best classifiers effectively**
  Research continues to find new methods to select the best classifiers among the pool of multimodal classifiers and combine them in the most effective way.

1.3.1 Making the Slice Bigger

With the enormous amount of visual information extracted from the images, the various coding and pooling methods and the variety of strong classification techniques available, not everything is used to build the categorization system. If too many features or parameters are used to train the models for concepts there is a risk of overfitting on the training examples. Moreover this would require loads of computing power and time as the feature representations are high dimensional and the current processing power limits the extravagance use of this visual information.

We can look at all this information as making a *Visual World*, figure 1.7, and usually a video concept detection system only takes a slice of this world with a number of annotated images, certain features and a classification algorithm that learns a model using this information. The amount of information going into the slice goes from raw image level information to classification decision which justifies the pyramidal structure. What we target in this thesis is how to make this slice bigger by exploring the Visual World in order to add extra and useful information with little effect on complexity of the task to improve video concept detection. So instead of cutting the standard slice from the Visual World we cut a bigger slice where we include additional information at various levels of the concept detection pipeline 1.2. Note that in terms of this Visual World, classifier fusion can be regarded as combining various slices.

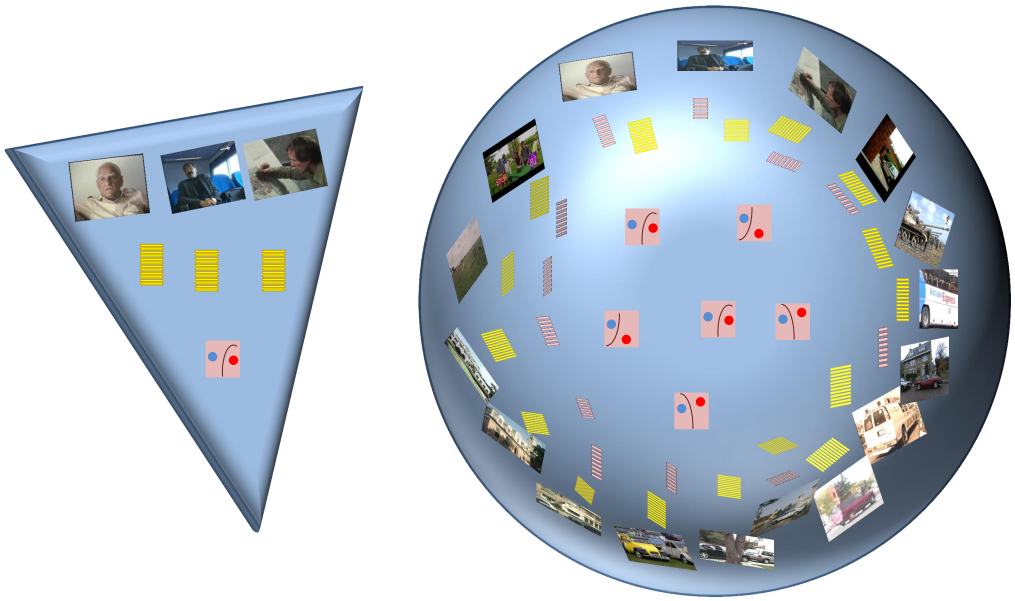


FIGURE 1.7: The Visual World with numberous images, feaures and classifiers.

1.4 Our Contributions

Our contributions target two major areas of the concept detection pipeline: feature coding and pooling, and Model learning where we focus on finding useful training examples. The following paragraphs summarize our contributions.

- We use state of the art features extracted from the images and work on their transformation into the mid-level representation. We use label information while building this representation in order to add some image and concept specific information into the feature.
- We also enhance the standard image representation by encoding the feature with transformation specific information.
- To improve the classification performance for a concept we explore examples of other related concepts in order to add useful and diverse information. The idea is that common or close concepts share information among themselves and it should be put to use when learning the classification models.
- We extend the idea above to look at concepts that are not so close, rather that are different, to learn from the dissimilarity between concepts.

- Finally we try to explore the unlabeled examples in a semi supervised setting to increase the amount of annotated data and use the best information out of it to learn the concept models.

This concludes the introductory chapter of this manuscript which is followed by a brief review of the state of the art methods in video concept detection and retrieval in chapter 2. We limit our discussion to methods important to the concept detection pipeline. Each of the rest of the chapters also include some review of popular methods as it makes more sense to describe the work that inspired us with the corresponding contributions. Chapter 3 includes the first of our contributions where we try to improve the mid-level video representation. In chapter 4 we base our work on the similarities and differences between concepts to try to find useful new examples to perform improved concept detection. Chapter 5 explores the unlabeled examples and presents criteria to select the most useful new examples. Finally in chapter 6 we summarize our contributions and draw conclusions. We also list possible future avenues to explore for automatic analysis of video content especially leading to video concept detection.

Chapter 2

State of the Art

Automatic analysis of video content is among the hardest problems faced by the computer vision and multimedia community. It receives a significant amount of dedicated research attention from academic and industrial research teams all over the globe due to its importance in the growing digital world. As pointed out in the introduction the research in this field is truly interdisciplinary and integrates works ranging from image and signal processing to statistical machine learning. Consequently it is imperative to review the state of the art in the vast field of visual content analysis specifically targeting video concept detection and retrieval. This chapter lists important contributions to each of the stages of the concept detection pipeline starting from raw feature extraction to fusing the high level decisions.

In the early 90s with the advent and increase of digital media and the availability of cheap media capturing devices research interest grew in the content based retrieval field [14, 16, 17]. Research exploded after the dawn of the millennium when Google launched its text based image search. It was evident that directly understanding the content of the images was the new target. Around the same time National Institute of Standards and Technology (NIST) also jumped into the action and later in 2003 formed a dedicated research track allocated to video analysis based techniques called TRECVID [18].

In terms of scalability of the content based image/video search problem the databases used have literally gone from indexing thousands [19, 20] to millions [12, 21–25] and now billions [26] of images in the last 20 years [17].

In the year 2000 Smeulders et al.[14] presented a review on content based image retrieval citing about 200 references. The problem of automatically understanding the visual content has received continued attention in the last 14 years with thousands of available works and many dedicated conferences and workshops. We only select some of those works that we feel are the most representative and cover more or less the stages of the concept detection pipeline presented in the figure 1.2.

2.1 Feature Extraction

Visual feature extraction converts an image to a mathematical representation and it forms the basis of any visual content analysis system. The features should capture enough information to differentiate the class instances from the non-class instances. Since the features are extracted from raw pixel values this step is susceptible to noise that may come from camera motion, variations in lighting, change in scale and viewpoint, occlusions, lower resolution, quality of the frame extracted or the movement of the objects in the frame themselves. The features extracted in this stage should be robust to all these noise and variations and should be compact. There is no **winner** when it comes to capturing the visual information due to the complex composition of data, the intra class diversity and the nature of the semantic concept to be detected. The features or image descriptors are mainly divided into two types: global and local features.

2.1.1 Global Features

Global features describe the properties of the image as a whole summarizing the information into one compact description. They are faster to compute and are highly scalable compared to their counterpart but lack enough discriminatory information. Color seems to be an important and intuitive global choice when trying to predict the semantic concept in an image. Globally dominant colors can give indications about the concept like if an image contains large components of blue and green it is likely to be an outdoor scene [27].

This is precisely true as **color features** have been there since the conception of the field of image analysis and are being used till date. The most straightforward use of color is in building the color histograms from the pixel values in the RGB space [28, 29]. To make the color histograms invariant to changes in light intensity and shadows, normalization is performed [30, 31]. Color spaces other than the omnipresent RGB have also been used explicitly. Weijer et al. [32] build color histograms in the HSV space where the H bins are weighted by the saturation making the representation robust to changes in scale and shift.

Hays and Efros [33] use the L*a*b* space to build 784 dimensional histograms with 4, 14 and 14 bins in the L, a and b respectively. Chang et al. [34] build a 12 bin histogram for 11 colors and one outlier bin. They extend their feature by building histograms for each color at finer resolutions along with mean and variance of the HSV channels. [35] extracts average color components of the LUV space (L:luminance, UV chrominance) of 16 pixels from the image divided into 4 X 4 blocks.

Color moments [36–38] are defined on the distribution of the RGB triplets. These are usually extracted up to the 2nd order, are rotation invariant and higher order moments also contain spatial information. *Color map* reflects the user's search intention with a small feature by allowing the user to indicate the spatial distribution of colors in the desired images [39].

Holistic representations capture dominant spatial structure of a scene into a low dimensional feature from a scaled down image [24]. Olivia and Torralba [40] proposed the GIST feature which summarizes Gabor filter responses, which was further compressed using different strategies [23, 41]. GIST features have also been extracted from image regions by accumulating responses from filters at different orientations and scale [33, 42].

Texture features capture important properties like smoothness, coarseness or some pattern that occurs repeatedly throughout the image. Pioneer textural features were extracted by Tamura et al. [43] emulating the human visual perception. For outdoor scenes textural properties might be geographically correlated [33] e.g. the building in a city's skyline or vegetation. A universal texton dictionary is a popular method that summarizes the pixel responses to a bank of orientation filters [33, 44, 45]. [34] decomposes each image into four sub-images using discrete wavelet transform and build nine texture features with different compositions of the decomposed images. Li and Wang [35] perform Haar transform on the L component of the LUV image space for each 4 X 4 block of the image. They use average of the 3 out of 4 wavelet components to build their feature containing 3 values for each 4 X 4 block. Recently discriminatory texture information is also extracted using *Local Binary Patterns* (LBP) from image patches and aggregating them into a global descriptor [46]. LBP is extended by considering the distance between the center pixel and its neighbor for improved image retrieval [47] but with a higher dimensional feature. More recently [48] has extracted LBP for different color spaces.

Global **edge distribution** is another important piece of information gathered from images. Like color, edge histograms are the simplest representation capturing cumulative edge orientation information in images [49]. Wang et al. [50] build a global edge descriptor concatenating histograms from 5 over-lapping image regions.

2.1.2 Local Features

Global image descriptors sometimes do not capture the spatial relationship between image objects, often carry redundant information and are usually not invariant to photometric and geometric distortions. Local features are described at key image locations or *interest points* that are identified in the image and supposedly carry valuable local information [51]. These points can be predefined though, e.g. in the form of a dense grid over the image [52–57]. The descriptor then calculates statistics based on the local neighborhood usually depicting the shape of the region.

The purpose of keypoint detection is to sample a sparse set of regions in the image that are invariant to geometric and photometric transformations. SIFT (Scale Invariant Feature Transform) detects interest points using Difference of Gaussian (DoG) operator but is only partially invariant to illumination changes and affine transformations. More robust detectors are proposed by [58] based on Harris-Affine detection, the Maximally Stable Extremal Region

(MSER) detector [59] and [60] based on image edge and intensity based region detection. SURF (Speeded Up Robust Features) [61] detects interest points using a Hessian-Laplace based detector working on integral images to reduce detection time.

The SIFT descriptor [62, 63] posed as a breakthrough in describing local regions and is one of the most frequently occurring descriptors in research works to date. The feature induces a histogram on a local image region that was first detected using DoG operator and then described in terms of 8 dominant orientations of the gradient. The gradient is calculated in the neighborhood of 4 X 4 spatial grid and is thus a 128 dimensional feature. SIFT is robust to geometric and photometric transformations and is invariant to scaling, translation and rotation of the image. Gradient Location and Orientation Histogram (GLOH) extends SIFT to a 272 dimension histogram computing the descriptor for a log-polar location grid [58]. PCA-SIFT [64] increases the efficiency of retrieval by reducing the dimensions of the SIFT feature, while SURF [61] estimates the dominant orientation of the interest point neighborhood and accumulates the horizontal and vertical wavelet responses (w.r.t. the dominant orientation) of 16 regions around the interest pixel resulting in a 64 dimensional feature. Finally using a power normalization on the SIFT has also proven to be affective as in [65] and RootSIFT [66] where square rooted SIFT descriptors are stored.

Dalal and Triggs [67] came up with the Histogram of Oriented Gradients (HOG) descriptor which is among the most successful features used for visual recognition. HOG mainly follows SIFT methodology and captures the local distribution of edge orientations where the location information is generally lost. The descriptor aggregates the edge orientations of local patches that are regularly spaced into a histogram. Several extensions and improvements of HOG have also been successful where the feature is calculated on variable sized regions [68], calculated from segmented foreground and background images [69] and from signed and unsigned gradient information [70]. [71] also augment the high resolution local HOG feature with LBP statistics.

2.2 Feature Coding and Pooling

Local features capture discriminatory visual information from all over the image but they are mostly not used directly for image categorization. The low level descriptors are summarized into an intermediate representation usually through a two-step coding and pooling method. There is an unspoken rule for the successful working of such a representation: similar images should have similar representations but it should be discriminative enough to distinguish from instances of other classes. The low level visual information is somewhat lost but the new representation is compact, is of fixed size and is more robust to descriptor noise and small transformations in the image which helps the model to be generalized to test images. This intermediate representation encoding local visual characteristics is closer to the human visual perception of the image and thus helps reducing the semantic gap [8, 72].

The breakthrough for representing visual information was achieved when vector quantization was used to build the Bag of Words (BOW) model [8, 20]. A universal *codebook* or *visual dictionary* is build using features from the training images by unsupervised vector quantization. Features are coded to the nearest code or *visual word*. Sometimes only the most informative features are used to construct the dictionary [73]. Average [8, 20] or max pooling [74, 75] is used to build the mid-level BOW feature vector. Multiple assignments encodes a visual feature as belonging to the k nearest visual words. *Soft assignment* further relaxes the coding process and probabilistically assigns a feature to the closest visual words [9, 10, 76]. It can be seen as assigning a weight to each encoded feature. However authors in [77] revert back to assigning a local feature to a single visual word with a *degree of participation*, that is calculated based on the previous assignments of local features to that visual word.

Grauman and Darrell [78] presented a multi-resolution histogram to encode local features with out using a universal dictionary. The Bag of Words or Bags of Features models are orderless and do not contain any spatial or structural information about the image. Lazebnik at al. [54] inducted spatial information into the BOW framework by building histograms for specific image regions, which they named Spatial Pyramid Matching. Each feature is coded and pooled for each of the histograms and one representation is build for the image accumulating all the histograms. Locality-constrained Linear Coding (LLC) [74] improves the spatial pyramid approach by encoding each descriptor into a projected local coordinate system with a different basis. Sparse Coding [7, 79] also provides an alternative to vector quantization by relaxing the cardinality constraint and removing the *extreme sparseness* of the representation. Sparse coding is more general and is ironically usually less sparse than a vector quantized representation. Both these works use max pooling to obtain state of the art image classification results. Boureau et al. [80] use local pooling in the feature space to avoid *different* features encoded with the similar code (vector quantized, sparse coded) to be pooled together.

The research in image representation is further carried to build higher level representations where visual words are supposed to co-occur for certain objects or concepts. Considering different visual words as a single unit will reduce ambiguity that may arise when different intended meaning of visual words are considered. This notion of inter-relation is captured in terms of *visual phrases* [81, 82] which consist of more than one visual words and help distinguish the object from non-class instances which do not contain those specific visual words.

Recently effective techniques have been shown to improve image retrieval by aggregating element wise differences of local descriptors [25] and aggregating Tensor products of those differences [83]. This results in a larger feature vector for a small visual codebook capturing important statistics about the distribution of the descriptors inside the clustering cell. The more general Fisher Vectors have been shown to capture vital higher level descriptors

distribution statistics for visual recognition tasks [11, 12]. First the global density of the distribution of image descriptors is estimated using Gaussian Mixture Models (GMMs) trained on all the descriptors from the training images. An image signature is then calculated in terms of Fisher Vectors which indicate the direction in which the parameters of the image model to be changed in order to best fit the global distribution. The GMM can be regarded as a generative visual dictionary and each Gaussian is considered a visual word. Supervectors [84] work with a similarly trained GMM on the image features and encodes the first order difference with the mean of the cluster with the image representation.

2.3 Classification

The problems of video concept detection and video/image retrieval generally fall under the paradigm of *supervised learning* where a number of annotated image instances are available to build the model for each semantic concept. The learned model is trained on the ground-truth images to *classify* new instances as belonging to the class or not. Supervised classification methods are divided into two broad categories; Generative Classifiers, which learn the joint probability distribution of the training examples and the outputs for each class and apply Bayes' rule to predict the class probability of the new instance, and Discriminative Classifiers, which directly learns the classification boundary using examples from both class and non-class instances. We briefly review some of the popular methods for both the categories in the sections that follow.

2.3.1 Generative Models

Generally once the joint probability is modeled *maximum a posteriori* (MAP) or *maximum likelihood* (ML) estimates are used to find the candidate class(es) for the new image. Some pioneering works in image classification directly employed Bayesian classification to classify images into indoor and outdoor scenes [85, 86]. [85] further classifies outdoor images as city and landscape and continues to classify landscape images into sunset, forest and mountain. Mori et al. [87] use a co-occurrence model to find the correspondence between image features and the text keywords. A more robust idea is to find the correspondence between test image and the labeled images using e.g. Cross Media Relevance Modeling (CMRM) [88]. The keywords associated with the labeled images are used to annotate the test image. Huang et al. [89] combine CMRM with other correspondence models to improve image annotation. Another direction is to include latent variables in the model to associate visual features with keywords, referred to as latent Semantic Analysis (LSA) or probabilistic LSA (pLSA) [90, 91].

GMMs are also used to model the distribution of visual features [92, 93]. Blei and Jordan [94] assumed a Dirichlet distribution is used to create the mixtures and used Latent Dirichlet Allocation (LDA) to generate the distribution of visual features and keywords. Fei Fei and Perona [52] use a modified LDA to model "'themes"' from image regions for natural scene

categorization. The model learns the distribution of intermediate level themes and codewords and classifies new instances using Bayes' rule. Zhang et al. [95] further extend LDA in to an image decomposition model that identifies the important visual words from the image. The visual words are sampled as appearing from background distribution, image specific distribution and topic specific distribution if they appear in images with similar semantics.

2.3.2 Discriminative Models

Discriminative learning does not model the data distribution rather learns directly the classification rules or separating hyperplane using the input data.

Support Vector Machines (SVM) classifier tries to find a decision boundary that maximizes the margin or separation between the negative and the positive training class instances. SVMs can be used in their linear form or with a non-linear kernel which finds the decision boundary in the high dimensional space induced by the kernel. *Kernel trick* maps the input features into higher dimensions and allows the SVM to find a classification boundary within that space without actually explicitly doing any manipulations in that high dimensional space. SVMs have been widely used for classifying visual data in the binary classification framework and most of the times outperform their counterparts for recognition tasks [96] due to their generalization abilities. Some notable kernels used to find the distances between images are the radial basis function (RBF) and the chi square kernel [97, 98], Histogram Intersection kernel [99], Pyramid Match kernel [78] and the Spatial Pyramid kernel [54].

Other discriminative methods like **Nearest Neighbor** (NN) classification [100, 101] and **Neural Networks** [102] have also been used to learn concept classifiers. Moreover Zhang et al. [103] combine SVM and K-NN by training an SVM classifier only on the image nearest neighbors using a kernelized version of distance obtained by the NN classifier. [104] present a kernelized version of the non-parametric Naive Bayes Nearest Neighbor (NBNN) classifier that finds distance between images and class features. Two images are deemed similar if they have similar distance to the class models.

Due to the easy availability of unlabeled video data and the difficulty in annotating video keyframes **Semi-supervised learning** methods have also been explored [105–109]. New examples are labeled using the existing classifiers and the most useful ones are added to the training set for a new iteration of learning.

2.4 Fusion

Fusion of visual features or multi-modal features is gaining importance these days in the field of video content analysis and retrieval as it benefits from diverse and complementary information from different media [110–112]. This comes with a certain cost due to the higher complexity of multi-modal analysis as the modalities involved have different characteristics.

There are various methods and different levels at which information from different media can be combined [110, 111, 113, 114]. The important questions raised when combining multi-modal information are when and how to fuse. Naturally each semantic concept or group of similar concepts exhibits different dynamics from others so their fusion pattern could be different also. This argument is strengthened by the fact that one classifier and one set of features does not perform the same for each concept [114] as some concepts are easy to detect with visual features only while others may not be. We briefly review the state of the art multimedia fusion methods dividing them into early and late level of fusion.

2.4.1 Feature Level Fusion

To perform fusion in feature space also referred to as early fusion, unimodal features extracted from different data streams are integrated into the single large vector $\mathbf{v}$ for training. A certain pre-processing is performed like e.g. normalization so that features be on the same scale. Classifiers are trained for a semantic concept using these large multi-modal feature vectors and usually there is only one learning phase handling all multi-modal features at once.

Different visual features are sometimes combined to increase the power of individual features [115]. However combining textual features with the stream of visual data [113, 116], as well as incorporating audio based features [117–119] has also proven to be helpful for certain video analysis tasks.

2.4.2 Decision Level Fusion

Fusion can equally be performed at later stage integrating decisions of individual classifiers thus called decision level fusion, also semantic level fusion [113]. Classifiers are learned for features of different media separately giving a decision like yes/no or in the form of a score or probability of presence of a semantic concept. The independent decisions can be combined using different rules or classifiers can be built to learn from the output scores. Fusion in decision space is easier to perform as the decisions from classifiers usually have similar format. Moreover scores from new sources of information can be easily added to the final decision with only re-training the fusion part.

Late fusion has been extensively used in the state of the art due its simplicity and scalability. Weighted linear combination is an easy and effective way to combine decision from different classifiers [97, 98, 114, 120–122]. The weights are usually learned using some part of the training data as a validation set. Another scheme is to learn a classification model from the decisions based on the training data like SVM [113, 123] or the EM [124].

2.5 Evaluation

There are many benchmark video datasets available to judge the performance of video concept detection but we limit our evaluations to the TRECVID benchmark [18]. TRECVID organizes a workshop each year and addresses important video analysis tasks including Copy Detection, Event Detection, Event Recounting, Instance Search and Semantic Indexing (concept detection). Many research teams including universities, technical laboratories and certain enterprises participate in the tasks. With the difficulty and variety of their evolving datasets, their widespread acceptance, their evaluation procedures and the opportunity they provide to share the research and results among the participants TRECVID benchmark, for many years now, is considered the *de facto* standard to evaluate performance of various video analysis tasks [16]. Therefore we have opted to evaluate our propositions only on the TRECVID datasets. Specifically we evaluated our proposed methods on the Semantic INdexing (SIN) task where the data came from various years including 2007, 2010 and 2013. The videos are pre-segmented into shots and a keyframe is provided for each shot along with the ground truth annotations for the training set and the old test set. The list of semantic concepts to be indexed is also pre-selected and defined by NIST.

We have used Average Precision (AP) to evaluate the performance of our video concept detection models, which is also the standard used in the TRECVID evaluations for Semantic Indexing [18]. The overall system performance is evaluated by calculating the Mean Average Precision (MAP) by averaging the concept APs. To understand the working and calculation of AP let us first look at the simple measure of *Precision* and *Recall*. We would like to know what is the performance of our concept model on a list of test frames. Let us say we know which frames in the test set are *relevant* and which are *non-relevant*, i.e. which contain the semantic concept and which do not, respectively. Then precision can be simply calculated as the ratio of the relevant documents that are predicted to be true to the total predicted documents:

$$precision = \frac{|\{relevantdocuments\} \cap \{retrieveddocuments\}|}{|\{retrieveddocuments\}|}$$

As stated in the introduction a concept detection model predicts a new test frame and outputs a probability of presence of the concept in the frame. So the retrieved documents in the equation above mean some of the most confident predictions. More specifically the performance is only judged on the top R predictions with are the best predictions acquired from the concept model.The recall measure on the other hand finds out what percentage of the relevant frames have been retrieved by the model.

$$recall = \frac{|\{relevantdocuments\} \cap \{retrieveddocuments\}|}{|\{relevantdocuments\}|}$$

where the number of all the relevant documents is known pre-hand.

The Average Precision, which is actually the area under the precision-recall curve, gives importance to the order of the retrieved result in the ranked prediction of the list containing test shots.

$$AveragePrecision = \frac{\sum_{r=1}^{R} precision(r) \times relevance(r)}{|\{relevantdocuments\}|}$$

where $precision(r)$ is just the precision at the r'th result in the ranked list and the $relevance(r)$ gives a 1 if the keyframe is labeled positive with the concept. So the precision is calculated right from the start with the most importance given to the first (top) prediction and so on.

Chapter 3

Enhancing Video Representation

Learning to index general purpose videos automatically is a very hard problem. First of all the videos are mostly homemade or self-made uploaded by users on video sharing websites such as YouTube or Dailymotion and thus lack a proper defined structure, are poorly edited, suffer from camera motion and are sometimes of poor quality. Most of the times these videos are labeled or tagged by the uploader who is also sometimes the maker of the video. More relevant tags are sometimes added by someone watching the video. Annotation of visual content as it is now suffers from problems of bias, lack of knowledge and user specific interests.

Coming to the content of a video itself and leaving aside the lighting, motion and quality concerns the definition of a semantic concept is itself very elaborate and also complicated increasing the difficulty to build a general model for the concept. Ranging from a mere object to a very complex scene comprising multiple objects a semantic concept can take any shape or form in the video frame or can contain multiple frames. Moreover the definition of a concept or a category could be vague due to the visual variations with which it can be represented in the video. It is true as stated in [10] that a certain level of abstraction is needed to represent a concept. Moreover there is always noise when visual information is extracted from the video frames. Further information loss is incurred when going from low level visual features to mid level count based (histogram) representation, referred to as the semantic gap.

In this chapter we strive to improve standard video representation models which consists of the second stage of the video recognition pipeline presented in the introduction. Specifically we improve the Visual Dictionary or the Bag of Words model which encodes keypoints to specific clusters and pools together all the keypoint descriptors assigned to the same clustering cell. In order to achieve this improvement we present two methods in this chapter, one functioning on the construction of the dictionary [125] and the other adding refinement to the signature [126]. We first present the Bag of Words model and discuss the short comings it faces for the visual indexing task before detailing the two proposals.

3.1 Visual Dictionary Construction/BOW Model

A variety of visual information is extracted from single images, a glimpse of which is presented in chapter 2 along with the state of the art. These visual features vary in size and contain significant information about the various aspects of the concepts depicted in the images. However as usually they are high dimensional, high in count for a single image and are too specific for a certain image type they are aggregated to form a rather moderate *mid-level* representation before further processing and analysis. This caters for the abstraction that we talked about earlier, unifies the visual description and usually reduces the size of the representation for one image. Moreover the amount of visual features extracted can vary from image to image but the mid-level representation is of fixed length for each image. However the precision is lost as the pooling usually averages or maximizes over a certain criterion and the concepts now have to rely on the overall effect of the features in the images instead of some truly discriminative individual features.

We consider the Bag Of Words (BOW) model which is a popular method to represent video frames, figure 3.1, based on quantization that is inspired from text retrieval [8, 20]. It is a histogram based representation of scene description obtained through vector quantizing thousands of visual descriptors into a discrete visual dictionary. The visual descriptors are usually patch based features describing key interest points in video frames and are referred to as Local Image Descriptors (LIDs). The interest points can be detected by maximizing some function to identify areas in the image containing rich visual information or can be densely sampled from the image at fixed points. LIDs are then computed for each of the interest points by analyzing their local neighborhoods and are usually high dimensional vectors of fixed length.

The descriptor vectors from the training set are quantized using some unsupervised clustering process, like k-means, to divide the visual space into adjacent Voronoi cells. The centroids of the Voronoi cells in the visual space represents Visual Words. Each visual word corresponds to a bin of the final histogram that counts the number of features assigned to that cell (bin) for an image. The number of clusters and thus size of the histogram is fixed at the time of clustering. Each image is then represented by a fixed dimensional histogram, irrespective of the number of features extracted from the image as shown in the bottom right part of the figure 3.1. This histogram vector is computed for each image and can be directly fed into a discriminative classifier like Support Vector machines (SVM) to build a model for each category.

Image description in the BOW framework generally faces two important issues. First is the selection of the appropriate dictionary size and the second is the loss of information as features from all the images are considered equally for dictionary construction. This loss is in the form of contextual information and descriptor specific information. We detail these issues in the subsections below.

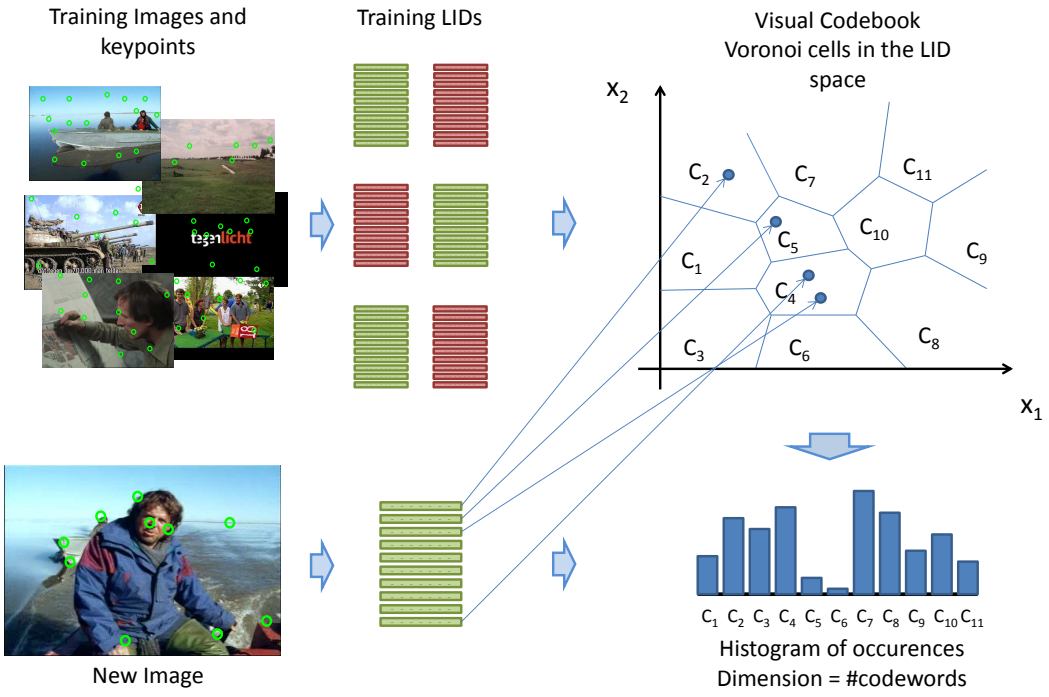


FIGURE 3.1: Visual Dictionary construction and histogram vector generation

3.1.1 The Dictionary Size

For video concept detection typical size of the BOW vector or visual dictionary varies from 200 to several thousands of words. This size is directly related to the categorization performance as well as retrieval efficiency. In figure 3.1 if we increase the number of features (LIDs) extracted from each image the final signature will remain the same. I.e. even if more and more visual content is extracted from each image the final representation will only show minor differences in the individual bin values as it is an approximation of the probability density of the LIDs in the descriptor space. Since we are dealing here with hard quantization we assume that this probability density is constant in each Voronoi cell. However it is obvious that a larger visual dictionary will approximate better this continuous density than a smaller one. This better approximation of the visual content thus increases the precision of the categorization.

However as the dictionary size increases there are more cells in the same clustering space and this affects the generalization ability of the model as noisy descriptors are miss-assigned. On the other hand if the dictionary size is kept small the discrimination is low as patches belonging to significantly different parts from the images are assigned to the same cell. There is thus a need to find a compromise between the dictionary size, its discrimination ability and its generalization capability. Although BOW is a very sparse representation, and with the increase in size the sparsity increases nevertheless the training and prediction efficiency are also affected with this increase in size. Besides, the time taken by k-means to construct

the dictionary is prohibitive for a large number of centers. Moreover more training data is required to train concept models for examples with larger BOW vectors as the classifiers will have more parameters. The matter of selecting the optimal visual dictionary size is thus crucial for the performance and efficiency of the concept detection system.

In this chapter, we present the work we did to increase the discriminative ability of a visual dictionary that is of small size. The first technique we present reduces the size of a large dictionary by merging Voronoi cells that result in minimum information loss while doing the approximation. Our second method adds refinement to a small dictionary to overcome the coarseness of the BOW model.

3.1.2 Loss of Contextual Information

Generating the visual vocabulary through unsupervised quantization from tens of thousands of low level descriptors does not capture semantic context as category information is not accounted for during clustering. Doing so, the expressive or discriminative power of the vocabulary is affected as only overall distortion is minimized and category information is not used increasing the semantic gap between the concept and the mid-level BOW feature. Nevertheless this does make the visual representation independent of the knowledge of the categories to be detected. We are however compelled to look into using the label information during the dictionary construction since the final goal is to detect individual concepts and label information could be useful. We can, for example, consider using contextual information to build histograms where bins k and l for all the images of *cats* are high and the same bins for all the images of *cars* are low. We want such discriminative information to be encoded automatically using examples' label information during the dictionary construction.

3.1.3 Loss of Descriptor Specific Information

The orderless BOW representation does not account for the importance of a descriptor to a histogram bin. Information about the location and structure of the LIDs and the position of the descriptor in the Voroni cell is usually lost. Even an outlier increases the count of a bin as it was the closest cluster center and it is given equal importance as the descriptor closest to the same cluster center. Smaller dictionaries are adversely affected by this as the Voronoi cells are larger and so the difference in the descriptors assigned to the same cell gets larger also, reducing the discrimination power.

3.1.4 Some Relevant Works

The problem of increasing the discriminative power of BOW model has been attacked by many authors in the recent years. We briefly overview some works that address the three problems detailed above.

Wang [127] builds a multi-resolution codebook by adding a new codeword at each step using hierarchical clustering and a selection criterion based on Boosting. This is done to find a compromise between a small codebook that lacks discriminative power and a large one that may result in overfitting. Perronnin et al. [128] represent each image with a bipartite histogram by building universal and class specific vocabularies using maximum likelihood estimation. Lin et al. [129] use a similar principle to bridge the semantic gap between the concept(s) depicted in the image and the low level features. k-means is used to generate separate class specific vocabularies followed by an agglomerative clustering on class codebooks to get the universal vocabulary. In both these works an image is represented by a set of histograms, one per class, using the amalgamated codebooks. Hao and Jie [130] present an improved BOW algorithm for scene recognition exploring discriminative power of codewords when representing different scene categories. They obtain a weighted histogram to code every image that highlights the discriminative capabilities of each codeword for each category.

To generate a discriminative codebook Winn et al. [53] present a two step clustering framework, where they compress an initial large dictionary by optimizing a statistical measure of discrimination that finds a compromise between low intra-class variance and inter-class discrimination. Moosmann et al. [55] build a set of randomized decision trees using the class labels with the leaf of a tree representing a spatial code (visual word). They calculate information gain of the split at each step of tree growing and use it as a threshold to split the tree based on the descriptor dimension at that level.

Su and Jurie [131] improve the performance of the BOW model by disambiguating different meanings of the BOW using semantic context. The semantic context is embedded in the BOW histograms to generate context specific representations. An image is then represented by combining different context-embedded BOW histograms by selecting the most appropriate context for each visual word. Wang and Mérialdo [132] weigh the *informativeness* of different visual words in the framework of kernel optimization. Given a concept, usually only some of the visual words frequently appear while the presence of the others are nearly random. In other words, some visual words are more informative or important for the detection of a specific concept, while the others may be noisy. In most current approaches, for different concepts, each visual word is treated equally. They find weights of each visual word for each concept by maximizing a Kernel Alignment Score giving larger weights to more important visual words. This improves the SVM performance for detecting that specific concept.

Philbin et al. [10] improve object retrieval in large scale databases by recovering information from descriptors which were lost in quantization. They use soft assignment to assign a descriptor to r closest cluster centroids, instead of a single centroid, weighted by the inverse of distance to the centroids. They try to overcome the problem of descriptor noise where a descriptor of a similar image patch may be assigned to a completely different visual word due to some distortions. Nevertheless all the descriptors assigned to the same visual words

are considered similar. Van Gemert et al. [76] also study visual word ambiguity through different weighted soft assignment methods.

In [133] authors have localized each descriptor inside a Voronoi cell based on a Hamming Embedding mechanism to increase the precision of the BOW histogram for the image search task. They perform image classification in a similarity space adapted from the Hamming Embedding based precision mechanism [65]. Jegou et al. [25] perform large scale image search using a signature obtained from calculating statistics on a small dictionary. The signature, called Vector of Locally Aggregated Descriptors (VLAD), though obtained from a small dictionary is itself large and affects the problem efficiency. Their work is a non-probabilistic approximation of the Fisher vectors [11, 12] represented by the gradient vector of the log likelihood depicting the direction in which the parameters of the generative model of the data should be modified to best fit the data.

3.2 Entropy based Supervised Merging

Our first proposal [125] tackles the first two problems presented in the section above by constructing a relatively small dictionary from a larger one using category information. Since dictionary construction achieved through unsupervised clustering does not take any advantage of the relationship between the features coming from images belonging to similar concept(s), this enlarges the semantic gap. We believe that the category information should be used in the dictionary construction process to encode some contextual information. Since the final classification tries to identify each category differently, encoding contextual information into the visual representation from an early stage may help improving the final concept model. This information can be used to build concept specific visual words or even concept specific dictionaries.

We aspire to build visual dictionary where the clustering cells are more consistent with the label information. Consider for example that within our training set we have a few images of the concept *Car* and our local interest points detector always detects wheels as some of the interest points. Now wheels can vary in visual appearance due to changes in illumination, occlusion, perspective distortion etc.. Moreover wheels of different cars could have different appearances. During the clustering for codebook construction not all the wheels are necessarily assigned to the same visual words. Label information can be used to merge clusters which largely constitute of wheel LIDs, figure3.2. This can be extended to other concepts to group together for example wheels of *Bus* or *Truck* as shown in the bottom of the figure 3.2.

However there may be cases where due to noise the visual word containing wheels also contains some instances of the visually close "windows" from the concept *Building*. Here we would like to partition the Voronoi cell into two cells maximizing the wheels and windows in

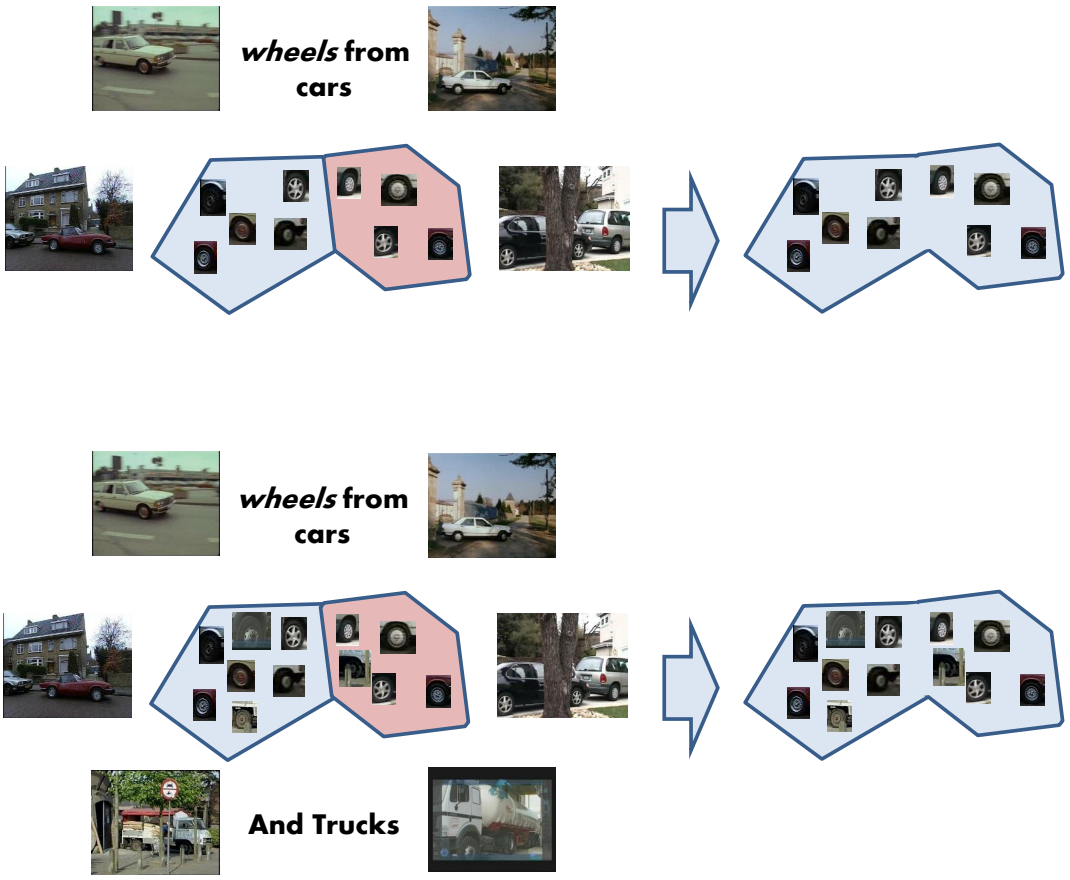


FIGURE 3.2: Related keypoints in neighboring cells: label information from images is used to merge the Voronoi cells

each of the new cells as shown in the figure 3.3. This again can be achieved using a criterion based on label information of the images.

Our proposal follows the first paradigm in figure 3.2 where we join two clusters accumulating all the assigned LIDs to the same histogram bin. We do the merging by designing a criterion that maximizes the information about the label distribution. We use a clustering method with only a few k-means' mean shift iterations using a better centers initialization based on K-means++ [134] to generate a larger than required number of clusters before doing a supervised merging. This significantly reduces the number of clusters (visual words) for faster training and retrieval. We initially merge neighboring clusters based on entropy minimization criteria that allows the generation of non-convex connex clusters. We present three such merging criteria in order to increase the discriminative power of BOW with an increase in the retrieval performance over the baseline. We then relax certain constraints in our merging criteria to allow the generation of non-connex clusters. We have used SIFT descriptors [63] calculated on keypoints extracted from images (keyframes), contrary to dense sampling [53, 55], labeled with one or more semantic concepts.

We also show that using our dictionary construction from supervised merging a smaller

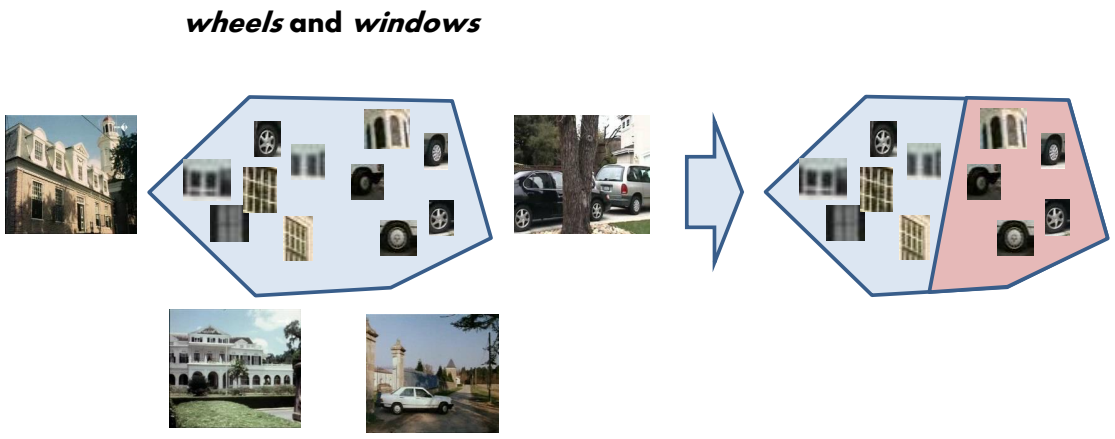


FIGURE 3.3: Unrelated keypoints in a clustering cell: label information from images is used to split the Voronoi cell

dictionary gives the performance comparable to the retrieval performance given by a dictionary upto 8 times its size.

3.2.1 Supervised Clustering Based on Entropy Minimization

In the two step clustering paradigm, Fig. 3.4, first of all the descriptors from the training images are quantized into a large number of visual words using k-means. The number of initial visual words (k-means clusters) is $p * D$, where D is the size of the desired dictionary. In the second step the number of clusters is reduced by $1/p$ by merging neighboring clusters repeatedly based on entropy driven information loss minimization criterion.

3.2.1.1 Concept Distribution Entropy Minimization

For deriving this minimization criterion we have to work at the individual descriptor level; i.e. we have to know the labels associated with each LID. We have m concepts (labels) $l_k \in \mathbf{L}, k = 1 \ldots m$ and we know with what concept(s) each image I is labeled with. Since video datasets are multi-label, each image can be associated with more than one concept. Thus in the end we know the labels of each image descriptor (keypoint). Now suppose as a result of the initial clustering we have $p*D$ clusters, and for each cluster $C_i \in \mathbf{C}, i = 1 \ldots p*D$ we know the labels of the keypoints assigned to it (we are only treating keypoints coming from labeled shots). For finding the number of keypoints belonging to a concept l in the cluster C_i we keep in mind that there may exist images that are labeled with more than one concept. Also generally keypoints extracted from a single image are assigned to different centers. Thus we compute the number of occurrences of concept l in the cluster C_i as:

$$|l \in C_i| = \sum_{I labeled with l} \frac{|LID(I) \in C_i|}{|LID(I)|} \tag{3.1}$$

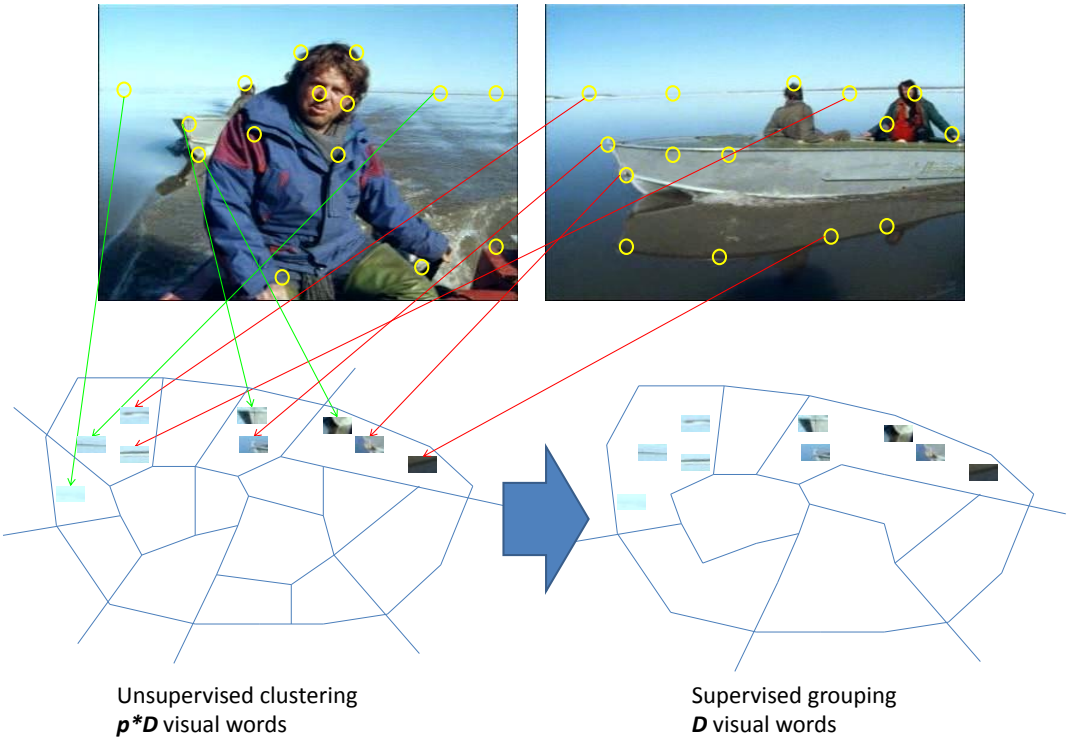


FIGURE 3.4: Supervised merging of visual words

We find next the conditional probability of the concept l given the cluster C_i:

$$p(l/C_i) = \frac{|l \in C_i|}{\sum_k |l_k \in C_i|} \tag{3.2}$$

The set $Nb\{i\}$ contains the neighbors of the cluster C_i where two clusters are neighbors if the midpoint between their centers is closer to those two centers than to any other center. When joining two neighboring clusters C_i and C_j, where $j \in Nb\{i\}$, all the keypoints in C_j are assigned to C_i, and all the neighbors of C_j are added to those of C_i. C_j is then deleted from the set of clusters i.e. $C = C \backslash C_j$.
The entropy of the concept distribution given the visual dictionary $V = \{C\}$:

$$H(L/V) = -\sum_C p(C) \sum_L p(L/C) \log p(L/C) \tag{3.3}$$

Since we are making further approximation by reducing the set of clusters to represent the probability density of the LIDs we lose information when we merge two clusters. This loss in information is represented by the entropy which is increased when any two clusters are merged. There are times when the increase in entropy is zero when the two merged clusters only contained LIDs labeled with similar concepts.

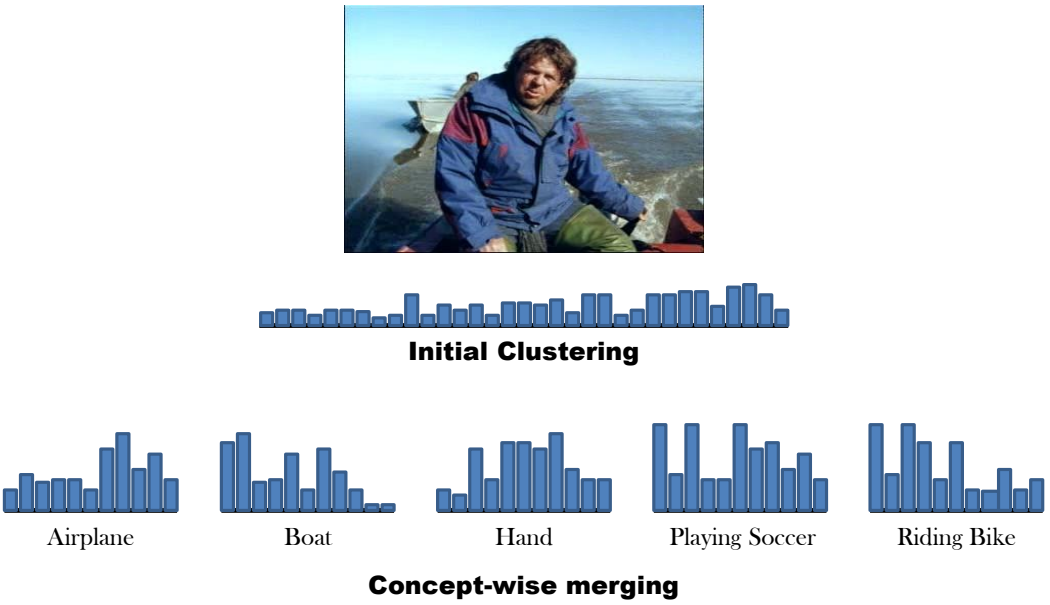


FIGURE 3.5: Related keypoints in neighboring cells: label information from images is used to merge the Voronoi cells

If we merge C_i and C_j, such that $j \in Nb\{i\}$, the resulting dictionary is V'_{ij} with one less visual word and entropy $H(L/V'_{ij})$.

$$V'_{ij} = V_{C_i \leftarrow C_j}$$

The combination $C_i \cup C_j$ that minimizes the increase in entropy is our target combination and those two clusters are merged together. We select i and j to minimize the new entropy.

$$\min_{i,j} H(L/V'_{ij})$$

This step is repeated $p * D - D$ times until the desired number of clusters D is reached.

3.2.1.2 Concept Dependent Entropy Minimization

The entropy minimization principle can be also used to find a merge of clusters independently for each concept using entropy of only that concept. This way we shall have one combination of clusters per concept and thus we will end up with a different BOW representation for each semantic concept, figure 3.5. Using the above notation, for the concept $l_k \in \mathbf{L}$ the entropy is given by:

$$H^{cd}(l_k/C) = -\sum_{C \in \mathbf{C}} \left[p(l_k, C) \log p(l_k/C) + p(\overline{l_k}, C) \log p(\overline{l_k}/C) \right] \tag{3.4}$$

where $p(\overline{l_k}, C) = p(C) - p(l_k, C)$ and $p(\overline{l_k}/C) = 1 - p(l_k/C)$.

Now for each possible combination of two neighboring clusters we will calculate the entropy to find the best merge by choosing the two clusters that result in minimum entropy increase, given by:

$$\min_{i,j} H^{cd}(l_k/V'_{ij}) \tag{3.5}$$

This step is repeated, reducing the total number of clusters by one each time, until the desired number of clusters is reached. This whole process is repeated for each concept resulting in a different clustering for each concept as well as a different bag of words model. An image is thus represented by a set of histograms, one per concept.

3.2.1.3 Average Concept Entropy Minimization

Another possibility to obtain a clustering combination is by combining the output of the concept dependent clusterings. This is done by taking the sum of entropy of all concepts for a merge of two clusters and then minimizing that sum for every possible combination of clusters. This clustering of average over all concepts is given by:

$$\min_{i,j} \sum_{l_k \in \mathbf{L}} H^{cd}(l_k/V'_{ij}) \tag{3.6}$$

where $H^{cd}(l_k/V'_{ij})$ is the concept dependent entropy as given in (3.4).

The difference between this criterion and the one presented in equation 3.3 where we minimized the entropy of the concept distribution is that here we look at each label individually and minimize the sum of concept dependent entropies. With concept dependent entropies each label has the same effect irrespective of the number of LIDs labeled with that label present in the clusters to be merged and we minimize this effect on average. With the concept distribution entropy minimization the label with more LIDs in the two target clusters has more effect on the entropy than the ones having fewer LIDs.

3.2.1.4 Relaxing Constraints

Based on the results shown in Sect. 3.2.2 we select the best entropy minimization based clustering criterion and make few changes. We have so far only used keypoints from labeled images to minimize the entropy for merging clusters. However the labeled images represent only a certain percentage of the whole corpus which is used to construct the initial dictionary. To reduce the bias of the labeled keframes over the unlableled ones we include all the keypoints in the second step of clustering. This is done by including all the unlabeled keypoints as the $(m+1)^{th}$ concept during the calculation of the entropy of the concept distribution and recalculating the mapping based on entropy minimization over $M+1$ concepts.

We further relax the constraint of merging only neighboring clusters where any two clusters (not necessarily neighbors) can be mapped together in the high dimensional disjoint clustering space. This allows the generation of a non-connex BOW model. These alterations are further explored in the Sect. 3.2.2 discussing the experiments and results.

3.2.2 Experiments

We present here experiments carried out on the TRECVID 2007 Sound and Vision database comprising 219 videos [18, 135] with around 36,000 video shots. A keyframe is extracted from each shot and is the shot representative. The training corpus consists of 110 videos and the other half is used for tests. Twenty semantic concepts are used to demonstrate the results. We have used 1 vs all SVM classifiers with chi-square kernel of degree 2 using the LIBSVM [136] package for each concept.

3.2.2.1 Supervised Clustering Results

Initially we evaluate the performance of supervised clustering for a final dictionary of 500 visual words using the three types of entropy minimization criteria. In our experiments the maximum value of p, as described in Sect. 3.2.1, is 8. That is the maximum size of initial visual dictionary obtained through k-means is 8 times the size of the desired supervised dictionary. To obtain a large initial dictionary we have used **k-means++** algorithm [134] for a better initialization in order to avoid a large number of k-means iterations, which is costly for a large number of centers. K-means++ is an initialization method that selects initial seeds far from each other while minimizing the effects of outliers. This is done by choosing the new cluster centers with a probability proportional to its distance to the closest center already chosen. After the initialization only 10 normal k-means iterations are performed to generate initial visual dictionaries.

Three Entropy Minimization Criteria

Using these large dictionaries supervised mappings are done from clustering space with 1000, 2000 and 4000 centers to 500 centers by merging neighboring clusters using entropy minimization criteria. In all three cases the final dictionary generated is always 500-words big which is then used to represent images as histograms. SVM classifiers are trained for each concept and independently for each set of histograms obtained through entropy minimization based mappings. The Mean Average Precision (MAP) for all 20 concepts is shown in the Table 3.1 for the 3 mapping criteria and for the 3 initial cluster sizes, along with the MAP obtained using 500 visual words achieved directly through k-means (baseline).

From table 3.1 we can see that with the increase in the number of initial centers the individual concept dependent entropy minimization criteria for mapping suffers from overfitting as it generates dictionaries for each concept independently. The image level labeling does not translate well to increase the discriminative power of the BOW model built for

TABLE 3.1: Mean Average Precision for 20 concepts for baseline (K-means) and three entropy minimization based mapping criteria. **Min Ent**: Concept distribution Entropy Minimization, **Cd**: Concept Dependent Entropy Minimization and **Cd (Av)**: Average Concept Dependent Entropy Minimization.

Dictionary Size	***K-means***	**Min Ent**	**Cd**	**Cd (Av)**
500	0.0739			
1000 to 500		0.0795	0.0757	0.0792
2000 to 500		0.0801	0.0758	0.0791
4000 to 500		0.0813	0.0727	0.0775

each concept separately. As the initial dictionary is made up using keypoints from all the images in the training set, generally the number of labeled is images is lower compared to the unlabeled images in the dataset. When merging visual words for each concept the number of labeled keypoints is lower (very low for some concepts) and this overfits the merged BOW representation.

This effect is carried on to the average (of concept dependent) entropy minimization as the retrieval performance is adversely affected with the increase in the number of initial cluster centers in the first step of clustering. Here however some loss is recovered due to averaging the concept dependent minimization, as evident from the last column of the table 3.1. Contrarily, merging neighboring clusters using minimization of the entropy of concept distribution given clustering improves retrieval performance with the increase in the size of the initial number of centers (Min Ent in table 3.1). This improvement is 10% overall when merging from an initial dictionary of 4000 visual words.

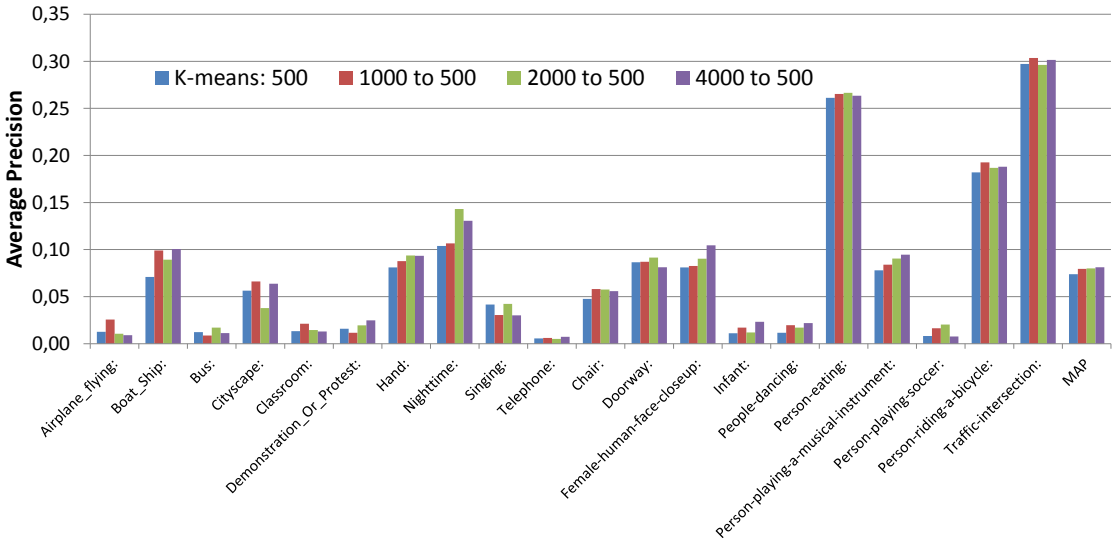
Concept-wise Performance for the Best Criterion

From table 3.1 we can select the best merging criterion which is the first criterion based on concept distribution entropy minimization. We use only this method to generate visual dictionaries of 1000 words obtained from larger dictionaries and check the performance on the similar dataset.

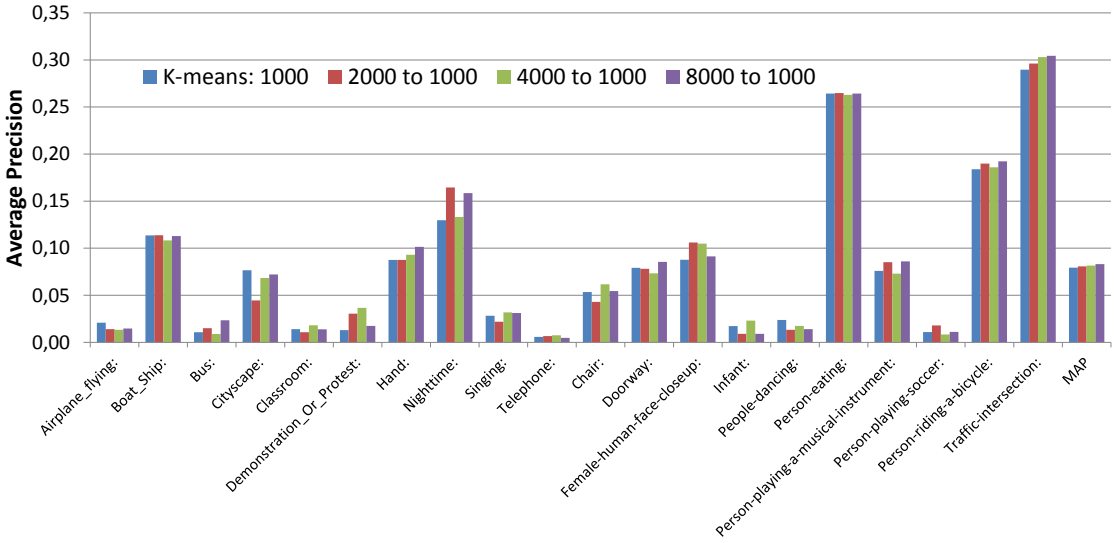
As it turns out the 1000 words merged dictionary outperforms the baseline of the same size and the performance increases as the initial dictionary size increases. Figure 3.6 shows concept-wise Average Precision (AP) results along with the MAP for the two baselines of 500 and 1000 visual words and entropy minimization based mappings with the value of p selected from 2, 4 and 8. Concepts like *Airplane Flying* and *Person Playing Soccer* that have a very low number of positives in the training set are adversely affected in their performance as the number of initial centers is increased. The MAP for 1000-words dictionary increases from 0.0796 (baseline) to 0.0831 with 8000 initial centers.

3.2.2.2 Alternative Mappings

We test retrieval performance for three simple modifications in the merging criterion. To see the effect of including the unlabeled examples in the second step of our clustering framework



(A)



(B)

FIGURE 3.6: Supervised clustering scores for 20 concepts using the first (best) entropy minimization criterion for (a) 500 and (b) 1000 visual words dictionaries

we include all keypoints coming from the unlabeled keyframes as a new concept making the total number of concepts $m+1$. The rest of the method remains the same which is the best performing entropy minimization criterion from the previous sub-section.

The second alternative is to allow the merging of non contiguous cells eliminating the constraint that only neighboring clusters can be merged, and the third version includes the unlabeled features while merging clusters over the whole clustering space. The MAP scores for a final dictionary of 500 words are presented in Table 3.2. Here we see good performance for the merging of 1000 initial clusters into 500 for the three alternatives and then a decline in the performance as the number of initial clusters is increased. This may be due to some overtraining as the number of choices for merging clusters are increased with the

relaxation of constraints. The problem with including keypoints from unlabeled examples is that they overpower the entropy minimization criterion and some of the mergings ignore the information coming from keypoints labeled with true concepts. This increases with the number of initial centers and explains the decline in performance. However the performance is still better than the baseline results in each case.

TABLE 3.2: Mean Average Precision for 20 concepts using three alternatives of the concept distribution entropy minimization based mapping. **Unlab**: Inclusion of the unlabeled examples, **All Space**: Allow merging of non-contiguous cells and **Unlab All Space**: Non-contiguous merging allowed with unlabeled examples.

Dictionary Size	***K-means***	**Unlab**	**All Space**	**Unlab All Space**
500	0.0739			
1000 to 500		0.0803	0.0805	0.0794
2000 to 500		0.0788	0.0795	0.0795
4000 to 500		0.0777	0.0755	0.0780

Maximum Concept-wise Improvements

Finally we give upper bounds of improvement for each concept with highest average precision score selected from the retrieval results of different initial dictionary sizes for concept distribution entropy minimization with (*i*) neighboring constraint, (*ii*) inclusion of unlabeled features, (*iii*) relaxing the neighbor constraint and (*iv*) the inclusion of unlabeled features with the relaxation of neighbor constraint. Tables 3.3-(a) and 3.3-(b) shows the upper bounds of improvements for the two step clustering performed with its alternatives showing significant increase in the individual score for each concept for 500 and 1000 word final dictionary sizes. Individual scores for each concept improve significantly with only the concepts *Person eating* and *Traffic intersection* showing little improvement as their performance is already quite high. An exception is the concept *Airplane flying* which shows a decrease in performance with supervised merging for 1000 words dictionaries as shown in Table 3.3-(b).

3.2.2.3 Small and Informative vs Large Dictionaries

In the previous sub-sections we have shown results of our technique for building a visual dictionary and compared them to retrieval results of the baseline using a dictionary obtained through sufficient number of k-means iterations. In those cases the sizes of the baseline and supervised dictionaries were same (500 words and 1000 words). We claimed that the retrieval performance of using dictionary obtained through supervised merging matches that of using a larger dictionary which is evident from the results in the Table 3.4.

SVM classifiers were trained for larger dictionaries containing 1000, 2000, 4000 and 8000 visual words. These are the dictionaries obtained in the first stage of the supervised merging using k-means. The training and retrieval time for these larger BOWs is much higher and the performance is comparable to the smaller supervised BOW models. From tables 3.4 (A) and (B) we can see that the MAP score for the visual codebook with 2000 words is

TABLE 3.3: Upperbounds of concept-wise improvements for dictionaries of (a) 500 visual words and (b) 1000 visual words

(A)

Semantic Concept	k-means	Entropy	Unlabeled	All-space	Unlab All	improvment
Airplane flying	0.0126	0.0257	**0.0266**	0.0266	0.0212	111%
Boat/Ship	0.0709	0.1005	0.1041	**0.1149**	0.1015	62%
Bus	0.0123	0.0170	0.0140	**0.0248**	0.0168	103%
Cityscape	0.0564	**0.0661**	0.0522	0.0652	0.0660	17%
Classroom	0.0132	0.0211	0.0265	**0.0463**	0.0197	250%
Demonstration	0.0158	0.0249	0.0239	**0.0325**	0.0168	106%
Hand	0.0809	0.0938	**0.0962**	0.0879	0.0872	19%
Nighttime	0.1037	0.1430	0.1487	**0.1545**	0.1460	49%
Singing	0.0415	0.0423	0.0399	0.0367	**0.0435**	5%
Telephone	0.0055	0.0072	0.0072	**0.0074**	0.0071	34%
Chair	0.0476	0.0581	**0.0618**	0.0576	0.0553	30%
Doorway	0.0865	**0.0915**	0.0857	0.0840	0.0810	6%
Female face closeup	0.0809	**0.1045**	0.1040	0.0959	0.0984	29%
Infant	0.0110	**0.0233**	0.0131	0.0170	0.0158	111%
People dancing	0.0116	0.0219	0.0218	**0.0301**	0.0196	159%
Person eating	0.2612	0.2664	0.2653	**0.2714**	0.2683	4%
Playing music	0.0779	0.0945	0.0926	**0.0949**	0.0884	22%
Person playing soccer	0.0081	**0.0203**	0.0179	0.0137	0.0138	151%
Person riding bicycle	0.1820	0.1926	0.1980	0.1949	**0.2100**	15%
Traffic intersection	0.2971	0.3036	0.3065	0.2947	**0.3096**	4%
MAP	**0.0739**	**0.0859**	**0.0853**	**0.0875**	**0.0843**	

(B)

Semantic Concept	k-means	Entropy	Unlabeled	All-space	Unlab All	improvment
Airplane flying	**0.0208**	0.0147	0.0130	0.0197	0.0178	-5%
Boat/Ship	0.1136	0.1129	0.1016	**0.1167**	0.1080	3%
Bus	0.0107	0.0235	0.0194	0.0212	**0.0317**	198%
Cityscape	0.0765	0.0720	**0.1042**	0.0697	0.0591	36%
Classroom	0.0140	0.0181	0.0177	**0.0394**	0.0197	181%
Demonstration	0.0130	**0.0366**	0.0339	0.0335	0.0332	181%
Hand	0.0876	0.1014	0.0997	**0.1023**	0.0959	17%
Nighttime	0.1297	0.1585	**0.1912**	0.1712	0.1570	47%
Singing	0.0282	0.0317	0.0397	**0.0405**	0.0367	44%
Telephone	0.0058	**0.0074**	0.0066	0.0059	0.0063	29%
Chair	0.0534	**0.0616**	0.0571	0.0581	0.0579	15%
Doorway	0.0792	**0.0855**	0.0825	0.0757	0.0777	8%
Female face closeup	0.0877	**0.1049**	0.1038	0.0957	0.1014	20%
Infant	0.0173	**0.0231**	0.0107	0.0151	0.0139	33%
People dancing	0.0238	0.0174	**0.0295**	0.0191	0.0181	24%
Person eating	0.2643	0.2643	0.2668	**0.2694**	0.2657	2%
Person playing music	0.0758	0.0860	**0.0890**	0.0830	0.0790	17%
Person playing soccer	0.0109	0.0111	**0.0129**	0.0120	0.0119	18%
Person riding bicycle	0.1839	0.1923	0.1875	0.1912	**0.2069**	13%
Traffic intersection	0.2967	**0.3045**	0.3014	0.2978	0.3024	3%
MAP	**0.0796**	**0.0864**	**0.0884**	**0.0869**	**0.0850**	

0.0814% and the MAP score for the 500 words merged dictionary from an initial 4000 words dictionary is 0.0813%. Similarily the performance for the codebook of 4000 words is 0.0830% and that of a 1000 words merged one with 8000 initial centers 0.0831%. So we can safely say that supervised clustering builds discriminative models that achieve performance similar to a dictionary 4 times their size.

If we further increase the size of the baseline dictionary (to 8 times the merged one) the difference in performance is not so great. For example the 8 times smaller dictionary

TABLE 3.4: Comparing MAP for 20 concepts using three large dictionaries vs corresponding smaller supervised dictionaries of (a) 500 and (b) 1000 visual words

(A)

1000		**2000**		**4000**	
k-means	**500**	**k-means**	**500**	**k-means**	**500**
0.0796	0.0795	0.0814	0.0801	0.0830	0.0813

(B)

2000		**4000**		**8000**	
k-means	**1000**	**k-means**	**1000**	**k-means**	**1000**
0.0814	0.0806	0.0830	0.0816	0.0847	0.0831

only results in 2% performance decrease as can be seen in the last two columns of the Table 3.4-(a) where we go from 4000 to 500 visual words and less than 2% performance decrease as evident in the last two columns of the Table 3.4-(b) where the size is reduce from 8000 To 1000.

The difference in performance will increase as the size of the first step dictionary increases as it becomes richer and richer. While merging helps to capture important semantic information the performance of the resulting supervised dictionary will be limited as the smaller dictionary will always be a coarser representation of the visual space. Although we are using the best merging criterion that minimizes the entropy of the whole concept distribution we still only calculate the mappings of the merging using only the training dataset and the resulting codebook still suffers from some overtraining.

As far as the computation overhead for supervised clustering is concerned it only uses the information from the image labels. Thus it does not perform any direct computation on the image features. The time complexity of supervised clustering is $O(n^3)$, with $n = p * D$, which is the cost borne once at the clustering stage during training phase giving a mapping from $p * D$ visual words to D visual words. After the two step clustering the costs of baseline and supervised dictionary for the remaining stages of video retrieval are similar.

3.3 Intra-BOW Statistics

Research in video retrieval systems is mainly inspired by the state of the art text retrieval where high dimensional descriptors are quantized to visual words making a Bag Of Words (BOW) histogram for an image. For a small BOW model potentially different descriptors could get assigned to the same visual word. Recently however refinements have been proposed to recover some of this *representation loss* for this simplistic model of visual description of images by studying the distribution of descriptors within the visual words [12, 25, 65, 133]. Following the same foot-steps we enhance the BOW by encoding [126] the position of each of the descriptor inside the quantized cell according to its centroid. Here a new small BOW is

built by quantizing the differences between descriptors and their cluster centers. Embedding this new feature with the original to represent images increases precision of video concept detection. We compare our method to a BOW based baseline on the TRECVID 2007 and the TRECVID 2010 [18, 135] datasets and show that adding the refinement proposed always improves the semantic indexing task. We also compare our method to that of [65] and show that it outperforms the Hamming Embedding Similarity based classification on the TRECVID 2007 dataset and illustrates comparable performance on the TRECVID 2010 set.

Our intention is to find a compromise between retrieval efficiency and discriminative power of the BOW model. We suggest to add important information to the coarse dictionary by exploring intra-BOW statistics rendering it more discriminative with a little increase in its size. We propose an enhancement of the traditional codebook by distinguishing the descriptors assigned to the same visual word, following the refinement proposed in [133]. We calculate the difference between descriptors and cluster centroids similar to [25] but instead of summing them up along each dimension we quantize them to build a secondary signature to be used along with the primary BOW feature. Furthermore our signature is much shorter in length than that of [25]. Our method is directly adaptable in the BOW framework as the added signature is easily computed directly from the quantization information as opposed to conversion into a similarity space [65].

We compute the proposed refinement for visual dictionaries of varying sizes and prove that adding the new refinement based signature to the BOW model always significantly improves performance of the video indexing task for the TRECVID 2007 and TRECVID 2010 datasets. The results also indicate that adding the very small refinement signature to a smaller dictionary outperforms the performance of a much larger dictionary thus reducing the training overload, required for learning models for large dictionaries, and the retrieval time. We also show that the indexing performance is superior to or equal to the technique proposed in [65].

3.3.1 Difference BOW

In the BOW framework all the image descriptors assigned to the same visual word are considered identical, irrespective of the difference between them in the high dimensional Voronoi cell. To make the visual dictionary more discriminative its size should be increased, however this affects the application efficiency along with the generalization capability of the dictionary. Furthermore the construction time of the dictionary increases with its size. On the other hand a small dictionary generalizes well to noisy descriptors but does not give good classification results. Using a coarse dictionary for classification precision is lost as potentially different image patches are assigned to the same visual word due to their somewhat similar appearance in an image. Consider as an example a visual word depicting *tire* in the visual space. Now this word may contain tires from cars, motor bikes or even bicycles. We intend

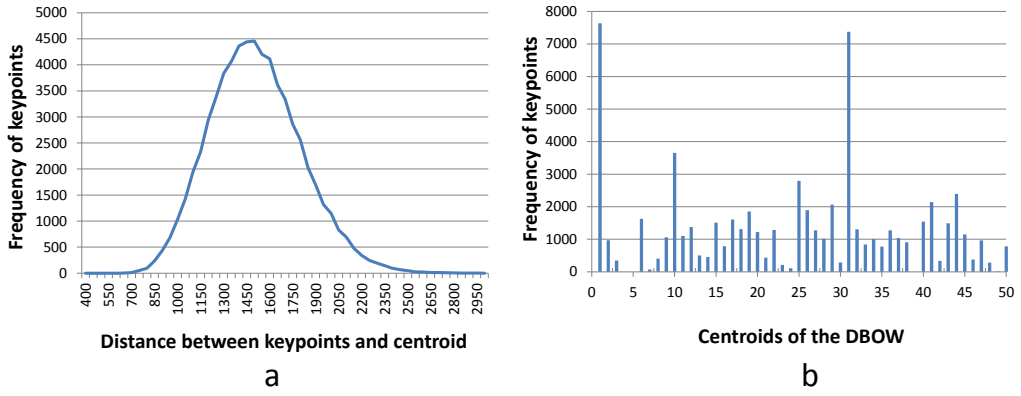


FIGURE 3.7: Distribution of Difference vectors in a Voronoi cell: (a) magnitude of Difference vectors, (b) quantized over a 50-centroid global difference dictionary

to further subdivide the space inside a Voronoi cell so that the *tires* that are visually very close to each other group together.

We thus try to calculate the position of each descriptor inside the clustered cell respective to its centroid and encode it along with the BOW representation. This is done to refine the BOW representation by differentiating each descriptor inside the same Voronoi cell from the others. Differentiating descriptors inside the clustering cell increases precision of the indexing task. Doing this we also find a trade-off between training / retrieval efficiency and discriminative power of the visual dictionary as the information we encode is small in size compared to the BOW model but is meaningful.

To acquire this descriptor specific information the idea is to partition the high dimensional clustering space to measure the similarity between the local descriptors assigned to the same visual word. In Hamming Embedding [133] authors localize each descriptor inside the clustering cell and then generate a low dimensional binary signature capturing this localization. Following the same inspiration we employ a simple *global* mechanism to localize a descriptor inside a Voronoi cell focusing on the notion that two *distinct* descriptors inside the same cell should have a sub-signature different from each other.

Each Voronoi cell in the high dimensional clustering space has descriptors located all over. We show distributions of Euclidean distances between the center and the assigned descriptors in the figure 3.7-(a) for a cluster (a Voronoi cell) with maximum number of descriptors, from a 500 words dictionary on TRECVID 2007 training data. The distance is also the magnitue of the difference vector between the descriptor and the cluster centroid. Although the distance is normally distributed the range of values for distance is very large and could be a good candidate to differentiate descriptors within the cell. But doing so does not include any location which can be recovered by considering difference vector as a whole instead of its magnitude. Figure 3.7-(b) shows the histogram of the difference vectors quantized over 50 bins for the same Voronoi cell. We can conclude that difference vector is a good candidate to further classify each descriptor inside a cell into a small number of classes.

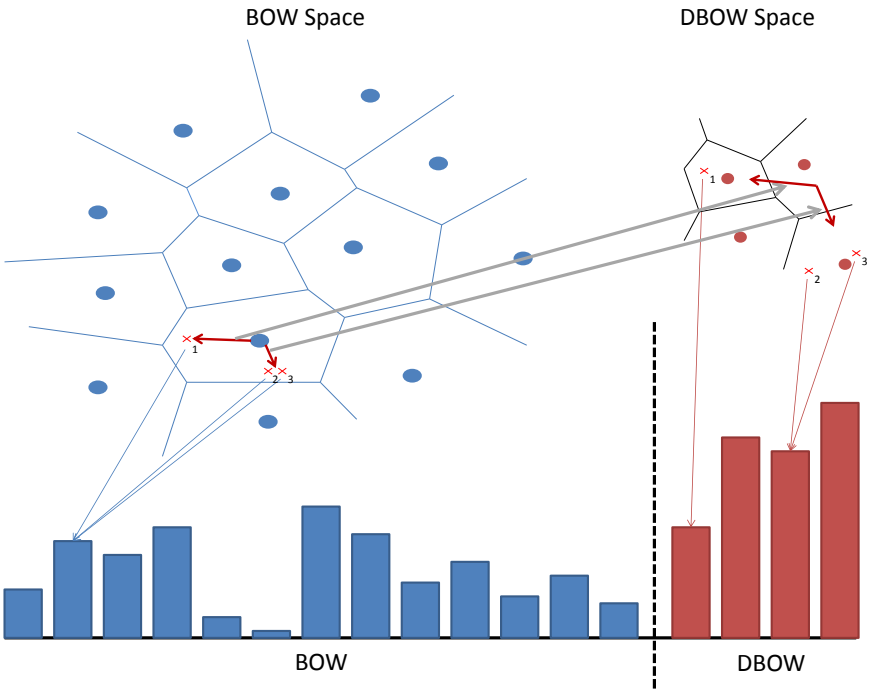


FIGURE 3.8: Representation of BOW and DBOW

We build a k-words visual dictionary $C = \{c_1 \dots c_k\}$ and to find the location of each descriptor x inside the Voronoi cell we follow [25] by calculating its difference from the nearest cluster center c_i as: $\overline{x} = x - c_i$. These difference vectors are stored and then quantized to generate a new dictionary $D = \{d_1 \dots d_l\}$ of size smaller than the original visual dictionary i.e. $(l << k)$. This leads to the formation of the difference BOW (DBOW) feature that are words added as an extension to the original dictionary to increase descriptor precision. The refinement proposed is shown in the figure 3.8 where keypoints belonging to the same visual word are assigned to different *difference words* based on their position inside the Voronoi cell. DBOW is a global model which is calculated quantizing the difference vectors from all the clusters of the BOW dictionary. Figure 3.7-(b) shows the distribution of difference vectors from the most populous BOW visual word assigned using this global DBOW model. Both the BOW and DBOW are used together to represent images.

3.3.1.1 Weighted DBOW

All the words of the DBOW, as explained previously, are given equal importance when representing images. Since each cluster has different number of descriptors but since the DBOW clustering is global each bin should have a separate weighting for difference vectors. Also DBOW vectors should be given image specific weights as for an example image certain bins might dominate the others in BOW.

We use frequency of keypoints in BOW to weight DBOW bins. To achieve this each difference vector belonging to a cluster is assigned a weight equal to the number of descriptors

belonging to that cluster, or the size of that BOW bin: $w_{\overline{x}} = |x \in c_i|$. The weight of a DBOW word w_{d_j} is then calculated by adding the weights of all the difference vectors that are quantized to d_j given by,

$$w_{d_j} = \sum_{\overline{x} \in d_j} w_{\overline{x}}$$

and normalized by the sum of weights for all DBOW bins. These weights are calculated for each image separately. We actually use square root of these weights in the experiments due to the high effect of the linear weights.

3.3.2 Experiments

3.3.2.1 Experimental Setup

Datasets: We have again used the TRECVID 2007 Sound and Vision database comprises 219 long videos [18, 135] totaling 200 hours of the sort news reports, documentaries, educational programmes, and archival video. The training corpus consists of 110 videos (21,532 video frames or *keyframes*) and the other half (22,084 keyframes) is used for tests. Twenty semantic concepts from the 2009 high-level feature extraction task[135] are used to demonstrate the results on the indexing.

The TRECVID 2010 IACC [18] dataset contains 11644 internet videos. Approximately 3200 videos of 200 hours make up the development set with a total of 119,685 keyframes. Rest of the approximately 8000 videos of 200 hours containing 146,788 keyframes are used for test. The list of concepts is the same from the TRECVID light semantic indexing (SIN) task from 2011 and 2012 comprising 50 concepts.

Local Features: We use Difference of Gaussian detector [63] to detect local interest patches or *keypoints* which are described by 128 dimensional SIFT [63] feature. A maximum of 500 features are extracted from a single video keyframe. These SIFT features are then clustered to construct visual dictionaries of varying sizes. The same set of SIFT features are used for finding the descriptor projections in calculating the Hamming Embedding (HE) based similarity [65] with which we compare our method.

As clustering such a huge number of descriptors is a tedious task we have again employed **k-means++** algorithm [134] for a better initialization. K-means++ initialization is followed by 10 normal k-means iterations to generate visual dictionaries. Visual dictionaries with 500, 1000 and 2000 visual words are computed separately for both the datasets. The same dictionaries are used for all the experiments, namely: the **BOW baseline**, the **Hamming Embedding Similarity** based feature and the **BOW-DBOW** feature.

The baseline comprise of classifiers trained on the simple BOW models. DBOWs of 10, 50 and 100 difference words are computed for each of the three visual dictionaries. Thus a total of 9 DBOW models are created for each dataset again using the rapid k-means method

(k-means++ and 10 iterations of k-means). For calculating the HE Similarity Scores signature again the baseline dictionaries are used as described later.

Classifiers: For all the experiments on the TRECVID 2007 dataset we have used non-linear SVM classifiers from LIBSVM [136]. In each setting, for each concept a 1 vs all SVM classifiers is used with chi-square distance using RBF kernel of degree 2. The SVM and kernel parameters are optimized by doing gridsearch on the development set. Contrarily for the TRECVID 2010 dataset we have used linear SVM to learn from a suitable feature map (homogeneous kernel map) built by the histogram intersection kernel [137, 138]. Here we only have to optimize the SVM parameter.

3.3.2.2 Hamming Embedding Similarity Feature

For comparison to the state of the art we have implemented the Hamming Embedding (HE) Similarity based image classification framework for the TRECVID datasets based on [65]. Jain et al. have embedded the HE based encoding into a similarity space for image classification [65].We have followed the similar steps described in [65].

In Hamming Embedding a short binary signature is added to each descriptor refining its location inside a Voronoi cell. A Voronoi cell is divided into 2^m *portions* by m hyperplanes. Each hyperplane is described by a binary hamming bit. Two descriptors lie on the same side of the hyperplane if their corresponding hamming bit reflects the same value. Two descriptors match if they lie in the same cell of the clustering space and if the distance between their Hamming signatures is upto a threshold h_t. Inferring from [65] we have used $m = 64$ and a fixed threshold of $h_t = 22$ for all experiments. System details can be found in [65] and [133]. It is important to mention here is that we have used SIFT descriptors from the development sets of TRECVID 2007 and TRECVID 2010 datasets separately to generate the Hamming Embedding hyperplanes (the median values) for each cluster center. This is also done separately for each of the three dictionary sizes from 500 to 2000 as they embed different clustering spaces.

Hamming distance between signatures cannot be directly employed for image classification as it does not produce a global feature for each image like the BOW model, but instead measures distance between two images. Thus an image is represented in similarity space by calculating and aggregating Hamming distance between its descriptors and a set of reference images, as [65]:

$$I_{HE} = [HE_{sim}(I, I_1)HE_{sim}(I, I_2)\dots HE_{sim}(I, I_N)]$$

where $HE_{sim}(I, I_i)$ is the similarity computed by HE between images I and I_i. N is the number of training images used for calculating the N dimensional HE similarity vector. The training images are chosen randomly form the development sets independently for the two datasets.

	Methods	1	2	3
Base Dictionary 500 Words	**Baseline**	0.0739	-	-
	HE Similarity	0.0781	0.0741	0.0770
	DBOW	0.0764	0.0796	**0.0830**
	DBOW weighted	0.0761	0.0810	0.0821
Base Dictionary 1000 Words	**Baseline**	0.0796	-	-
	HE Similarity	0.0815	0.0777	0.0820
	DBOW	0.0804	0.0831	**0.0850**
	DBOW weighted	0.0821	0.0845	0.0813
Base Dictionary 2000 Words	**Baseline**	0.0814	-	-
	HE Similarity	0.0737	0.0768	0.0801
	DBOW	0.0824	0.0835	0.0854
	DBOW weighted	0.0848	**0.0868**	0.0829

TABLE 3.5: Mean Average Precision of all methods for 20 concepts (TRECVID 2007) for 3 different dictionary sizes

To keep harmony we have used the number of random training images N to generate the HE similarity signature equal to the size of the underlying dictionary. So for calculating the HE similarity signature of size $N = 500$ we have used the BOW model with 500 visual words, for $N = 1000$ the underlying BOW model used is 1000 and so on for all experiments. We do the power normalization of the HE similarity scores signature as described in [65] for values of $\alpha \in [0, 1]$ and obtain best results for the values 0.3 and 0.5 doing cross validation on the development set for both datasets.

3.3.2.3 Results

Results for three versions of each method are shown in the tables 3.5 and 3.6 for TRECVID 2007 and 2010 respectively. For each size of base dictionary there are three versions of HE similarity score based classification. Version 1 is the performance of HE similarity score based feature without any normalization, version 2 represents results with normalized feature using $\alpha = 0.3$ while version 3 uses $\alpha = 0.5$. Similarly for DBOW based methods (DBOW and weighted DBOW) the versions 1, 2 and 3 show DBOW vector of size 10, 50 and 100 added to the (BOW of) base dictionary for all sizes. For baseline there is only version 1 which shows classification results on the base dictionary for the three sizes.

TRECVID 2007

Table 3.5 shows the Mean Average Precision (MAP) scores for 20 concepts of the TRECVID 2007 dataset for the three versions of the HE similarity and DBOW based features and compares them to the baseline. It proves the discriminative power of DBOW bins added to the BOW based feature for all dictionary sizes. Adding the DBOW bins to the BOW improves results significantly even with the addition of as little as 10 difference bins. Another important result produced is the performance of the small BOW-DBOW feature compared to that of a large BOW feature. As evident form table 3.5 Adding 100 difference words to the base dictionary of 500 words outperforms the performance given by a baseline dictionary almost twice and even four times its size.

We show the increase or decrease of the performance of the methods relative to the baseline methods. To this end for each of the methods in Table 3.5 we select the best score for each concept and show its difference from the score obtained by the baseline for that concept. Thus for HE similarity the best performance is chosen from the normalized and un-normalized versions of the method and for DBOW method the choice is made between the scores from varying sizes of DBOW. These relative scores are shown in figure 3.9. For

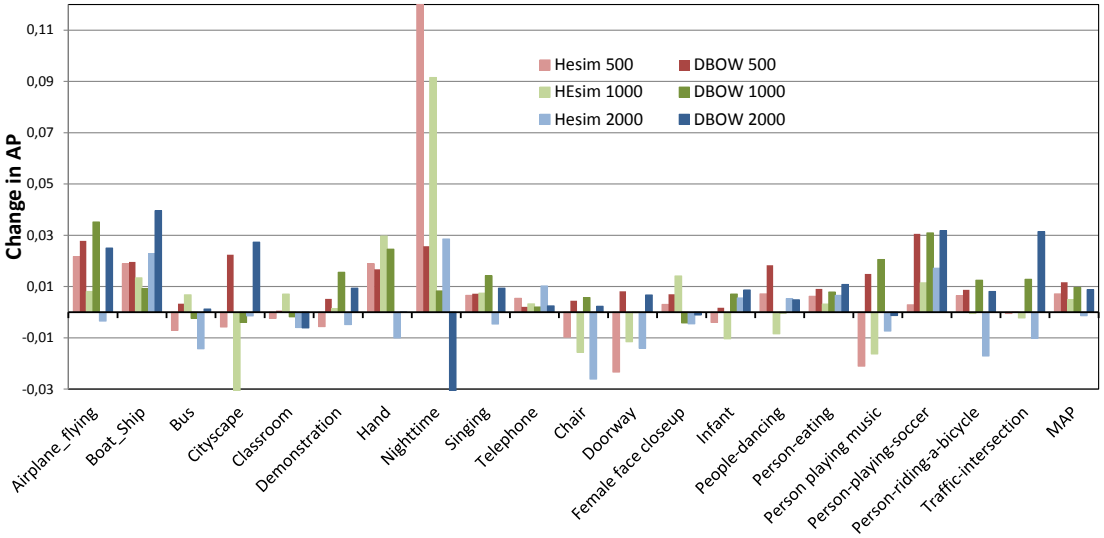


FIGURE 3.9: Concept scores for TRECID 2007: increase/decrease in Average Precision score over the baseline for the best performing HE similarity and DBOW methods for the 3 base dictionary sizes.

each color the light bar represents the best HE similarity relative AP score for a concept and the darker one represents the best DBOW based AP score relative to the baseline. Note that DBOW out performs HE similarity method for most of the 20 concepts for the three base dictionary sizes. Only the HE similarity performs exceptionally well for the concept *Nighttime*.

TRECVID 2010

Results on TRECVID 2010 dataset follow almost similar trends as those on TREVID 2007. All the experiments are carried out in the similar fashion as done for the 2007 dataset with the three base dictionaries of 500, 1000 and 2000 visual words. Here linear SVM with histogram intersection kernel is used to train classifiers for 50 concepts in each setting. The values of α are again selected as 0.3 and 0.5 for normalizing the HE similarity score and the SVM parameter C is chosen from the set of 3 values $\{0.01, 0.1, 1\}$ for each concept based on the validation done on the development set.

Table 3.6 shows results in similar fashion as table 3.5 and proves the effectiveness of adding the DBOW to the BOW representation. DBOW outperforms the baseline for all dictionary sizes and performs comparable to the HE similarity based method. Table 3.6 also

	Methods	1	2	3
Base Dictionary 500 Words	**Baseline**	0.0336	-	-
	HE Similarity	0.0353	0.0364	0.0356
	DBOW	0.0349	0.0359	**0.0365**
	DBOW weighted	0.0356	0.0352	0.0360
Base Dictionary 1000 Words	**Baseline**	0.0368	-	-
	HE Similarity	0.0406	0.0412	0.0420
	DBOW	0.0409	0.0413	0.0420
	DBOW weighted	0.0412	0.0413	**0.0421**
Base Dictionary 2000 Words	**Baseline**	0.0403	-	-
	HE Similarity	0.0447	0.0422	0.0445
	DBOW	0.0430	0.0419	0.0415
	DBOW weighted	0.0441	**0.0459**	0.0442

TABLE 3.6: Mean Average Precision of all methods for 50 concepts (TRECVID 2010) for 3 different dictionary sizes

shows that the DBOW performs comparable to the basic BOW double its size. The overall low scores are due to the linear SVM used on the complex dataset.

Figure 3.10 shows the relative increase or decrease in performance of the 50 concepts for the best performing HE similarity and DBOW methods compared to the BOW baselines for the three dictionary sizes. Again the DBOW outperforms HE similarity based method for most of the concepts except *Computer_or_Television_Screens*, *Dark-skinned_People*, *Doorway*, *Indoor*, *Military_Base*, *News_Studio*, *Road* and *Vehicle*. DBOW performs better on average for the three cases. HE similarity occasionally performs worse than the base line.

HE signature computes similarity to training images and is thus very dense. This adversely affects the SVM training time specially for the non-linear kernel used for TRECVID 2007 dataset. On the other hand DBOW signature is also dense due to its smaller size but the sparsity of BOW diminishes the dense effect and training the classifiers takes much less time than for HE similarity vectors.

3.4 Conclusions

We have seen that the discriminative ability of the Bag of Words model increases when performing the two step supervised clustering. The performance of a much smaller dictionary obtained through supervised merging reaches that of larger dictionaries obtained through k-means. The merging step is fast and incorporates already available label knowledge for calculation of the entropy. Although class specific merging of clusters overfits the BOW representation the performance is high as long as the initial number of clusters is kept low.

The introduction of DBOW bins encoding the position of descriptors inside the clustering subspace shows a lot of promise in rendering a compact dictionary more discriminative. We compared our method to a state of the art method for image classification [65] showing that it is consistently outperformed on two video datasets for the semantic indexing task. Here again we have seen a small dictionary with the added DBOW vector outperforming a much larger one.

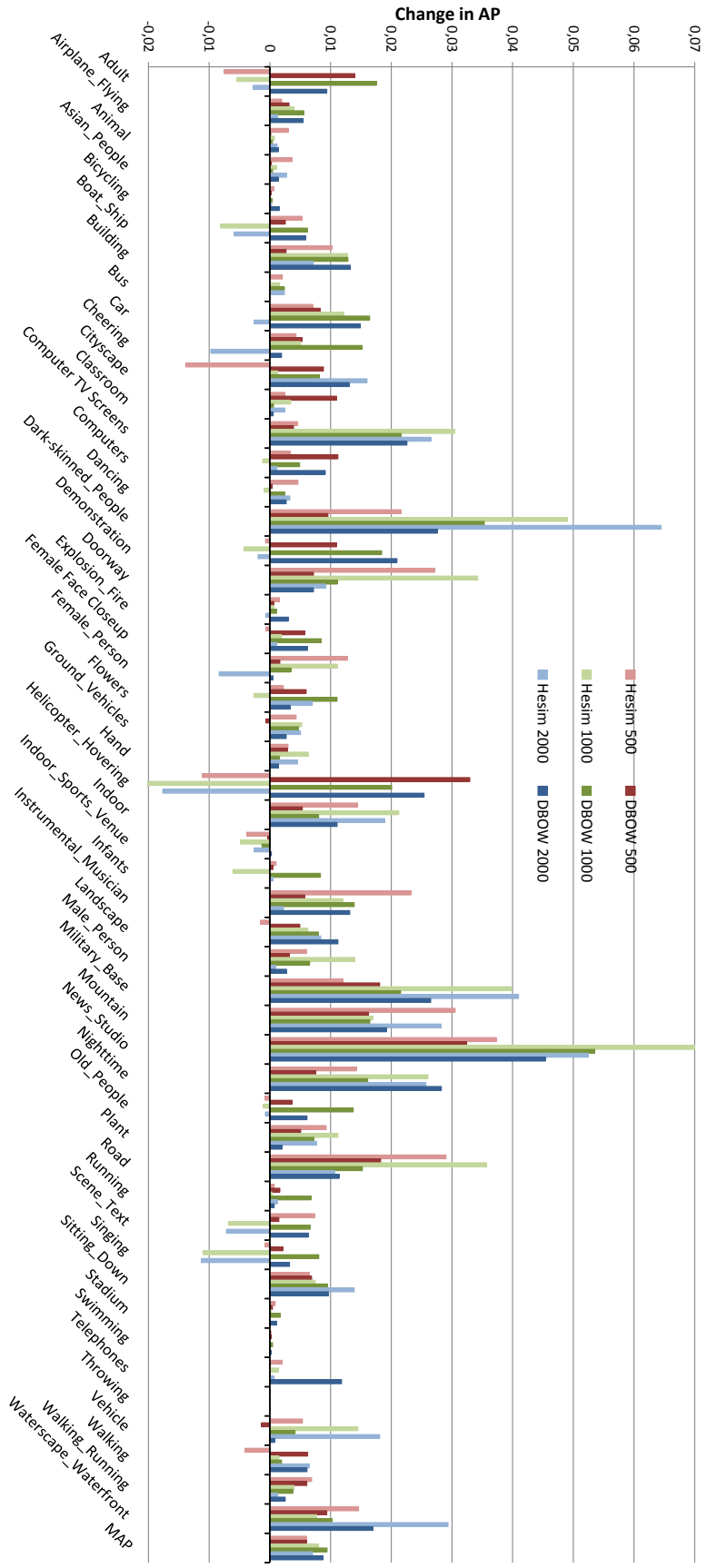


FIGURE 3.10: Concept scores for TRECID 2010: increase/decrease in Average Precision score over the baseline for the best performing HE similarity and DBOW methods for the 3 base dictionary sizes.

Chapter 4

Leveraging from Multi-label Data

For visual indexing or video concept detection the norm is to follow single label classification where annotated examples are used to learn classifiers for each semantic concept. In this paradigm, figure 4.1, each concept has its own set of positive examples and the correlation between concepts or any form of similarity is completely disregarded.

FIGURE 4.1: Single Label Learning Paradigm or One vs. All Learning

The performance of such a system is limited mainly because of the unavailability of sufficient annotated examples, descriptor noise and the semantic gap that is the representation difference between the high level concept and the low level feature. Finding the optimal parameters of the learner for each concept adds to the difficulty of this task. Since acquiring sane annotations for semantic concepts is expensive and some categories lack enough positive examples, we argue that making use of the multi-label nature of the video datasets and

jointly training concepts together can positively affect the performance of the learning task [139, 140].

This leads us to explore a new learning paradigm in which we try to combine concepts or labels and learn them together. A glimpse of this new paradigm can be seen in the figure 4.2 where all the concepts become part of some groups or meta-concepts. Classifiers are trained for these new groups or meta-concepts and results for each concept are somehow derived. The important questions raised here are: Can we group concepts together? Can we increase training resources for a concept by adding examples from a similar concept or a concept that shares some similarity and does it benefit? How do we define similarity between concepts? How do we differentiate two concepts or two meta-concepts? How do we acquire concept scores from the group scores? Can we decrease the eventual number of classifiers? In the sections that follow we set out to answer these questions by presenting some techniques to find similarities between concepts and combine them for training. We give detailed experimentation for each of them. We also briefly describe state of the art techniques working on multi-label datasets and multi-class classification specifically Hashing, Attribute learning, Binary Codes and Error Correcting Codes and describe the novelties in our methods. In what follows by concept we mean labels and also classes.

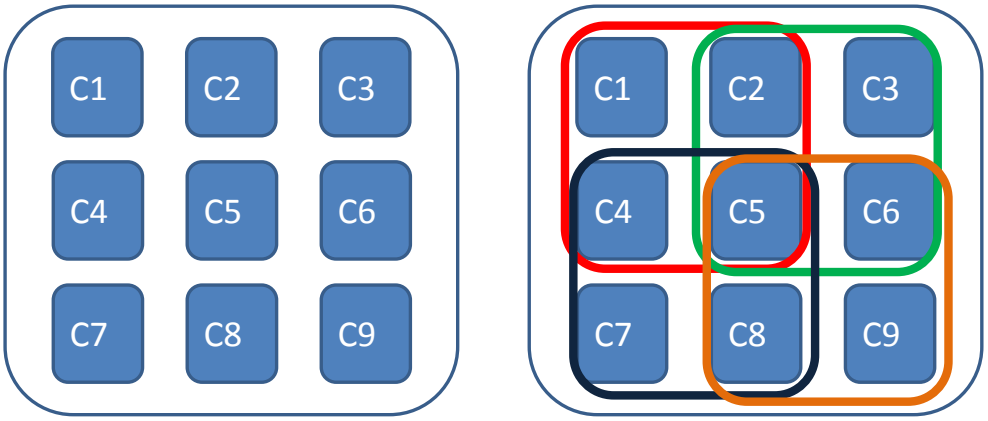


FIGURE 4.2: Single Label learning vs Groups of Labels learning

4.1 Learning from Multi-label Datasets and State of the Art

Typical image and video datasets are multi-label in nature in that the classes are not mutually exclusive. The visual content is very rich and usually comprises multiple objects or concepts, in a broader sense, in a single image or a video frame. A city skyline picture, for example, contains many objects and any video is typically tagged with more than one semantic concepts. We use this intrinsic multi-labeling with a twofold objective; to increase

the training resources per label and to increase the number of classifiers per label which will in the end be combined to try to maximize the performance on the training set.

Manually annotating the video data is a wearisome task and for many categories in the video datasets there is a lack of annotated examples. Also since manual annotations are distorted by the annotators point of view and are personalized important frames may be missed or in the worst case be wrongly annotated. But since most of the examples of a typical video dataset e.g. TRECVID are multi-labeled we can draw similarities between labels based on statistics calculated on the annotations or using some other similarity measures.

The argument we present here is that since there exists many commonalities among labels different concepts can complement each other for training examples if they are considered together for training. Considering them together means that if they are very similar they should be trained together to augment training resources and simultaneously they should be classified against each other in order to highlight their differences. An example could be to merge the examples of *Car* and *Road* together to train them as a strong *group* classifier and at the same time arrange for at least one of those to fall into another *group* so that they can be distinguished against each other for the cases when the car is in the garage for example or the road is empty or is crowded only by buses. Figure 4.3 differentiates the regular classification for video concept detection from joint group classification of concepts. Moreover the group classification can be looked in two different ways: **(i)** where a group of concepts is trained against all other concepts, and **(ii)** where a group is trained against another *distant* group or a group with least resemblance.

It is important to note here that the final classifier built based on joint concept information is a regular binary classifier as evident in the figure 4.3. Next we briefly describe some methods addressing multi-label classification and present some closely related domains before describing our methods to solve the problem.

4.1.1 Multi-label Classification

Content based retrieval, semantic indexing or concept detection can be naturally regarded as multi-label classification problems. In multi-label classification the aim is to learn from data labeled with possibly more than one label and predict a set of labels for an example. This combination of labels in the predicted set could not be present at the time of training. It is important to distinguish Multi-Class Classification, where each example is considered to have only one label while the classification results in one of more than two classes. As for the more general multi-label classification each instance can be predicted to have multiple classes.

[141] has divided the approaches to solve multi-label classification into two categories: **Algorithm Adaptation** and **Problem Transformation** methods. As the name implies in

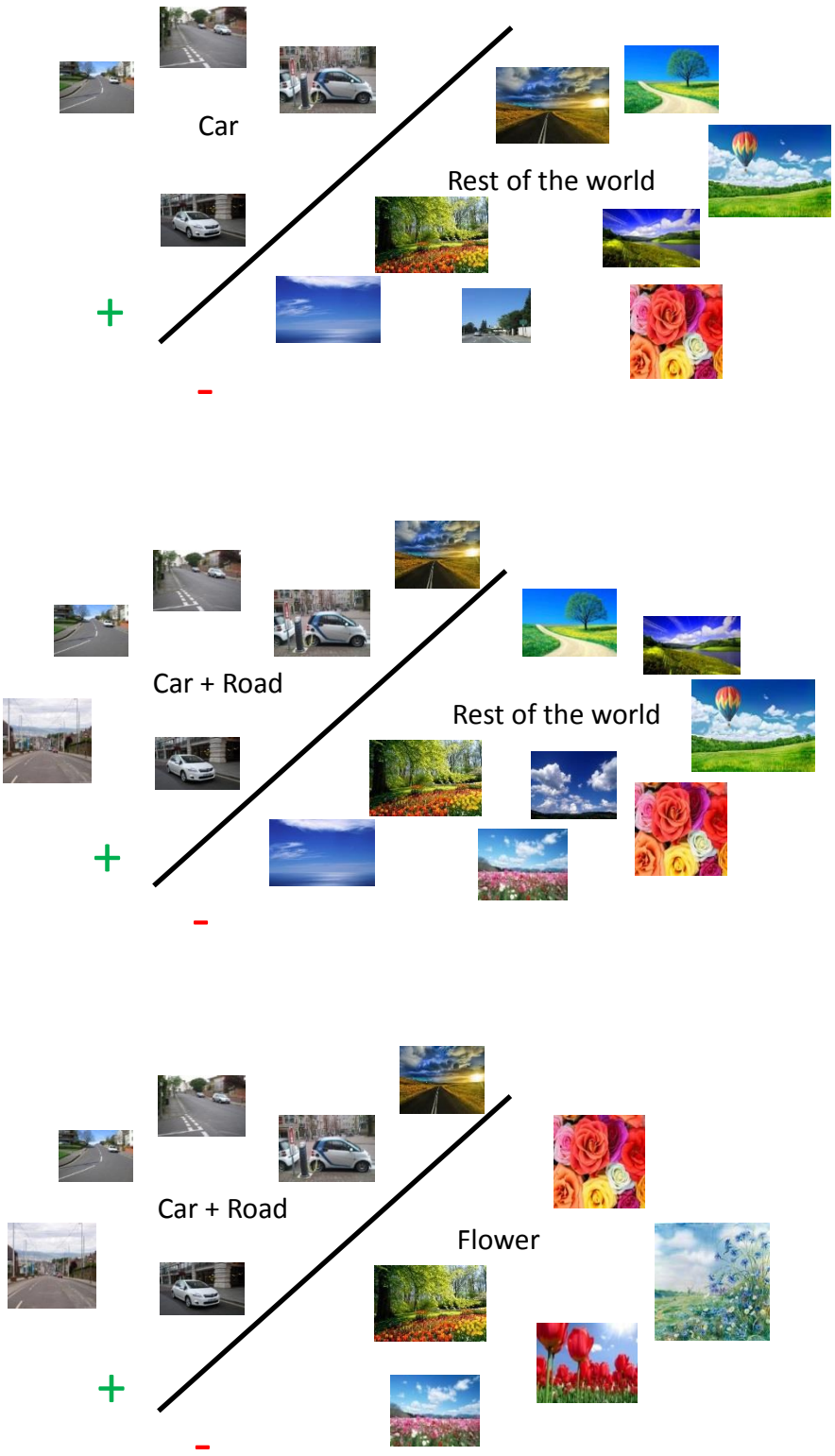


FIGURE 4.3: Single Label learning vs Groups of Labels learning

Algorithm Adaptation methods the learning algorithm is molded or altered to cater directly for multi-label data. More details and some popular examples can be found in [141–143].

Problem transformation methods have been most favored by researchers where the multi-label learning problem is transformed into single label learning problems and after the predictions multi-label results are acquired. The kind of multi-label classification we perform also falls under the umbrella of problem transformation methods for multi-label classification problems. We divide the concepts or labels into groups which are not mutually exclusive i.e. labels can co-occur in different groups. Then we merge all the annotations together within a group and train a single label classifier for that group. In the end we trace back classification score for each single label based on appropriate combinations of the groups classification scores. In fact in terms of multi-label classification the single label classification for video concept detection that we detailed at the start of this chapter is also a problem transformation. What we are trying to do here is to profit from the *multi-labelness* of the rich video data which is forgotten when handling each concept separately.

RAKEL is a well known multi-label classification algorithm that works on Label Powersets (LP). LP methods considers each distinct combination of labels that exist in the training set as a different class value of a single-label classification task [144, 145]. Subsets of labels of fixed size k are generated randomly and single label classifiers are learned for all the label combinations in the powerset of this subset. These single label classifiers may take into account label correlations if adequate number of examples are present. In the end each LP classifier predicts values for the k labels.

4.1.2 Knowledge Transfer and Semantic Sharing

Us humans have the great ability to recognize previously unseen objects by relating them to a known object. For example if someone sees a four legged animal (e.g. a puma) for the first time they can correctly identify the family (e.g. cat) and can make correct guesses about its characteristics (e.g. it is a mammal, carnivore etc.). This is possible because we identify the properties this new object shares with the other objects we know. This trend of sharing information between classes has been reflected in the research in computer vision to recognize unseen objects, classify objects with very few positive examples and to improve recognition of previously known classes. These techniques use similarity (or dissimilarity) between classes to build new features and are regarded as label sharing, knowledge transfer and feature adaptation in the literature.

Bart and Ullman [146] propose to minimize the prohibitive cost of collecting annotated training data by learning new classes with only one example using similarity with other classes. In [147] they use information from previous classes to select features for the novel class, also with only one example. Fergurs et al. [148] calculate semantic distance between two labels using the WordNet [149] hierarchy. The label sharing is used to build a *semantic*

matrix which reflects similarity between classes. This matrix is used in a semi-supervised learning framework to improve classification results on large image datasets. In [150] authors use linguistic knowledge bases to automatically determine the knowledge that is to be shared among classes to recognize objects.

Following we review certain techniques that implicitly or explicitly share information among classes for visual processing.

4.1.2.1 Visual Attributes

[151–153] introduced attributes that describe visual objects. Attributes can be physical for example visually discriminative parts like *leg* and *wheel*, or descriptive like *blue colored* and *striped* or a property that some object might have and other do not. In image and video retrieval a concept or more specifically an object is composed of a set of attributes. The goal is to learn about the object's attributes that either replaces the need to learn directly an object classifier or help in identifying the object category with complementary information. So if an image is predicted to contain so and so attributes it can point to a specific object or a class of related objects. Specifically an attribute can be represented by a binary classifier with 1 and 0 indicating whether the attribute is present in the object or not. An object can then be represented by a string or classifier outputs or a *binary code*. Each test image can then be associated with the closest object category based on its predicted binary code or the string of attribute classifiers output. Visual attributes have been used for object naming by being defined previously [151, 152, 154–156], however they can always be discovered on the go based on the training data [157, 157–159]. Moreover they have been useful in the case where the classes in the training and test set are disjoint [154, 160, 161].

Farhadi et al. [152] stress the importance of using attributes for describing objects with rare or no positive examples. So even if an object category had no examples for training a classifier, attributes can be identified for an example image explaining its properties and inferring important clues about the object's identity. Furthermore, learning objects in terms of attributes can overcome the intra class diversity. Thus if two examples of the same object (e.g. cars) are visually different they are most likey to contain similar key attributes that will help identifying the object. Third important contribution they believe attributes bring in is the ability to differentiate further among the objects of the same class, e.g. dogs with and without spots. For learning attributes they first select features that are able to distinguish the attribute within each containing class. For example to learn the attribute "'wheel"' they identify features to learn "wheel" of car, "wheel" of bike and "wheel" of cart etc.. Then all the features are use together to learn the "wheel" classifier over all the classes.

Lampert et al. [154] take attribute based classification a step further and perform learning for disjoint training and test datasets for object detection. They present two attribute learning methods to predict unseen test classes. The first is a straightforward mechanism

where attribute-class relationships are fixed for both training and test classes and attributes are learned using the training classes only. In the second method the attributes act as a layer connecting labels from training classes to the test classes. The multiclass predictions from the training classes on the test set yields a labeling of the attributes which in turn produce a labeling of the test classes. Yang and Mori [162] find correlations between pairs of attributes and model the attributes as continuous values. They build models for global attributes as well as class specific attributes. This allows to differentiate the attributes that are specific to each category and provide complementary information to the global attributes.

Kumar et al. [156] use 65 describable attributes for face verification which are represented by binary SVMs. Parikh and Grauman [158] discover new nameable attributes using human help that are discriminative. Russakovsky and Fei-Fei [163] mined relationships between object categories in the ImageNet [164] using a number of appearance-based attributes. Cai et at. [165] use semantic attributes for image search re-ranking. The semantic attributes they used are histogram based visual features extracted from various image regions. Authors in [166] take user feedbacks on images as well as attributes to refine search results of a content based retrieval system.

Parikh and Grauman [161] go beyond categorical labeling of attributes and present relative attributes. They learn a ranking function for each attribute that defines the degree to which the attribute is present in the image. These relative attributes are also used to define unseen categories in relation with the observed categories.

Use of attributes in large scale video retrieval or classification is rare, however [167] describes a video concept sparsely from a set of around 6000 weak attributes. Such weak attributes include classifiers scores on low level visual features and classemes, image distance to some randomly selected images based on the visual features and some discriminative attributes. Weak attributes for a query concept are found through a semi-supervised graphical model using correlation between the concepts and the attributes labels.

4.1.2.2 Hashing and Binary Codes

Closely related to attributes is hashing [41, 168] which tries to distinguish examples by assigning binary representations to individual images. The goal here is to distinguish the object classes while taking care of the within class variability. The solution assigns *compact* binary codes directly to the data points which are images in most cases. These codes are similar for semantically similar items and can be efficiently computed for new images. Retrieval is very fast as matching images is replaced with only calculating hamming distance between the binary *codewords* for images [41, 169, 170].

The binary codes should be able to distinguish between classes, should be flexible to cater disparity within class and should be easily calculable for new images [41, 159, 171]. Since finding binary codes for a dataset is an NP hard problem [41] there have been many

techniques to efficiently compute these codes for a given problem. Spectral hashing [41] calculates binary codes by thresholding a subset of eigenvectors of the Laplacian of the similarity graph.

Torralba et al. [23] use two data dependent approaches working on image neighborhood information to generate binary codes for large scale image recognition. Raginsky and Lazebnik [172] use random projections from a low dimensional mapping of the original high dimensional. Jégou et al. [25] use binary codes to efficiently utilize memory for very large scale image search. They use an efficient quantization jointly with dimensionality reduction to obtain the binary codes.

Rastegari et al. [159] combine attribute learning with generating binary codes and jointly learn all the attributes (binary codes) together for all the training data. They also validate their method for novel category recognition where the test set contained different classes than those present at training. Spherical hashing [171] assigns each image a binary code by dividing the feature space into hyperspheres instead of hyperplanes. Each image can lie either inside or out of a hypersphere and each hypersphere is indexed by a bit of the binary code. Two images for which the corresponding bits lie in similar hyperspheres represent a stronger bound than their bits lying on the similar side of a hyperplane. Moreover the length of the codes are half as those in the predecessor with improved performance.

4.1.2.3 Repetetive Codes

We can also look into multi-class classification for getting hints to solve the more general classification problem at hand. Hastie and Tibshirani [173] proposed to use pairs of concepts to solve the multi-class classification problem. Classifiers are built to distinguish each pair of concepts and in the end results are combined for the concepts. In this pair-wise method examples of one class are treated positive while examples of the other class are considered negative samples. Good classification performance can be expected when the two participating concepts are really different. For the cases where the difference is not so much and the concepts are rather alike the classifier strives to learn the properties that distinguish the two. Though this requires sizable amount of valid training data.

Another more general technique that directly models the relationship between labels is using repetitive coding or specifically Error Correcting Output Codes (ECOC) [174]. Targeted primarily to solve multiclass classification, ECOC divides the problem into many binary classification problems. Results from the binary classifiers are combined to obtain the multiclass prediction. The ECOC mechanism essentially consists of two steps. The first step is called encoding where a binary codeword is assigned to each class. Like hashing and problem transformation methods to solve multi-label classification problems, the encoding for ECOC can be data dependent or independent of the training data. Decoding is the second step where information from the binary classifiers is gathered to obtain the final predictions. The

individual classes are repeated in the binary classifiers and thus the final prediction contains error correction properties.

The framework builds binary partitions (dichotomies) of the set of labels in the coding step. This means that every label participates in a dichotomy with some labels on either side of the dichotomy. In the corresponding classifier examples from all the labels on one side are considered positive and examples from the rest of the labels as negative. For the labels a coding matrix M is defined where $M \in \{-1, +1\}^{|L|,N}$. The $|L|$ rows of the coding matrix are the codewords for each of the label and the N columns indicate the membership of the corresponding label to either of the +1 or -1 class. Important relationships between concepts can be coded using this framework. For decoding the hamming or the Euclidean distance between the predicted codeword and the codewords of all the training classes (labels) can be calculated to find the corresponding class. More sophisticated weighting strategies can also be adopted to infer the target class based on individual classifier performance or some other loss measure [175–177].

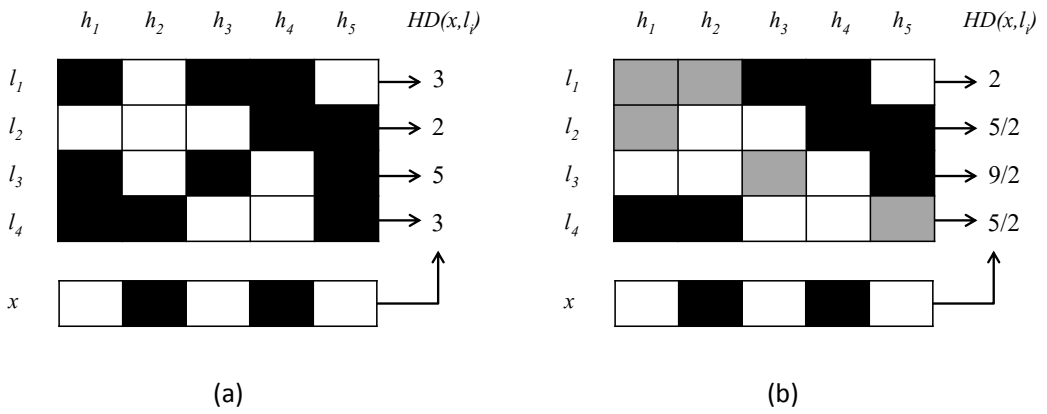


FIGURE 4.4: M matrices for 4-class ECOC representations: (a) Binary ECOC design. The codeword for the new predicted example x is closest to class having label l_2. (b) Ternary ECOC design where the gray entries correspond to the 0 in the dichotomies. Here x is assigned the label l_1 based on hamming distance.

ECOC has recently attracted research interest in the field of multi-label classification after conquering multi-class classification. Allwein et al. [178] introduce ternary ECOC for multi-class categorization problems which allows to increase the number of dichotomies thus encompassing better the data and label distribution. In ternary representation each label can either be a part of the classifier (dichotomy) with +1 or -1 bit or it can be omitted from a dichotomy with a 0 bit. This matrix M now looks like:

$$M \in \{-1, 0, +1\}^{|L|,N}$$

Ternary representation encompasses one-vs-all approach with one of the labels as +1 and all the rest are -1, and one-vs-one or the pairwise approach where apart from two labels all the rest are 0 in the dichotomies. Figure 4.4 shows an example of the M matrix for the binary and ternary ECOC and also shows a simple decoding using hamming distance HD.

Ternary ECOCs are also natural extension of ECOC methods for multi-label classification [176, 179, 180]. Authors in [181] perform class retrieval using ECOC on multi-label datasets. They use semantic distance and feature distance between labels in unison, and alter the ECOC decoding process to get ranked lists for each concept. Armano et al. [176] take ECOC from binary to ternary representation to cater multi-label Text Categorization. The dichotomies are built in a straight forward manner by using all the present combinations in the training label set.

Ferg and Lin [182] present an ECOC framework for multi-label classification problems and demonstrate its success on the Random K-labelset (RAKEL) algorithm. Contrarily [180] did not find ECOC to be suitable for RAKEL but found good classification results with Bode-Chauduri-Hocquenghem (BCH) ECOC which outperformed the binary ECOC. However they did not test ternary ECOC. Pujol et al. [183] add dichotomies to an already generated ECOC setup with certain dichotomies by minimizing classification error on validation data.

Wang and Forsyth [184] partitions the label space of 1000 categories to generate binary trees for multi-class image categorization. They outperform random partitioning of label space with ensemble (forests) of label trees where for each category the highest score is selected among the trees of the forest as the classification result. In one ensemble the error from the previous tree is used to generate the next one by using some validation data.

More recently the trend has been to find the most representative set of binary classifiers in the ECOC framework. This necessarily means using the training data to generate the dichotomies. Rocha and Goldenstein [185] reduce the number of required learners and add new ones that complement the existing set.

4.2 Group SVM

Learning all possible label combinations is an insurmountable task and practically the label sets that actually exists are very sparse [141, 144]. This sparseness thus helps greatly to reduce the classifiers to be learned. Methods to find label sets to be learned can be divided into data or label dependent and data independent approaches[141]. Data independent approaches randomly select label sets like RAKEL [144] while we propose a data dependent approach that exploits visual correlation between labels or concepts to find good label sets.

Our goal is to learn from multiple labels and minimize the number of multi-label classifiers [139]. We try to find intelligently the sets of labels to be trained together for learning, building multi-label classifiers or *group* classifiers as we call them. We propose to use the visual similarity between concepts to partition the label space into multiple overlapping groups and then learn classifiers for those groups. Thus we achieve multi-label classification through group based classification. The groups learned are effectively binary classifiers that combine annotations from different concepts and learn a 1-vs-all classifier on the new set of annotations. Individual concept labels are then inferred from the multi-label group predictions. The labels predicted are always the same and are defined at the time of making groups. The number of concepts belonging to one group is not similar.

Difference From Attributes and Hashing

Group based classification [139] is different from attributes in that a concept is defined by a set of multiple attributes while a group contains multiple concepts. Looking at it in another way an attribute also contains many concepts, or more specifically it is present in many concepts, but the number of attributes to be learned is far greater than the expected number of groups to be learned. Furthermore each concept is identified by a unique combination of the groups. We are trying to improve video concept detection performance with using a small number of classifiers to be learned. Definition of attributes is very specific to the type of dataset and the task at hand. The number of object specific attributes increases rapidly with the addition of more diverse content. Also the attributes are usually named in advance but the groups of concepts or labels are not fixed or pre-defined.

In hashing techniques two examples of the same concept have very similar binary codes [159, 171], or two (visually) close examples have very similar binary codes. Contrarily in our grouping approach codes are assigned at concept level. Thus two examples of the same concept have the exact same binary codes as they belong to the similar set of groups. Two concepts that are closely related to each other may have very similar binary code, i.e. they belong together to many of the similar groups.

We explore a quick way of grouping visually close concepts together which outperforms a method of randomly grouping labels and state of the art multi-label classifier *RAKEL* for different group sizes. We surpass significantly the concept detection performance over the

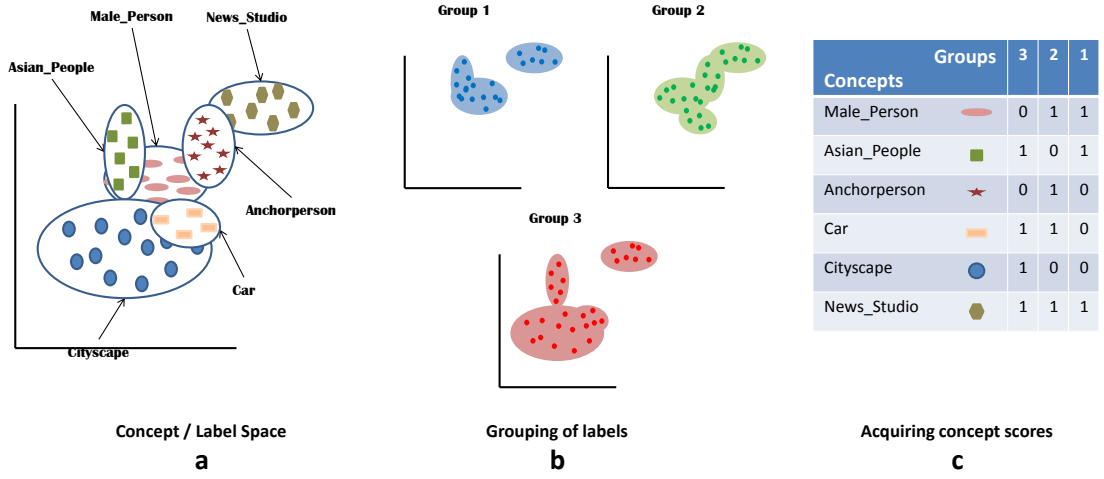


FIGURE 4.5: Grouping of concepts into non mutually exclusive partitions: a) A simplified space is shown with some labeled (and multi-labeled examples. b) Grouping of concepts. Concepts may appear in multiple groups. c) Each concept is a unique bit string and no group is empty.

baseline with fusing information from as little as 10 group classifiers for a total of 50 concepts on the TRECVID 2010 dataset.

4.2.1 Group Based Classification

Similarity of concepts can be judged in many different ways, including web semantics information, ontology rules or relationships between concepts provided with some video databases like for example TRECVID 2010 [18]. Example of such rules are *Anchorperson* is *Adult* and *News Studio* contains *Anchorperson*. To quantitatively express the similarity of concepts, intersection of common annotations can be used to find a similarity index between two concepts. Also feature vectors belonging to shots containing concepts can be used to find distance between two concepts. The criterion for clever grouping of concepts we do uses feature vectors for finding similarity between concepts to group them together.

After the concepts are grouped together into different groups, each concept is assigned to a number of different groups. The idea is that each concept is uniquely identified by a combination of outputs of certain groups. Thus if the concepts *Car* and *Road* appear in the same group, one of them should belong to at least one of the other groups to differentiate it from the other. In other words each concept is represented by a *unique* bit string with length equal to the number of total groups. Each bit of the bit string represents a group and the value of the bit is 1 if the concept belongs to the specified group. Figure 4.5 shows our intuition of the group based classification scheme. Ideally for C concepts $\log_2 C$ group classifiers are enough for learning as each concept can be identified with a unique bit string.

4.2.1.1 Clever Grouping

We use average feature vector for each concept (label) to find the alikeness between concepts. Average feature vectors v_c for each concept $l_c \in L$ are obtained by averaging all the feature vectors for images containing the concept from our training set. The total number of labels is $|L|$ and so is the number of average vectors. The feature vectors are Bag of Words histograms [20] as described later in the experiments section. Let (x_i, y_i) be a training example where y_i is a vector of labels in the multi-label setting. Average feature vector for a concept is given by:

$$v_c = \frac{\sum_{c \in y_i} x_i}{|c \in y_i|} \tag{4.1}$$

The clever grouping criterion we use is based on visual similarity between concepts i.e. we use the similarity or the inverse of the distance between the average features as the closeness. Figure 4.6 shows the average feature vectors and also the visual space. This is the space where the concept or label features are projected and represented by the distances between them.

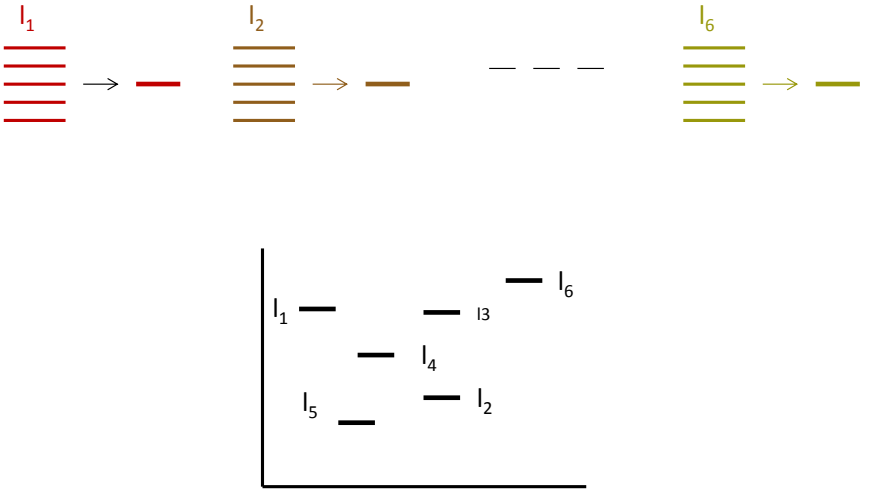


FIGURE 4.6: Average visual features and the induced *visual* space.

Now we are going to generate D groups based on the visual similarity between concepts. First we consider the scenario where where the number of groups is less than the number of concepts $D < |L|$.

Grouping for D < |L|

Groups are generated in a reverse mechanism. That is to say that concepts are assigned to the groups one by one. So for example we start with concept 1, assign it to the groups and carry on with the next concept. To start the C average vectors are first clustered into D

centers using k-means clustering with random initial centers. After clustering we generate a list for each average feature vector containing top n closest centers. We follow with an allocation like soft assignment [76].

For each concept the list is sorted with the closest center at the first rank, second closest ranking second and so on. Each concept is always assigned to the closest group. Next the concept is assigned randomly to the next closest center with decreasing probability. The decrease in probability is proportional to the increase in the distance to the next closest center. When each concept has been assigned, D groups are generated corresponding to the D cluster centers. Clustering is done so that the outliers (concepts whose average features are far from others) are considered for grouping as well when $D < |L|$. Figure 4.7 shows the 3 clusters generated for the dummy problem shown previously in the figure 4.6 with 6 concepts. The three cluster centers correspond to the 3 groups generated with different concepts. Note that each concept appears in different groups and this combination is unique, i.e. no two concepts appear in exactly similar groups all over, figure 4.7.

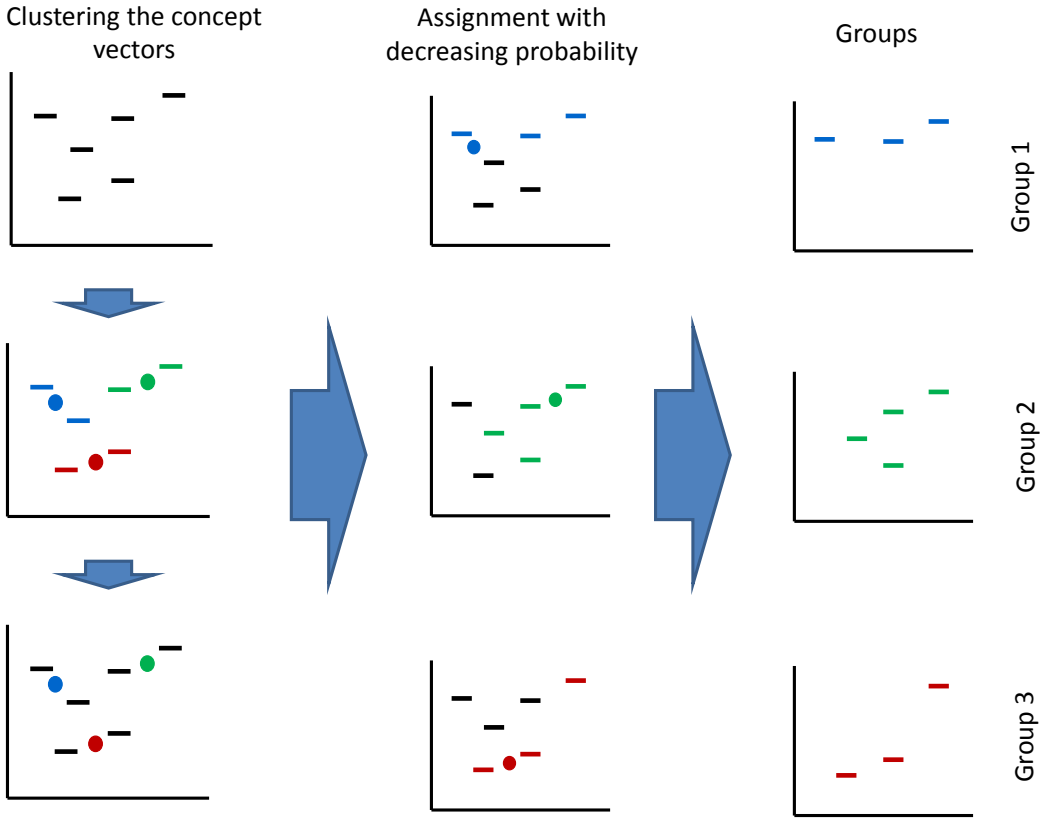


FIGURE 4.7: Clever grouping when then number of groups is less than the number of concepts.

Grouping for D $\geq$ |L|

in the case where the number of groups is greater than or equal to the number of concepts, i.e. $D \geq |L|$, we drop the clustering mechanism. To create the first group we start with randomly selecting an average feature vector of a concept and drawing a number n randomly. Then n

closest concepts with minimum distance to the selected concept are assigned to the first group along with the randomly selected concept making the first group. This process is repeated until D groups are generated. The n closest concepts are first sorted with the concept with the minimum distance ranked at first position and are assigned to the group with decreasing probability. This decrease in the probability of assignment is proportional to the increase in distance of the selected concept to the next closest concept. 6 groups are generated for the toy problem of figure 4.6 and are shown in the figure 4.8.

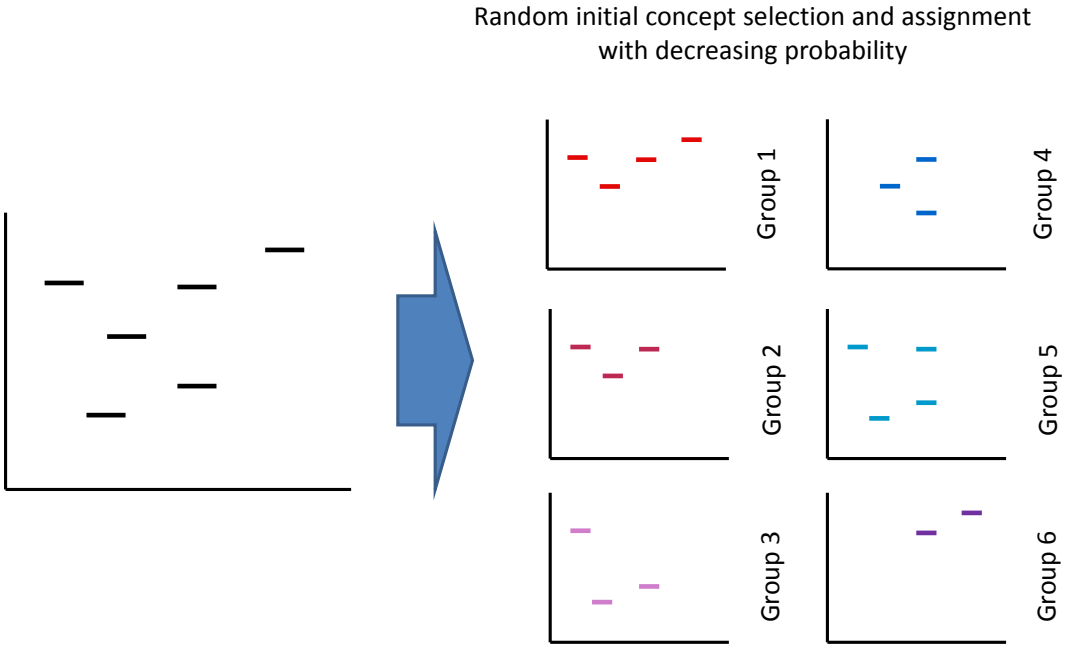


FIGURE 4.8: Clever grouping when then number of groups is greater than the number of concepts.

Binary codes and conflict resolution

The concepts are assigned sequentially that is the bit strings are generated sequentially. Each concept can be considered as represented by a binary code with each bit representing the corresponding group. A 1 indicates that the concept is present in the group and 0 is put in case the concept does not make part of the group. In case of a conflict between the assignment of two concepts, i.e. when two bit string are exactly the same, the two concerned bit strings are regenerated until the conflict is resolved. Since during assignment there is some randomness while keeping the visual similarity a priority we expect to resolve the conflict with a few retries. In this way each concept is uniquely identified by a different combination of groups. The case where $D \geq C$ there is rarely a conflict but if there is, the grouping is regenerated. The maximum value of n is fixed as 12 in our experiments with $C = 50$. It can be confirmed from figures 4.7 and 4.8 that each concept induced a unique combination of groups. For figure 4.7 where the number of groups is less than the number of concepts the

bit strings for each concept are: (l_1: 001), (l_2: 110), (l_3: 011), (l_4: 010), (l_5: 100) and (l_6: 111). The maximum number of concepts represented by D is $2^D - 1$.

We then determine the labels of the members of the groups. All the examples belonging to those labels become part of that group. More specifically two examples that are visually very close do not necessarily end up in the same group unless their respective average feature vectors are close. Unlike attributes these groups are complex entities or complex attributes as they combine annotations from many objects (concepts).

Random Grouping

We compare clever grouping of concepts with random grouping. In random grouping no similarity criterion is used rather concepts are grouped randomly into groups. For each group n is randomly drawn and then n concepts are randomly selected from which bit strings are acquired for each concept. The process is repeated if there is a conflict between the bit strings of any two concepts. This process remains similar whether the number of groups is smaller or greater than the number of concepts.

Acquiring Concept Score

For both grouping criteria an example is considered positive if any of the participating labels (concepts) is positive for that example. The number of positive examples increase rapidly for a group with concepts having disjoint positive annotations. All the remaining examples in the training dataset are treated as negative examples. Each group is then trained in a 1-vs-all fashion. Prediction on a test frame generates the score $s(f|g)$ which is the score of the test frame f for the group g. Concept score is then calculated on the normalized groups scores as:

$$s_g(f|c) = \sum_{c \in g} s(f|g)$$

giving the score of the frame f for concept c. We do not subtract the scores of negative groups i.e. the groups which do not contain c as we found experimentally that using this information worsens the results.

4.2.2 Experiments and Results

We present here experiments carried out on the TRECVID 2010 datasets.

4.2.2.1 Experimental Setup

Dataset and Features: We have used the TRECVID 2010 IACC [18] dataset containing 11644 internet videos. This comprehensive dataset is divided into the training part with 3200 videos of 200 hours with a total of 119,685 keyframes. Rest of the approximately 8000 videos of 200 hours containing 146,788 keyframes are used for testing purposes. For testing the performance of various multi-label or group classification based systems on video concept

detection we have used the list of 50 concepts from the TRECVID 2011 Light Semantic Indexing task.

We use 128 dimensional SIFT features [63] to describe local patches extracted using a Dense grid of points on the video keyframes [186]. The points on the grid are distanced 8-pixels apart. All the SIFT descriptors from the training set are then used to build a 500 word visual dictionary using k-means. For classification we have used linear SVM to learn from a suitable feature map (homogeneous kernel map) built by the histogram intersection kernel [137, 138].

We have used Average Precision (AP) to measure concept detection performance as used in TRECVID semantic indexing benchmark [18]. For the overall performance we use Mean Average Precision (MAP) of 50 concepts.

RAKEL: For RAKEL we fix the value of $k = 3$ which is also known to give the best results [144]. We adapt the RAKEL algorithm for Average Precision in that we generate score for each shot so that a sorted list can be generated for each concept. Each LP classifier is a multi-label classifier that gives classification scores for the k concepts. We call a k-labelset a group with k concepts and each group predicts scores for each of the k concepts. In the end to obtain the score for each concept, scores from all the LP (group) classifiers, of which that concept is a part of, are added and normalized.

Late Fusion: We have also used late fusion to combine the baseline results with the 3 group based approaches. Weighted linear fusion is used in order to obtain a single output score for each concept $s(f|c)$ that is used to rank the video frame f for the concept c:

$$s(f|c) = w_s s_s(f|c) + w_g s_g(f|c)$$

where $s_s(f|c)$ and $s_g(f|c)$ are the concept scores acquired from the single label classification (baseline) and the group based approach respectively. The scores are rescaled according to one another using min-max normalization. The weights w_s and w_g are optimized over the development set.

As all the three approaches include some randomness we have repeated each experiment 5 times. We show the mean and standard deviation for the scores.

4.2.2.2 Results

The Semantic indexing results on the TRECVID 2010 test dataset for various approaches for the list of 50 concepts are presented in figure 4.9, containing: i) OVA: One vs All or single label classification which is also the baseline, ii) Random grouping, iii)Random-OVA: fusion of baseline with random grouping, iv) clever grouping, v) Clever-OVA fusion of baseline with clever grouping, vi) RAKEL, and vii) RAKEL-OVA: fusion of baseline with RAKEL. The

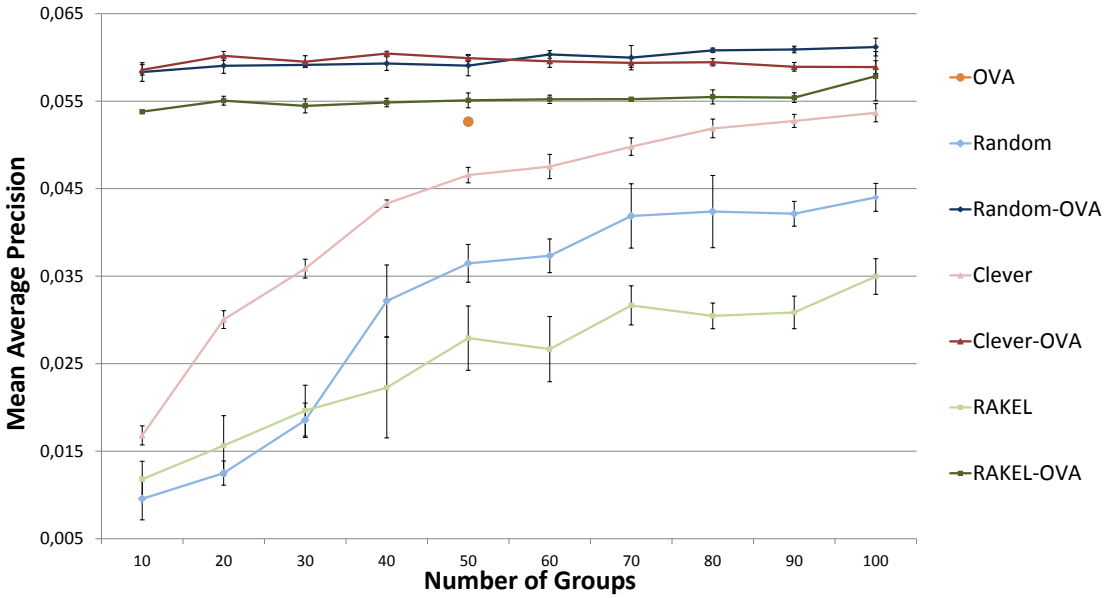


FIGURE 4.9: Mean Average Precision for 50 concepts for different grouping criteria and their fusion with the baseline.

circle in the center shows results for the One versus all classification which is our baseline. For baseline each concept is treated as a separate label and a 1-vs-All SVM is trained for that label using all of its positive examples.

Performance of various grouping approaches increases almost linearly with the increase in the number of groups with random grouping of concepts performing better than RAKEL for almost every group size. Intelligent grouping significantly outperforms the other two techniques and approaches single label classification performance for training around 80 intelligently formed groups. Further increasing the number of groups increases marginally the performance over the baseline.

Figure 4.9 also presents the results of fusing various group based techniques with the baseline. Fusion with RAKEL improves concept detection performance over baseline and increases linearly with the increase in the number of groups. The best performance is acquired using 100 groups with 10% overall increase in the indexing performance over the baseline. Clever grouping when fused with single label classification provided best improvement for a very few number of groups trained. This is observable in the upper left part of the figure 4.9 as fusing concept scores from only 10 clever groups with the baseline improves MAP score from 0.0527 to 0.0592 with 12% increase in performance. Using 20 intelligent groups the improvement is 15%. The fusion performance is good with clever grouping till the number of groups is 40 and after that the MAP decreases linearly, when the number of groups equals or is greater than the number of concepts. With random grouping the trend is somewhat similar to RAKEL as the fusion performance increases with the increase in the number of groups. Although the MAP here is better than that of RAKEL for every grouping. Fusion results with random grouping lingers close to the fusion results with intelligent grouping.

The performance is slightly inferior to intelligent grouping for up to 50 groups and then outperforms intelligent grouping as the number of groups increases further. Best MAP score is 0.0612 which is a 16% increase over the baseline observed with 100 random groups used (upper right corner).

Analysis of Fusion Weights: Although intelligent grouping outperforms random grouping for concept detection for all group sizes this does not stay the same when the group based scores are fused with single label classification scores. To further analyze this we look closer at the fusion and investigate the fusion weights assigned to the concepts for different number of groups. Figure 4.10 plots the evolution of the fusion weights for the three multi-label approaches for different group sizes derived from the training data. The figure shows the average of the weights assigned to each concept score derived from the group based approaches compared to concept scores from individual concept learning in the linear fusion. The fusion weights increase with the number of groups for the three approaches as more groups means better classification at concept level except when there is over-fitting or when the group based approach performs worse than the baseline for a certain concept. For clever grouping the average weight quickly reaches the level where both the group based and the single label approaches contribute almost equally to the final score. Thus when the number of groups equals 50 both approaches contribute exactly equally for the final performance and as the number of groups increases more weight is assigned to clever grouping on average. In figure 4.9 we see that the clever grouping approaches single label classification with the increase in

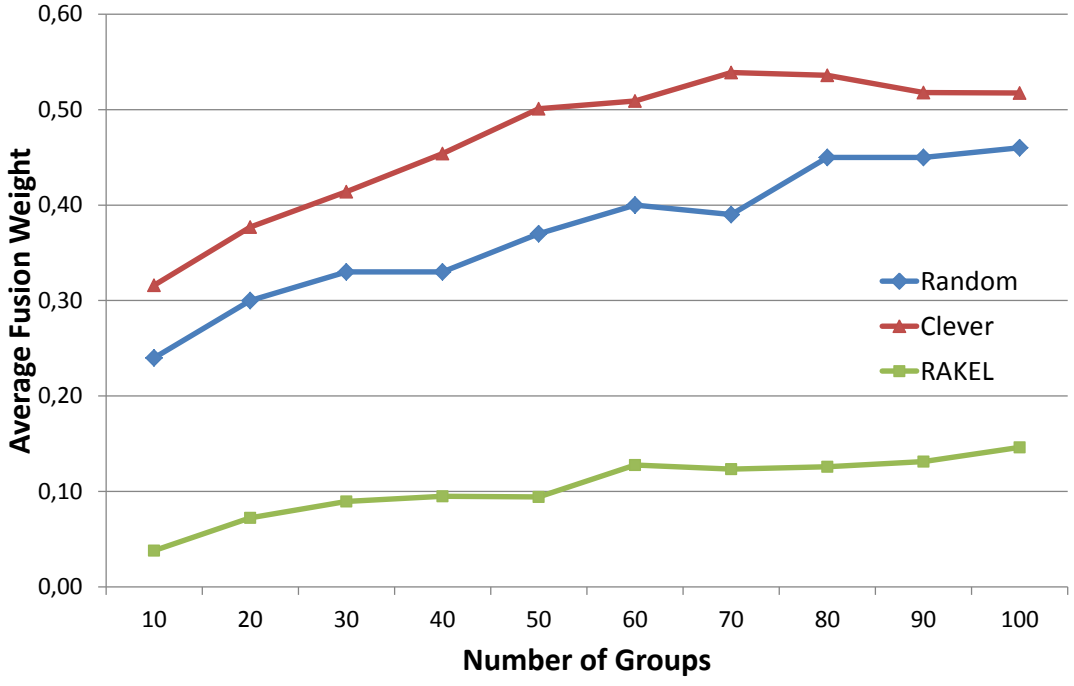


FIGURE 4.10: Evolution of the fusion weights with the increase in the number of groups.

the number of groups. Thus adding more and more groups with concepts grouped on visual similarity will always converge to single label concept wise classification and fusing those two together will only results in marginal improvement.

Note that this is not the case for random grouping technique as the maximum weight assigned on average is around 46%. In other words random grouping contributes around 46% to the fusion even with 100 groups. Thus as the performance of random grouping is always inferior to baseline in figure 4.9, fusing it with baseline always brings in complimentary information and improves performance with the increase in number of groups. However clever grouping does bring more useful information in fusion than random grouping if number of groups is kept inferior to the number of concepts. Thus we are able to improve 12% and 15% over the baseline with only 10 and 20 new classifiers trained respectively in addition to 50 single-label classifiers.

One explanation for the not so good performance of RAKEL is that the label set in the TRECVID dataset we used is very sparse i.e. only a few combinations of labels are possible. In the RAKEL mechanism each group has $k = 3$ and thus up to 8 single label classifiers are trained for each group. For good classification results the number of positive examples for each combination of labels should be adequate [144]. We have found that in the Label Powersets the number of examples for the label set {0, 0, 0} where all the 3 concepts are negative dominates the number of examples for other label sets. This complicates things as the label sets like {1, 0, 0} or {1, 0, 1}, where one concept is truly distinguished against others have very few positive examples for training. The final score for each concept in the end is thus dominated by the negative score of the classifier trained on examples from the label set {0, 0, 0}. From the TRECVID 2010 training data and our setting of the RAKEL algorithm we have on average 6188 positive training examples for the label set {0, 0, 0} for every LP classifier compared to only 168 positive examples for other label sets. Thus an LP classifier lacks the examples for the truly discriminative label combinations owing to the relatively poor performance.

4.3 Label Clustering and Repetitive Codes

Since classes share many examples with other classes it is reasonable to try to identify meaningful combinations among classes. These combinations are not mutually exclusive but are unique as each set encodes a specific partitioning of the labels. Looking at it differently each set adds or glorifies a certain aspect of the concept(s) within the combination. This is true for the group based classification presented in the previous section.

Now we try to look at the similarity as well as the differences between the concepts. The concepts are divided into partitions rather than groups. In the previous section a concept was part of different groups and the combination was unique for each concept. So the difference between two close concepts was that they differed in at least one group. Here we rely on the information that a concept is *like* some concepts and *not like* or *different from* some others. This kind of grouping and division is shown in figure 4.3-c, at the start of this chapter, where positively labeled examples from certain concepts are placed on either side of the classifier.

We identify opposing partitions from the set of labels that satisfy some similarity criterion. Labeled examples are then gathered for both the partitions and are trained with a binary classifier. Following questions are raised for a successful construction of such a system: **(i)** How the labels should be divided in partitions, **(ii)** what should be the size of each partition, **(iii)** how many different divisions of the labels are required and **(iv)** how each label can be differentiated. We briefly answer these questions here before responding to them in detail with our proposed approach in the following subsections.

It is important to first recap about Error Correcting Output Codes (ECOC) that have been successfully employed to solve multiclass classification problems [174] and are also making their way into multi-label classification [176, 179, 180]. The mechanism usually contains numerous suboptimal classifiers and a single label is repeated in many of those classifiers. The repetition tends to minimize the effect of the errors made by single classifiers by combining the predictions from all the classifiers for that label. Our partitioning and training proposal fits the ECOC mechanism as each division of partitions is a dichotomy where some labels lie on either side. Each row of the ECOC M-matrix represents a codeword for a label which we should enforce to be unique. A classifier is trained for each dichotomy and predictions from each dichotomy are combined to decode the final result.

While there have been approaches to randomly assign binary codes to classes and generate the ECOC M-matrix [178] we use a data dependent approach exploiting the resemblance between concepts to generate the repetitive codes. Mohamadi et al. [181] use semantic distance and feature distance between labels in unison for the ECOC decoding process to get ranked lists for each concept. We also use a combined similarity measure but contrarily we used is for the encoding. The similarity criterion is based on the distance between visual features of the concepts and the difference between the annotations. This similarity is used in a clustering framework to find partitions of the labels. Concepts that are visually or

semantically close should be part of the same partition or should be on the same side of the dichotomy. Thus if we are to assign from the set of concepts {*Car*, *Road*, *Beach*} to two partitions, it is reasonable to assign *Car* and *Road* to the same partition.

We use parameters to change the similarity between concepts as well. As we generate more and more dichotomies it is important to differentiate between concepts that are too similar. Carrying on with the previous example if *Car* and *Road* always lie in the similar partition they will not be distinguishable. So the similarity between *Car* and *Road* is diminished after they have been assigned to the similar partition. If they are assigned again to the same side of the dichotomy the similarity is further diminished. This is repeated until each concept has a unique codeword.

Difference From Group Classification, Attributes and Hashing

This technique adapts ECOC for multi-label classification and is essentially different from the group based classification presented before. In grouping only the similarity between concepts is used to build groups, for which classifiers are trained considering all the other examples as negatives. Here partitions are made based on similarity as well as dissimilarity among concepts. Moreover we do not use *all* the negative examples provided by the dataset rather only the positive examples from both the partitions lie on either side of the classifier.

This is again essentially different from attributes as we do look at the properties that certain classes have but simultaneously these properties are absent from other classes. Moreover attributes require manual definition and labeling and also the knowledge of presence or absence in concepts. Hashing works at image level while here again we build codes for classes and all the examples of the said class should have the same code.

4.3.1 Proposal

We use the similarity between concepts to explore the hidden structure (relationship) between them. This exploits complementary information among concepts and improves classification. The repetition is expected to further improve performance with error correction. In the training phase we build the codes for each class and then train classifiers for the dichotomies. It is worth mentioning here that in our approach one dichotomizer does not divide all of the labels. This means that we are dealing with the ternary ECOC where the codeword for each class contains bits that can take values in $\{-1, 0, +1\}$. The bit value of 0 gives the flexibility to classical ECOC to cater multi-label data by *not* considering some of the labels in the dichotomies. So a dichotomizer for one bit of the code will only be trained to distinguish between examples of the participating labels whose corresponding bits are -1 and +1. This suits us perfectly as we want our partitions to be compact with a few *similar* concepts distinguished from the other *dissimilar* partition with also with a few concepts. The length of our class codeword is N bits and so we build N dichotomizers, or train N binary

classifiers. For a test sample we acquire predictions from all the classifiers and generate score for each concept using our specific decoding.

Distance or similarity between classes can be calculated by using a number of relationships that exist between classes. Many classes share common examples in video datasets and this information can be used to find similarity e.g. the overlap or intersection of examples. Linguistic knowledge bases have been used to find the information shared among classes [150]. Ontological rules are usually given with datasets that dictate relationships between classes. Furthermore these relationship can be hierarchical or direct. WordNet [149] has also been used to find the semantic distance between classes [148]. We use the distance between features in unison with a semantic distance that we calculate directly from concept annotations. This distance (or similarity) is used to build a similarity vector for each concept which contains pairwise similarity values with all the other concepts. These similarity vectors are used to automatically find the codewords for each class. In addition we weight the similarity between every pair of concepts and these weights change dynamically during the construction of the codewords. Finally for each bit of the codeword we learn a binary classifier that separates examples for which the bit value is +1 from those for which the value is -1. Details of the system are presented in the next subsections followed by detailed experimentation.

4.3.1.1 Code Generation by Clustering

We calculate distance between the visual features of the labels using the training set as done in the group classification technique. For that we first calculate the average feature vector v_c for each label l_c as done in equation 4.1. Then $d_v(i,j)$ is the Euclidean distance between l_i and l_j which is referred to here as the visual distance since it finds the distance between the visual features of two labels.

This distance is combined with the semantic distance which is calculated using the intersection of positive annotations for the examples present in the training dataset. For each label c we have a set of positive annotations A_c and since we are dealing with multi-label data the semantic similarity between two labels is given by:

$$sim_s(i,j) = \frac{|A_i \cap A_j|}{|A_i \cup A_j|} \tag{4.2}$$

which is the ratio of common annotations between labels i and j. Here common annotations mean all the examples which are positive for the two labels in question.

Since we use the distance between concepts in the ECOC generation process we negate the semantic similarity to find the semantic dissimilarity as: $d_s(i,j) = -sim_s(i,j)$. The total dissimilarity or distance between labels is the weighted sum of the two measures

$$d(i,j) = \lambda * d_v(i,j) + (1-\lambda) * d_s(i,j) \tag{4.3}$$

The factor λ weights the contribution from the visual and semantic similarities.

We use this distance between two labels directly to find the partitions of labels. The number of labels that make part of a partition depends on a measure of distortion within the partition. The weight we use to alter the similarity between two labels is called repulsion between the two labels. This repulsion is changed dynamically during the ECOC generation process to avoid the same labels going to the same partitions over and over again. As the name of the variable suggests repulsion is increased between labels that were in the same partition in the previous iteration. Similarly we reduce the repulsion between concepts that never fall in to the same partition. The factor $repulsion_{ij}$ that weights the distance between lablels l_i and l_j is multiplied by the distance from equation 4.3 to find the final distance:

$$dis(i,j) = d(i,j)repulsion_{ij} \tag{4.4}$$

The algorithm to build the ECOC is presented in the figure 4.11. The set of labels is repeatedly partitioned to generate dichotomies until all the labels have a unique ternary codeword b_i. Repulsion is initialized as a value which is the same between all labels. To generate one set of opposing partitions or a dichotomy seed labels are selected that are the farthest from each other. The first seed is the one who has minimum cumulative repulsion with all other labels. This ensures the selection of the label that has never (or hardly ever) been selected before. The second seed is the farthest label according to the distance in equation 4.4. Two partitions L_1 and L_2 are maintained and closest labels are assigned at each iteration until the partitions are distorted. The closest label to a partition is found using the distance of label to the partition as:

$$dis(i, L_k) = \frac{\sum_{j \in L_k} dis(i,j)}{|L_k|} \tag{4.5}$$

where $dis(i,j)$ comes from equation 4.4.

After the assignments the labels that distorted the partitions are removed. The labels in L_1 are assigned the bit value of +1 and those in L_2 get the value for the k'th bit -1. All the other labels are assigned 0 for their respective bit value. Repulsions are also updated after the assignments: repulsions between the labels of the same partition are increased by $\alpha - \beta$ and repulsions between other labels are decreased by β. β is usually smaller than α and could be 0 as the convergence can be solely controlled by α. We have experimentally fixed the values of α and β to 0.3 and 0.05 respectively. The partitioning stops when each label is assigned uniquely to partitions.

Figure 4.12 shows the procedure to find the distortion within a partition. Each partition is considered a cluster within the space represented by the distance between labels. As a partition grows more concepts are added that are far from the center of that cluster. The distortion is controlled by the parameter Θ for which different values are tried in the

Input L: a set of labels
Output Error Correcting Output Code for each label $M(l,n), l \in L, n \in \{1,N\}$
Initialize
$repulsion_{ij} = 1, i, j \in L$
$t \leftarrow 1$
repeat
  $\forall i \in L \;\; repulsion_i \leftarrow \sum_{j \in L} repulsion_{ij}$ ▷ Select seeds
  Select $p \;\; s.t. \;\; repulsion_p \leq repulsion_j, j \in L$
  Select $q \;\; s.t. \;\; q : \underset{q}{\operatorname{argmax}}\, dis(q,p)$
  $L_1 \leftarrow p$
  $L_2 \leftarrow q$
  $L \leftarrow L \backslash p, q$ ▷ Remove seeds from the full label set
  repeat
    if $DISTORTION(L_1) = 0$ **then**
      $L_1 \leftarrow L_1 \cup \underset{i}{\operatorname{argmin}}\, dis(i, L_1), i \in L$
      $L \leftarrow L \backslash i$
    end if
    if $DISTORTION(L_2) = 0$ **then**
      $L_2 \leftarrow L_2 \cup \underset{j}{\operatorname{argmin}}\, dis(j, L_2), j \in L$
      $L \leftarrow L \backslash i$
    end if
  until $DISTORTION(L_1) = 1$ **and** $DISTORTION(L_2) = 1$
  $L_1 \leftarrow L_1 \backslash i$ ▷ Remove the labels that distorted the partitions
  $L_2 \leftarrow L_2 \backslash j$
  $M(t,i) \leftarrow 0, i \in L$ ▷ Assign binary codes
  $M(t,i) \leftarrow +1, i \in L_1$
  $M(t,i) \leftarrow -1, i \in L_2$
  $repulsion_{ij} \leftarrow repulsion_{ij} - \beta, i, j \in L$ ▷ Update repulsion values
  $repulsion_{ij} \leftarrow repulsion_{ij} + \alpha, i, j \in L_1$
  $repulsion_{ij} \leftarrow repulsion_{ij} + \alpha, i, j \in L_2$
  $t \leftarrow t + 1$
until $\exists M(i,) = M(j,), i, j \in L, i \neq j$

FIGURE 4.11: ECOC construction

experiments. If according to Θ the farthest label is far enough no further labels are added to the partition. Θ directly affects the number of labels assigned to each partition and thus it also affects the convergence of the ECOC generation. A bigger Θ leads to the generation of fat partitions with more labels while thinner partitions are made with a smaller value for Θ. Note that at least two labels make up a partition if $\Theta > 1$.

For generating the training set for classifiers all the examples x_i belonging to the labels in the first partition are treated as positive examples and the examples belonging to labels in the second partition are considered negatives. So the positive and negative examples for each classifier are actually only the positive examples of concepts on either side and this excludes

```
1: procedure DISTORTION(L)
2:     if |L| = 1 then return 0                ▷ A partition cannot have only one label
3:     end if
4:     dmin ← min dis(i, j)  i, j ∈ L, i ≠ j
5:     dmax ← max dis(i, j)  i, j ∈ L, i ≠ j
6:     if dmax > θdmin then
7:         return 1                            ▷ The partition is distorted
8:     else
9:         return 0
10:    end if
11: end procedure
```

FIGURE 4.12: Find the distortion within a partition of labels

all the other negative examples that were originally provided for the labels (e.g. the negative annotations in the training set for each label). For each dichotomy we have the following two sets of examples:

$$\begin{aligned} S^{+} &= \{x_i : y_i \in L_1\} \\ S^{-} &= \{x_i : y_i \in L_2\} \end{aligned} \tag{4.6}$$

A binary classifier is then trained using S^{+} and S^{-} for each of the N dichotomies.

4.3.1.2 Decoding for Ranked Predictions

Using algorithm in figure 4.11 we have generated unique codewords of size N for each of the labels making up the M matrix, $M \in \{-1, 0, 1\}^{|L|,N}$. Since we need to generate ranked lists we use the scores directly instead of classifier decisions for decoding purposes. Every classifier for a dichotomy $n \in N$ gives us a score $s_n(+1|f)$ for the test frame f which is first normalized. The score for each label is calculated as [176, 178]

$$s(l|f) = \sum_{n=1}^{N} s_n(+1|f) M(l, n) \tag{4.7}$$

so if a label lies in the second partition and is positive for the test example, the negative score of the classifier is multiplied by the -1 value of $M(l, n)$ to make it a positive contribution for that example on that label.

We have also tried to weight the contribution of labels from the classifiers depending on the size of the partition. We believe that the number of labels in a partition affects the performance of a single label due to annotation noise and if the classifier is not strong enough it will not cater the diversity within a partition that comprises of a variety of labels. We thus give high weights to partitions with fewer labels while decoding. This is achieved by

simply dividing the score in equation 4.7 by the size of the respective partition:

$$s(l|f) = \sum_{n=1}^{N} \frac{s_n(+1|f)M(l,n)}{|M(,n) = M(l,n)|} \tag{4.8}$$

4.3.1.3 Ensemble of ECOCs

Equation 4.3 gives us the choice to alter the similarity in favor of visual or semantic criterion and in result create many different partitionings. I.e. the value of λ is changed to generate many ECOC M matrices and this generation is very quick since it uses precomputed similarity information which itself takes only one pass over the training set. Furthermore with each value of λ we can change the distortion criteria using Θ to change the size of the partitions (and also the resulting M matrix). We combine the scores from each of the ECOC and combine it with others to generate ensembles of ECOCs. For combination we have used weighted linear fusion. If $s^k(l|f)$ is the test frame score for the label l from the k'th ECOC then the fusion weights are calculated as follow from the K ECOCs:

$$s(l|f) = \sum_{k=1}^{K} w_l^k * s^k(l|f)$$

to give the final ensemble score $s(l|f)$ for the test frame. This fusion is done separately for each label and fusion the weights are optimized on the training set.

We have built varied combinations of ensembles from different ECOC generations. The value of λ is taken from $\{0, 2.5, 5, 7.5, 1\}$ with $\lambda = 0$ dictating the generation of a *semantic* ECOC and $\lambda = 1$ leading to a *visual* ECOC construction. Θ take on from the following four values which are fixed through experimentation on the training set: $\Theta \in \{1.25, 1.5, 2, 4\}$. We have built 9 ensembles by changing these parameters. First five ensembles have 4 ECOCs each where the value of λ is fixed and Θ is varied. Similarly four more ensembles are generated for fixed λ with 5 ECOCs owing to the changed λ.

4.3.2 Results and Experiments

Now we present results for the proposed ECOC generation method for Multi-label ranked predictions and see the effects of the parameters in various ensembles and the weighting strategy proposed in equation 4.8. We will also look at some of the ECOCs generated to see which criterion generates more intuitive partitions.

4.3.2.1 Datasets and Setup

We have performed experiments on the TRECVID 2010 (TV2010) and TRECVID 2013 (TV2013) [18, 187] dataset.

For the TV2010 containing 400 hours of 11,644 internet videos with 119,685 keyframes

for development and 146,788 test keyframes we generate ranked lists for 50 concepts from the TRECVID 2011 Light Semantic Indexing task [18]. The visual features used are 128 dimensional SIFT features [63] which are densely extracted [186] and are used to build a 500 word visual dictionary using k-means. For classification we have used linear SVM to learn from a suitable feature map (homogeneous kernel map) built by the histogram intersection kernel [137, 138].

For the TV2013 dataset, which engulfs the TRECVID 2010, 2011 and 2012 datasets, consisting of 800 and 600 hours of videos for training and test respectively we extract dense SIFT features and generate a dictionary of 1000 visual words. Considering the huge amount of data the classifier used is a linear SVM trained on the homogeneous kernel map built on the input features where Pegasos training [188] is used to minimize the training time. The development is done using the list of 60 concepts while 38 concepts were evaluated by NIST in the year 2013, for which results are presented.

4.3.2.2 Results

The video concept detection performance is judged by calculating Average Precision (AP) on the first 2000 shots returned for each label (concept) on the test datasets.

ECOC Generations

Since we are interested in looking at the performance of Ensembles of ECOC we do not discuss how each ECOC generation performs. We have generated many different ECOC matrices for different parameters as discussed in the last section. To have an idea of how the labels are partitioned for ECOCs we present two examples in the pages below. We only show here ECOC matrices generated for the TRECVID 2010 database based on the training set.

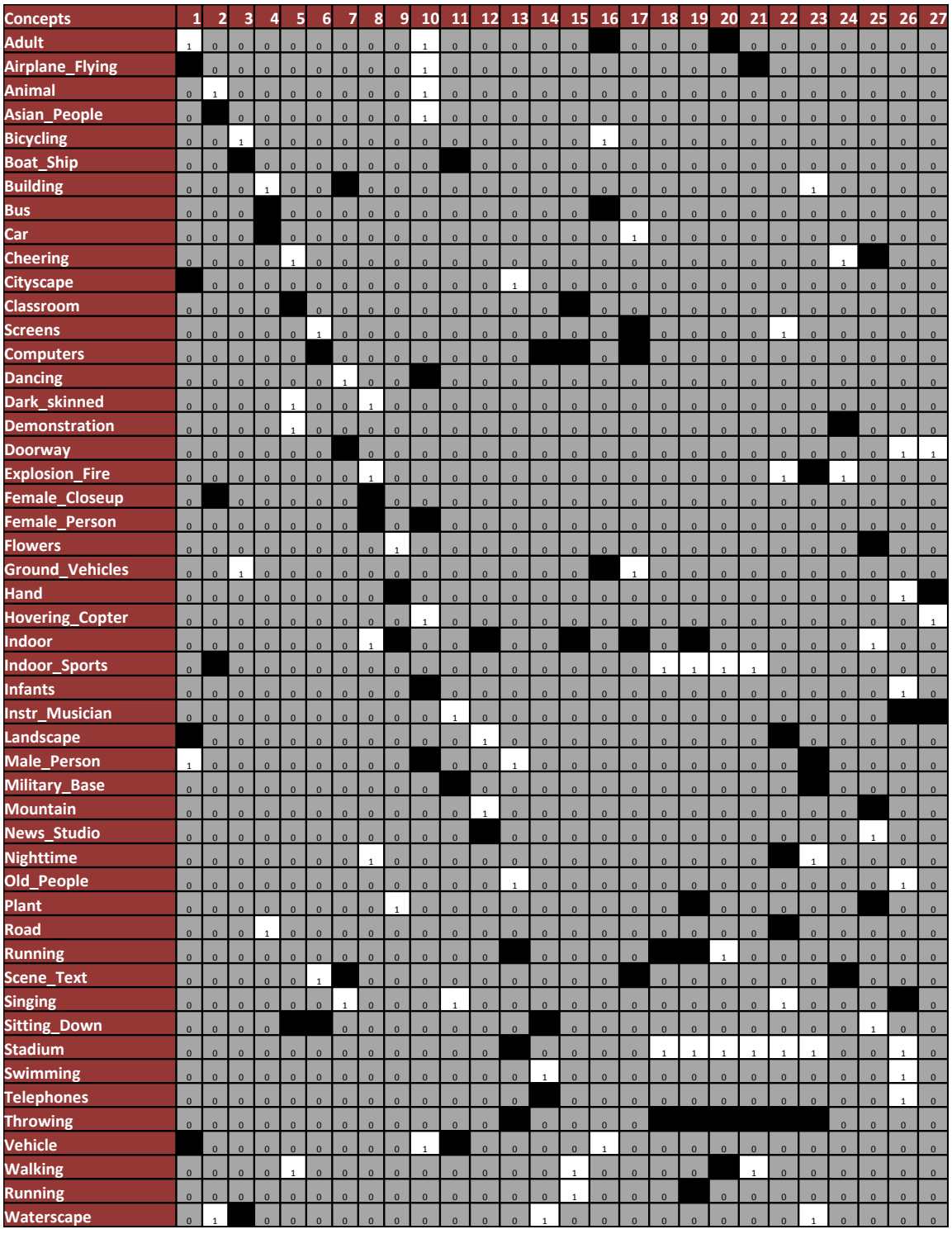


FIGURE 4.13: ECOC matrix generated for TRECVID 2010 dataset with $\lambda = 0$.

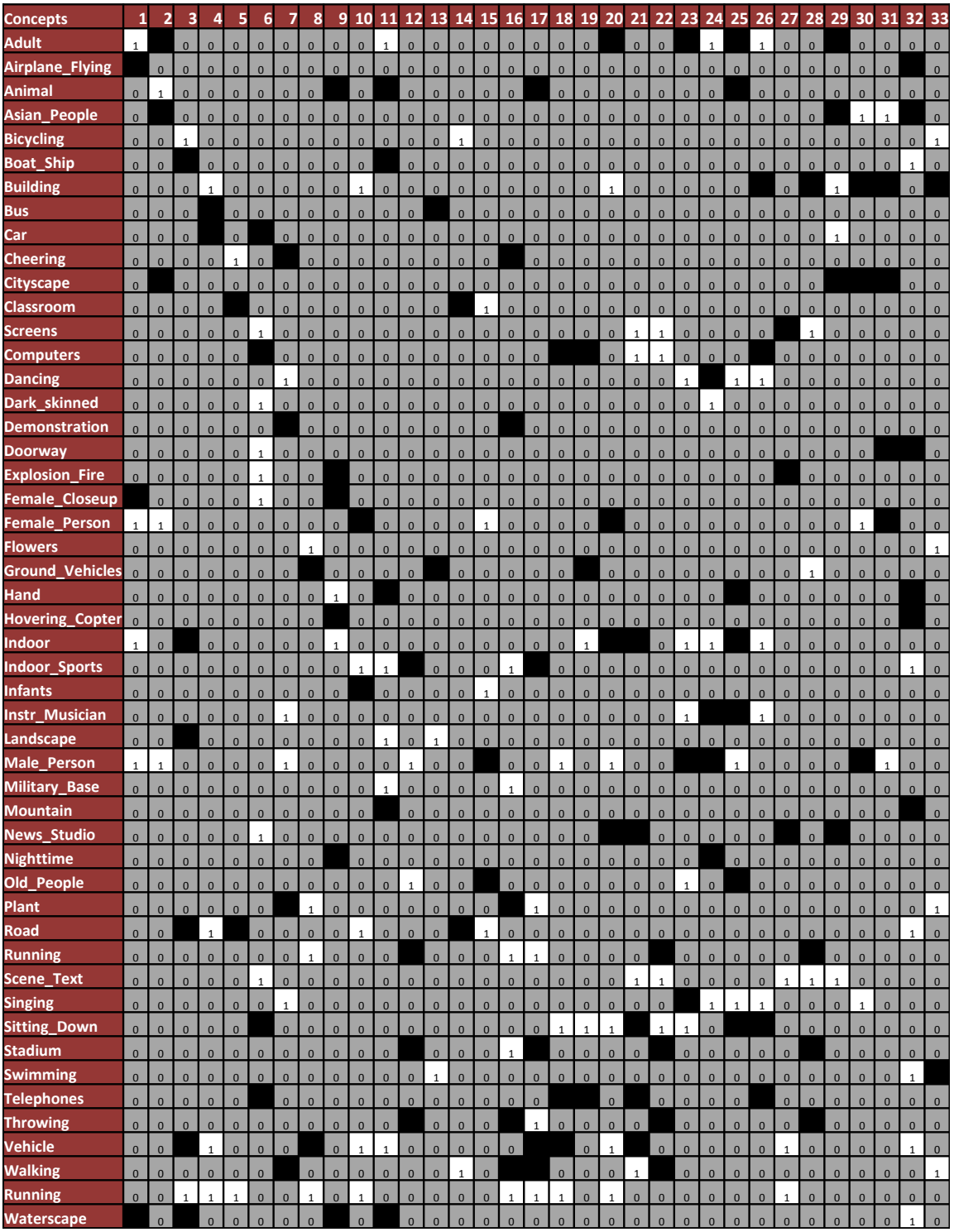


FIGURE 4.14: ECOC matrix generated for TRECVID 2010 dataset with $\lambda = 1$.

We can see that the partitions generated when the criterion was semantic distance $\lambda = 0$, generates groups that are more coherent with intuition. As discussed above with bigger Θ more labels fall into the partitions and thus the number of dichotomies necessary to distinguish the labels are generally less. The performance of the best tree for TRECVID 2010 is a MAP of 2.33% and for TRECVID 2013 is 1.94%.

Method	ENS	ENSw	ENS-OVA	ENSw-OVA
OVA	5.27			
1	4.67	4.68	6.56	6.50
2	4.41	4.34	6.40	6.41
3	3.87	4.10	6.05	6.10
4	3,98	4,29	6,08	6,09
5	4.87	4.91	6.58	6.60
6	4.68	4.87	6.47	6.59
7	4.49	4.63	6.30	6.34
8	4.47	4.62	6.52	6.44
9	3.99	4.60	6.13	6.27
10	**5.08**	**5.22**	**6.61**	**6.67**

TABLE 4.1: MAP scores for various ensembles for TRECVID 2010

Ensembles of ECOCs

We have generated 9 ensembles for the 20 ECOC generations. First 5 ensembles contain 4 trees each where λ is fixed and Θ is changed. Next 4 ensembles are generated for the four values of Θ where λ is fixed. We have also tested another ensemble of ECOC generations where we combine results from 6 best ECOCs based on the performance on the validation set. This 10'th ensemble is generated for both the TRECVID 2010 and 2013 dataset. We present the MAP score for all the concepts in the test set for the 10 ensembles (ENS), table 4.1 for TRECVID 2010 and table 4.2 for the TRECVID 2013 dataset. We also use the weighted decoding which weighs more the concept score from the partitions with fewer total concepts for an ECOC, equation 4.8. Different (ENSw) ensembles are generated for ECOCs with weighted decoding. The baseline is single concept learning or one-vs-all (OVA) classification where each concept is treated as a single labels and a classifier is learned using all the positives of only that concepts. Finally late fusion is done using the ensembles and the single label classification, ENS-OVA and ENSw-OVA.

Method	ENS	ENSw	ENS-OVA	ENSw-OVA
OVA	6.91			
1	**5.17**	**5.38**	7.82	7.87
2	3,74	4,04	7,52	7,66
3	2.69	3.10	7.52	7.91
4	3.39	3.92	7.67	7.66
5	3.58	3.63	7.48	7.51
6	4.35	4.41	**8.06**	8.02
7	3.98	4.21	7.48	7.64
8	3.70	3.75	7.47	7.59
9	4.23	4.43	7.61	7.65
10	4.35	4.39	8.01	**8.10**

TABLE 4.2: MAP scores for various ensembles for TRECVID 2013

Concept-wise Performance

We look at the performance of each concept for the 10'th ensemble which combined 6 best ECOCs based on the validation set. The performance on the test set for TRECVID 2010 and TRECVID 2013 is shown (in log scale) in the figures 4.15 and 4.16 respectively. The concepts are sorted by the number of positive training examples in the development set. For TRECVID 2010 the concept "Helicopter_Hovering" has the lowest count at 18 examples and the concept "Male_Person" has 7,945 on the other extreme. For TRECVID 2013 the count goes from 247 examples for the concept "Bus" to 11,545 for the concept "Animal". We can see that generally concept with fewer training examples benefit more from the shared ECOC classifiers. Moreover ENSw10-OVA largely outperforms the baseline for most of the concepts for both the datasets.

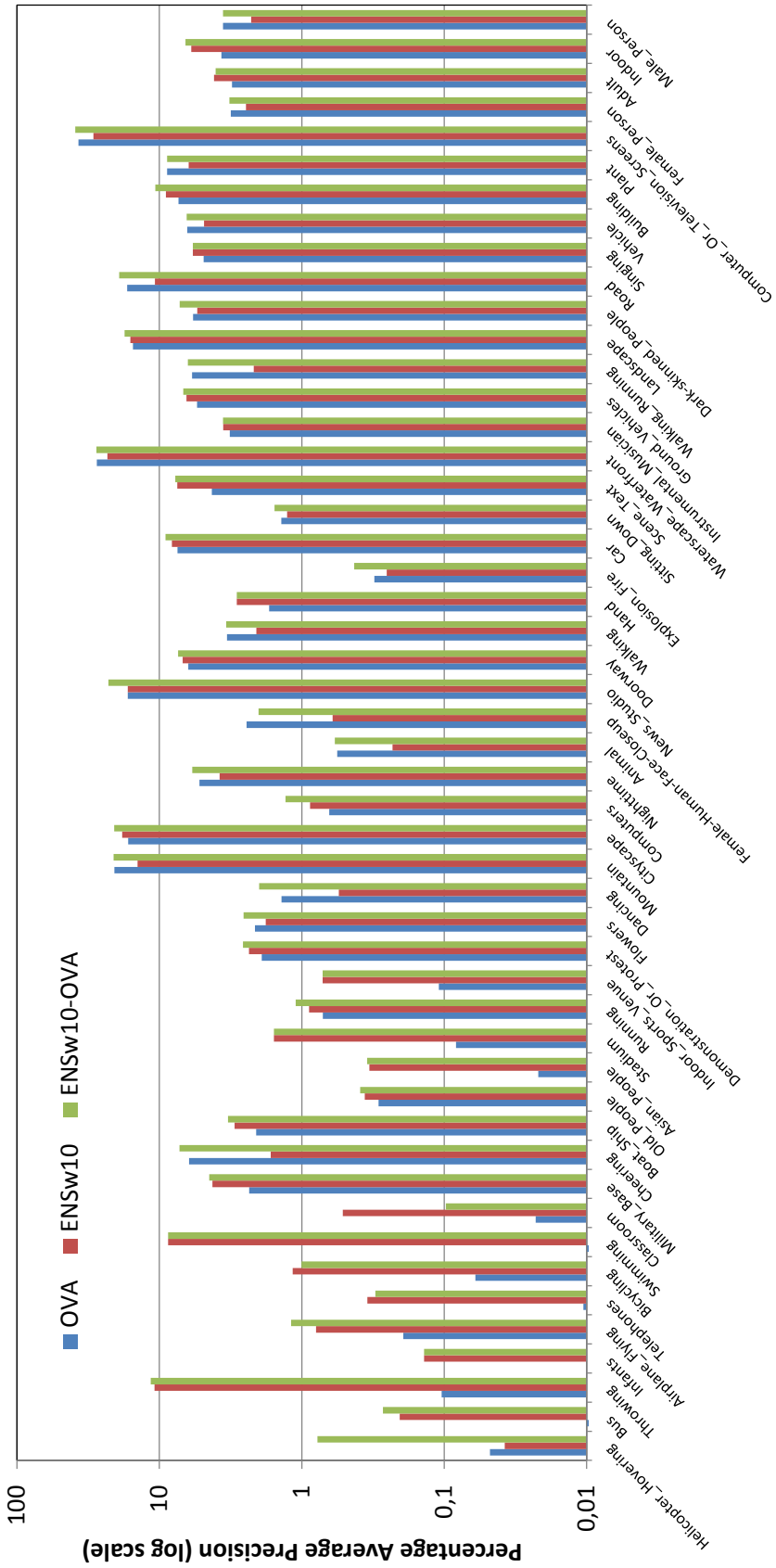


FIGURE 4.15: Average Precision for 50 concepts: Ensemble of 6 ECOCs, TRECVID 2010.

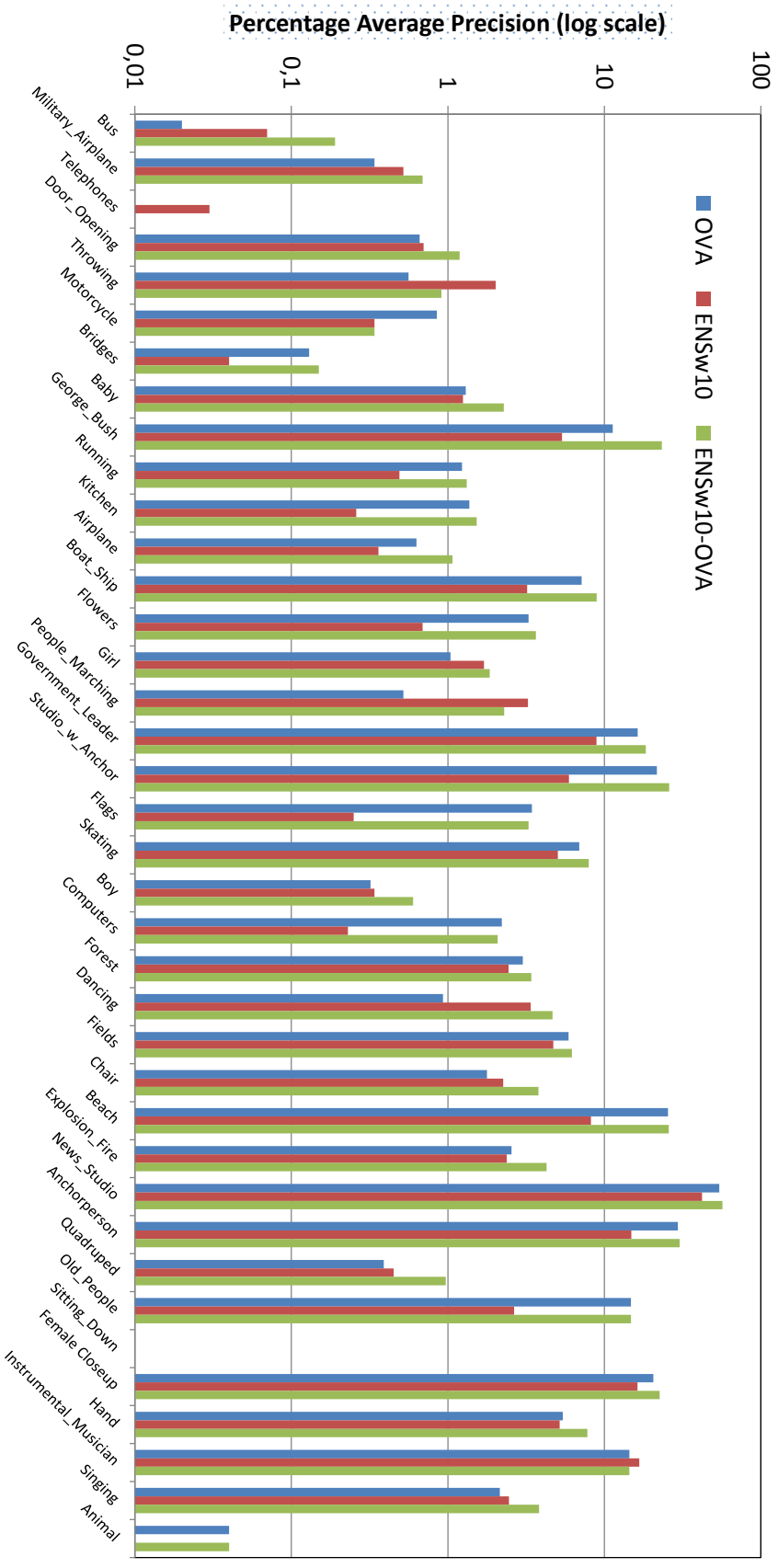


FIGURE 4.16: Average Precision for 38 concepts: Ensemble of 6 ECOCs, TRECVID 2013.

4.4 Label Trees and Repetitive Codes

In this section we continue to work on the 3rd paradigm in the figure 4.3 where we partition concepts into groups using the similarity and the difference between labels simultaneously. The goal is again to overcome sparse annotations problem by increasing the number of positive examples per concept with the number of classifiers per concept, though sub-optimal, augmented too.

We use identical similarity criterion between labels as in the previous section but the division in partitions is done differently. We divide the labels into binary partitions starting from the whole set of labels until each label is differentiated from the others. This induces a binary tree on the labels where each node splits the labels into two subsets and the leaves contain single labels [140]. Based on the similarity criterion the labels that are *similar* are kept together till deep into the tree, until a time comes eventually where they are placed in the opposite sides of the partition. This way each label embodies a unique combination of partitions. Wang and Forsyth [184] also build trees by partitioning the *label space*, as they call it, using a similarity and error criterion on a validation set. Validation set is also employed by [183] to add nodes (dichotomies) to an already generated tree that partitioned the set of labels. We do not use any validation data rather rely directly on out similarity criterion to generate ensembles of trees. For combination we use weighted fusion to generate the final ranked list.

Each node of the binary tree divides the receiving set of labels into two disjoint partitions. We divide the concepts or labels into partitions iteratively, until all the labels are divided. Then we merge all the annotations on either side of the partition and train a single label binary classifier for that partition. In the end we trace back classification score for each label based on appropriate combinations of the scores of the partitions to which that label belongs. The label partitioning is viewed in terms of ECOC framework where a set of simple sub-optimal classifiers can achieve the performance of a good complex one. Then an effective weighting strategy is adopted while decoding to emphasize more on important partitions or dichotomies, generating list of scores for each label. Finally we build groups or ensembles of trees by varying the similarity criterion and fuse the information from trees to get the final classification score. We have performed experiments on the TRECVID 2010 the TRECVID 2013 dataset and we show that the proposed approach is complementary to single label classification by showing significant improvement on the TRECVID Semantic INdexing (SIN) task. We also shed some light on the importance of the number of labels to start the partitioning with and argue that more labels induce a better partitioning and thus improved classification results.

This approach is again different from group based classification, hashing and attribute learning for the similar reasons listed in the previous section, however it does differ from the label clustering based partitioning method. The main difference is that here the number of

partitionings is limited. In label clustering many dichotomies were generated until each label is uniquely assigned to the partitions, but here the number of partitions is always $|L| - 1$, where $|L|$ is total the number of labels as a binary tree is generated. Also the former could include any two labels in a partition based on the current similarity but for the later only the labels in the parent set are partitioned based on the depth of the tree.

Moreover there are partitions that only consist of only one label. This happens at each leaf of the tree which contains only one label in one partition against the other partition that may contain 1 or more labels. Thus each label enjoys being alone in a partition at the leaves of the tree. The similarity criteria and the generation of final scores for each label is done similar to the label clustering method but with a slightly different weighting strategy. Another difference is that the distance is no longer weighted (with repulsion) dynamically.

4.4.1 Proposed Approach

Similarity or distance between labels or concepts can be calculated through different means like the distance between features or ontological graph distance. We build a tree in the *label space*, which is the space represented by the distances between the labels. We again use the visual distance and the semantic dissimilarity based on the common annotations ratio among labels to find this distance between labels. The two measures are used simultaneously to split nodes and construct the label tree. The root node splits the whole label set into two partitions. Each node then splits the receiving partition into two subsets until each leaf contains only one label. So in case of partitioning tree we ensure that each concept is eventually different from the most similar concept.

The partitioning criterion should be designed in a way to increase the *learnability* of the tree while maximizing the number of possible examples that can be assigned to either side of a tree node.

Learnability is the ability of the tree to learn from a particular partitioning of the labels and generalize on a test set [184, 189]. So a tree with poor or counter intuitive partitioning will have low learnability and thus will perform poor on the test set. As an example a tree partitioning the labels {*Car*, *Road*, *Anchorperson*, *News Studio*} into {*Car*, *Road*}, {*Anchorperson*, *News Studio*} is more learnable than a tree splitting the initial set into {*Car*, *News Studio*}, {*Anchorperson*, *Road*}. Since dividing the label space at each node directly influences the learnability of the label tree [184] we use similarity measures that follow intuition by keeping two close labels in the same partitions till deep into the tree.

4.4.1.1 Tree Construction

Similar to the label clustering in the previous section we calculate the distance between visual feature vectors and the positive annotations and combine them to find the total dissimilarity between 2 labels at shown in the equation 4.3.

Input L: a (sub)set of labels. A node t
Output A division of L into *left* and *right* partitions
if $|L| = 2$ **then**
    $L_l \leftarrow l_1$
    $L_r \leftarrow l_2$
else
    $\{i, j\} \leftarrow \underset{i,j}{\operatorname{argmax}}\, d(i, j), i, j \in L$ ▷ Select seeds
    $L_l \leftarrow i$ $L \leftarrow L \backslash L_l$
    $L_r \leftarrow j$ $L \leftarrow L \backslash L_r$
    repeat
        $L_l \leftarrow L_l \cup \operatorname{argmin} d(i, L_l), i \in L$
        $L_r \leftarrow L_r \cup \operatorname{argmin} d(i, L_r), i \in L$
        $L \leftarrow L \backslash L_l$
        $L \leftarrow L \backslash L_r$
    until $L = \emptyset$
end if
$M(t, i) \leftarrow +1, i \in L_l$ ▷ Assign binary codes
$M(t, i) \leftarrow -1, i \in L_r$

FIGURE 4.17: Node Splitting Algorithm

The parameter λ controls the contribution of the visual distance and the semantic dissimilarity. Since there is no repulsion to weight the distance between the labels we use $d(i, j)$ directly to find the similarity between l_i and l_j. The distance between a partition L_k and a single label l_i is the average distance to all the labels actually present in the partition, as:

$$d(i, L_k) = \frac{\sum_{j \in L_k} d(i, j)}{|L_k|} \tag{4.9}$$

where $d(i, j)$ is taken from the equation 4.3.

Figure 4.17 shows the algorithm for splitting one node of the tree which receive a certain set of labels as input. The algorithm simply selects the two farthest labels as seeds and then builds two clusters (left and right) around those seeds. At each iteration the label closest to either cluster is assigned to that cluster until no more label is left to be assigned. Note that the new dissimilarity is calculated between the label and the cluster center but not the initial seed. Two partitions are generated each of which is further passed through the same procedure to complete the tree. Since its a binary tree we call the two partitions *left* and *right*.

For training each node is considered one binary classifier since it partitions the labelset into two sets. All the examples belonging to the labels in the left partition are treated as positive examples and the examples belonging to labels in the right partition are considered negatives. So the positive and negative examples for each classifier are actually only the positive examples of concepts on either side. We borrow notation from [184] to build the two

sets of examples as follows:

$$S^+ = \{x_i : y_i \in L_l\}$$
$$S^- = \{x_i : y_i \in L_r\} \tag{4.10}$$

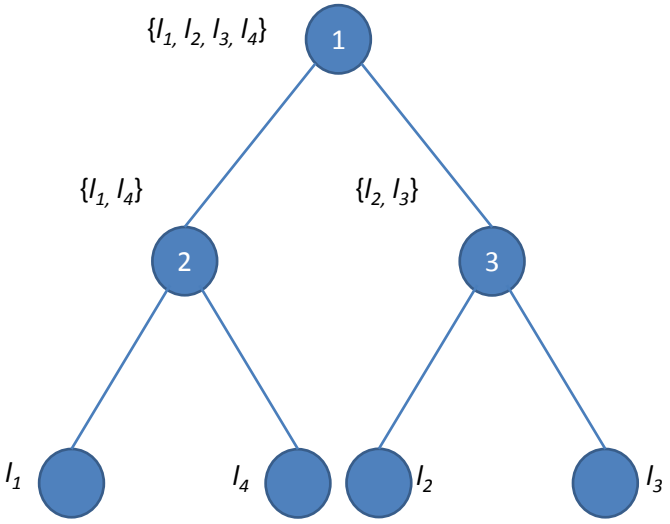
S^+ and S^- are used to train the binary SVM classifiers. Since we are dealing with mutli-label data as opposed to multi-class data, in case of contention the example is assigned to the partition containing the minority class. That is to say if an example belongs to more than one labels, and the two labels are divided by a node, the example will belong to the node with fewer total examples. This makes sense as in case of a multi-label example belonging to a general class, *Vehicle* and a specific class, *Car* the example should be for *Car* since it differentiates *Car* from the other *Vehicle*s. For the other case of simple majority vs. minority it is better not to reduce the already limited number of examples for the minority class. This kind of contention normally occurs near the leaves of the tree since labels close to each other are kept in the same partitions until it is necessary to separate them to tell them apart.

4.4.1.2 Ternary Codes and Error Correction

The tree construction bears close resemblance to the multi-class classification approach Error Correcting Output Codes (ECOC) [174, 176, 178, 180]. To look at the benefits of the repetitive codes for classification we first explain the resemblance. ECOC builds a set of sub-optimal binary classifiers for each label where final prediction is obtained by simple pooling of the participating classifiers for that label. Each classifier epitomizes a dichotomy which partitions the label space just like a node of the tree built in the previous subsection.

The technique presented here is essentially ternary ECOC where a label can be either part of a dichotomy (-1,+1) or not (0). Note that the tree is binary as each node splits the labels it receives from its parent node into two partitions; left and right, which are +1 and -1 for the ECOC. All the other labels that are not received (and hence not split) by that node are 0 in the ECOC framework. The nodes or dichotomizers are built incrementally optimizing criterion in equation 4.3 to divide most profiting class labels. This code is also called the codeword for that category and the codewords for all the categories make up the rows of the ECOC M matrix, where $M \in \{-1, 0, 1\}^{|L|,N}$. $|L|$ is the number of labels and N is the length of each codeword or number of classifiers. Here this also represents the number of nodes of the label tree. Each class label encodes a unique code and the (hamming) distance between two classes that are close is low. Figure 4.18 depicts a multi-label tree and the corresponding M matrix [183].

As we are generating ranked lists for test frames again, every classifier $n \in N$ gives us a score $s_n(+1|f)$ for the test frame f. The score for each label is calculated as done in the equation 4.7. Since the dichotomies turn from general to more discriminative as the depth of the tree increases we have found that more weight assigned to the specific dichotomies



	n_1	n_2	n_3
l_1	1	1	0
l_2	-1	0	1
l_3	-1	0	-1
l_4	1	-1	0

FIGURE 4.18: A binary tree which partitions the label-set and the corresponding M-matrix with 3 dichotomies. Each node of the tree represents a column of the M-matrix

deeper in the tree results in better performance. So instead of the weighting based on the partition size used in the previous section for the label clustering approach we have used the following,

$$s(l|f) = \sum_{n=1}^{N} \frac{n}{|M(l,.)|} s_n(+1|f) M(l,n) \tag{4.11}$$

with $s(l|f)$ finally normalized for a tree. $|M(l,.)|$ is the number of classifiers for label l. Each leaf of the label tree is a single label so $s(l|f)$ can also be regarded as the score at leaf corresponding to label l.

The first and foremost advantage of any technique augmenting labels is the increase in the number of examples per category which affects significantly performance of the categories with fewer positive examples. In ECOC this advantage is not limited to just the increase in training resources per category but also increases the error correcting capabilities of the system due to repeated classifiers [176, 182, 183]. Even if some dichotomies partition badly the label space others are expected to correct the mistakes due to repetition and provide reliable final results. This advantage is further strengthened when we use ensemble of trees for a single problem as described in the next subsection.

For good error correction we argue that the label set should be dense in relation with the number of examples. Since we have established that related labels add examples for each other, a dense multi-label dataset induces a tree which in which partitions are more learnable. Thus if the label set is very sparse, unrelated labels are always forced to group together in partitions, and we will have fewer training examples per dichotomy and a poor per label classification in the end. We support these claims with experimentation later in

the section 4.4.2.

4.4.1.3 Ensemble of Trees, Forest

The label partitioning tree uses pre-computed information and is quick to build. We take the liberty to build multiple trees by varying our similarity criterion. This is done by changing the value of λ in equation 4.3. The trees can be combined by averaging the scores of the corresponding leaves, choosing the highest score [184] or finding a weighted combination of the corresponding outputs from the trees. For label (leaf) l the weighted combination from two trees t_1 and t_2 is learned as follows on the validation data:

$$s(l|f) = w_l^{t_1} * s^{t_1}(l|f) + w_l^{t_2} * s^{t_2}(l|f)$$

We have built ensembles of upto 6 trees using linear weighted fusion of trees. For each ensemble the parameter λ in equation 4.3 is varied from 0 to 1 uniformly. E.g. for an ensemble of 3 trees λ is 0, 0.5 and 1 for each of them.

4.4.2 Results and Experiments

4.4.2.1 Datasets and Setup

We have performed experiments on the TRECVID 2010 (TV2010) and TRECVID 2013 (TV2013) [18, 187] dataset with exactly the similar setup as in the previous section.

4.4.2.2 Results

The video concept detection performance is judged by calculating Average Precision (AP) on the first 2000 shots returned for each label (concept) on the test datasets.

Overall Results

Table 4.3 shows the Mean AP (MAP) scores of the proposed label forest (LF) technique and compares them to the one-vs-all (OVA) classification, groupsvm (GS), which is presented in the section 4.2 and the ECOC ensembles technique (ENSw) presented in 4.3. We take the best result from tables 4.1 and 4.2 for the ECOC based method where weighted decoding is used and an ensemble of 6 ECOC matrices is employed to generate the ranked lists. For the LF the scores from individual trees are maximized (LFM) for each concept as in [184], averaged (LFA) and fused with optimal weights (LFO). Trees are also build by partitioning the labels randomly dropping the visual and semantic closeness criterion, which are then fused by finding optimal weights (LFR). Furthermore LFO, GS and ENSw are fused with OVA using linear weighted fusion which further improves performance. For table 4.3 all the forests used contain 6 trees and for GS the best results are shown containing 100 groups for 50 concepts. Results on GS for TV2013 are not available.

Methods	TV 2010	TV 2013
OVA	5.27	6.91
GS	5.37	-
ENSw	5.22	4.39
LFM [184]	4.52	3.45
LFA	5.28	4.72
LFR	5.14	4.52
LFO	**6.29**	5.13
GS-OVA	6.04	-
ENSw-OVA	6.67	8.10
LFO-OVA	**7.05**	**8.38**

TABLE 4.3: MAP scores for TRECVID 2010 and 2013

Examples per Classifier

Using binary dichotomies to train classifiers the number of labeled examples are considerably reduced when compared to the one-vs-all approach since the negative examples provided with the dataset are discarded and only positive examples are used for training. Table 4.4 shows the average number of examples per classifier for the two datasets, rounded to the closest hundred, acquired from the development sets using all concepts (50 for TV2010 and 60 for TV2013). This removal of negatives comes with a cost which is visible in the performance of LFO for TV2013 where each classifier has almost 10 times few examples.

However if we look in terms of number positive examples per label there is a considerable increase for both datasets. Since all the examples on either side of the dichotomy for the label tree are actually positive examples the numbers in the third column of the table 4.4 are positive examples for our approach. The increase is manifold from the original number of positives annotations per category on average. For the almost quadrupled TV2013 the number of labels is lower with respect to the dataset size so the label partitioning does not entirely capture the relationships between concepts as some counter intuitive partitions may be formed.

	OVA (pos+neg)	OVA (pos)	One-Tree
TV2010	45,000	1,600	9,500
TV2013	78,000	1,700	9,100

TABLE 4.4: Average number of examples per classifier

Reducing the set of labels

To further make our point we have performed an experiment with rather smaller label sets on the similar datasets. That is to say we make label partitioning trees for fewer labels. To achieve that we divided the set of 50 concepts randomly into 5 sets of 10 concepts each for TV2010 and then repeated the similar experiments. Similarly for TV2013 the 60 concepts set was divided into 6 sets of 10 and trees were generated separately for each of the 6 sets of labels. For both the datasets for every set of 10 concepts we generated 6 trees making

one ensemble, finally making 5 ensembles for TV2010 and 6 ensembles for TV2013. Final classification scores were calculated separately for each ensemble and in the end we have AP scores for 50 and (38 evaluated out of 60) concepts for TV2010 and TV2013. Table 4.5 shows the MAP for the two datasets with divided label sets, which is considerably less than the MAP acquired using the full set of labels for a 6-tree ensemble, table 4.3. The number of total examples on average per classifier is also reduced, as understandable, to 6,400 for TV 2010 and 5,700 for the TV2013 dataset. Thus the performance of the label forests approach is critical to the number of labels used and increases with the increase in the label-set size.

	LFO	**LFO-OVA**
TV2010	5.21	6.00
TV2013	3.12	7.01

TABLE 4.5: MAP scores for all concepts using subsets of concepts for tree generation

Concepts with Few Positives

To see the effect of concept detection performance on the concepts that have lesser positive annotations we select the 10 concepts from each dataset with fewest positive examples. We compare performances of LFO with 6 trees and fusion of LFO and OVA and plot the performances on the log linear graphs in figures 4.19 and 4.20.

Figure 4.19 shows the AP scores for 10 concepts from TV2010 dataset with an average 82 positive examples per concept in the development set. All the concepts benefit from the addition of labels, even those that had an AP of 0 acquired with OVA. Fusing OVA and

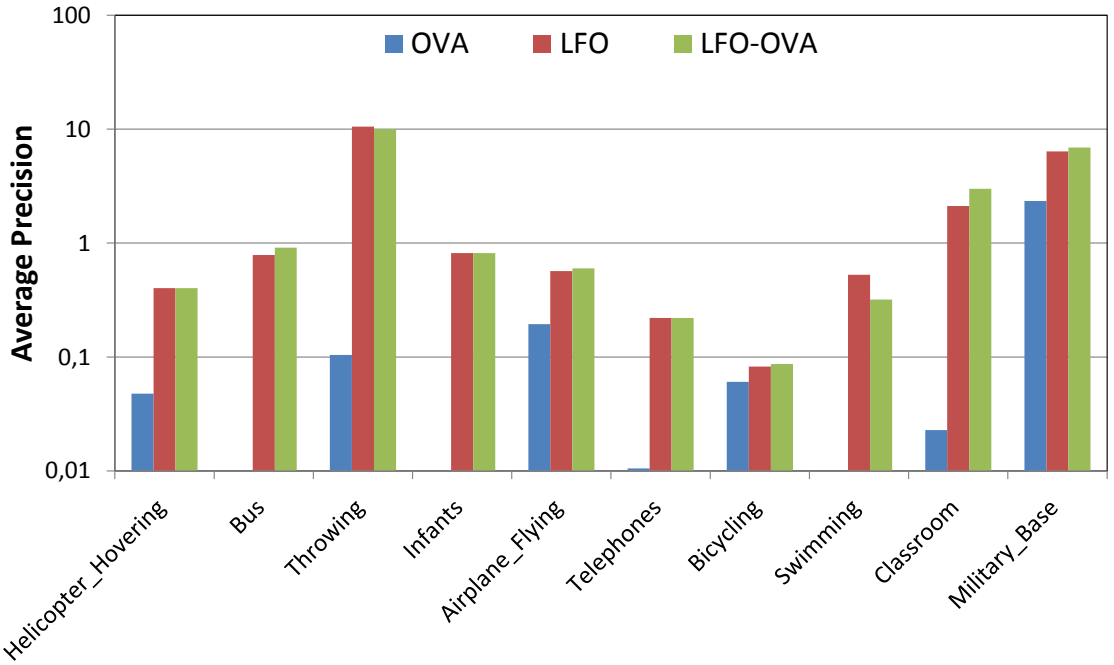


FIGURE 4.19: AP scores for 10 concepts with fewest positive annotations in TV2010

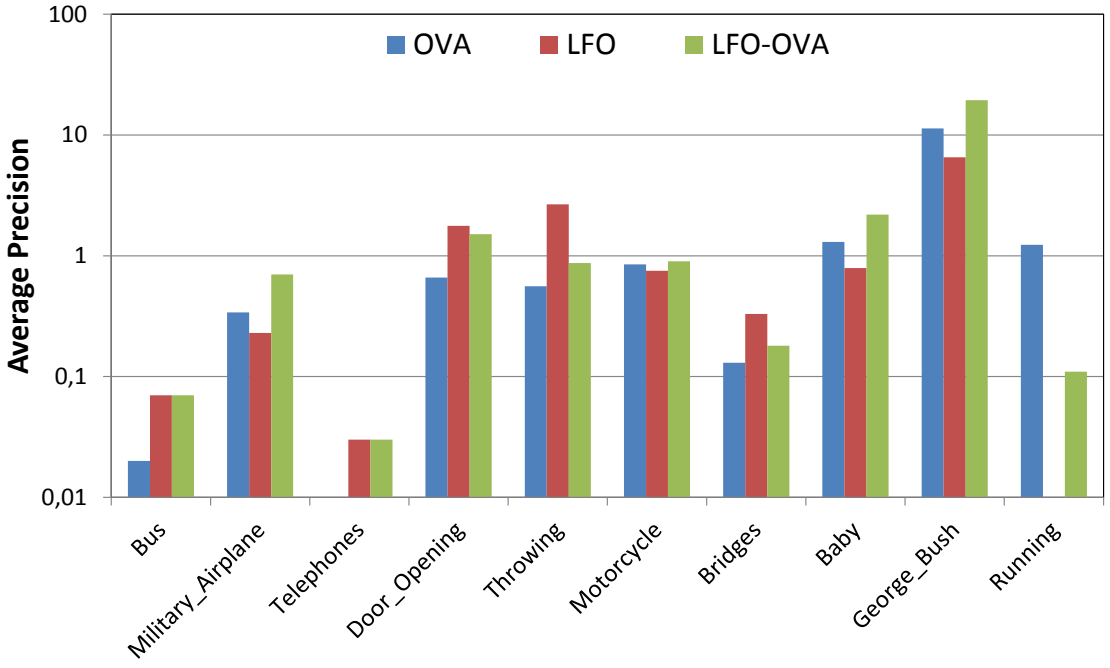


FIGURE 4.20: AP scores for 10 concepts with fewest positive annotations in TV2013

LFO further improves performance except for "Throwing" and "Swimming" with a negligible drop.

For TV2013 the number of positive examples for the 10 concepts with fewest examples is 610 on average. Results are shown in figure 4.20 and here LFO is not as effective but the fusion tries to recover some of the loss. However LFO does bring more shots in the first 2000 for the concepts "Telephones" and "Bus" for which there was none or only a few previously with OVA.

Groups of Trees

Figures 4.21 and 4.22 show performance of various groups of trees for both the datasets. We build forests of 2 to 6 trees using the proposed method which is compared to random label partitioning for the same number of trees. Both methods are then fused with OVA for each group of trees. Since the tree generation using the algorithm in figure 4.17 generates balanced tree, the random tree generation is also done in a balanced way for a fair comparison. The depth of every tree is similar and so is the number of SVMs. The proposed method always outperforms random partitioning for single classification and also for fusion with OVA. The black box in both the figures represents MAP score for OVA. For TV2010, figure 4.21, every forest performs better than OVA and when fused the performance increases by around 30% for each forest. LF approaches do not perform as good as OVA for TV2013, 4.22, due to reasons listed earlier but again the fusion results in around 20% increase for every forest. The improvements over the baseline are verified in their significance by randomization testing [190] for the proposed approach, for both the datasets.

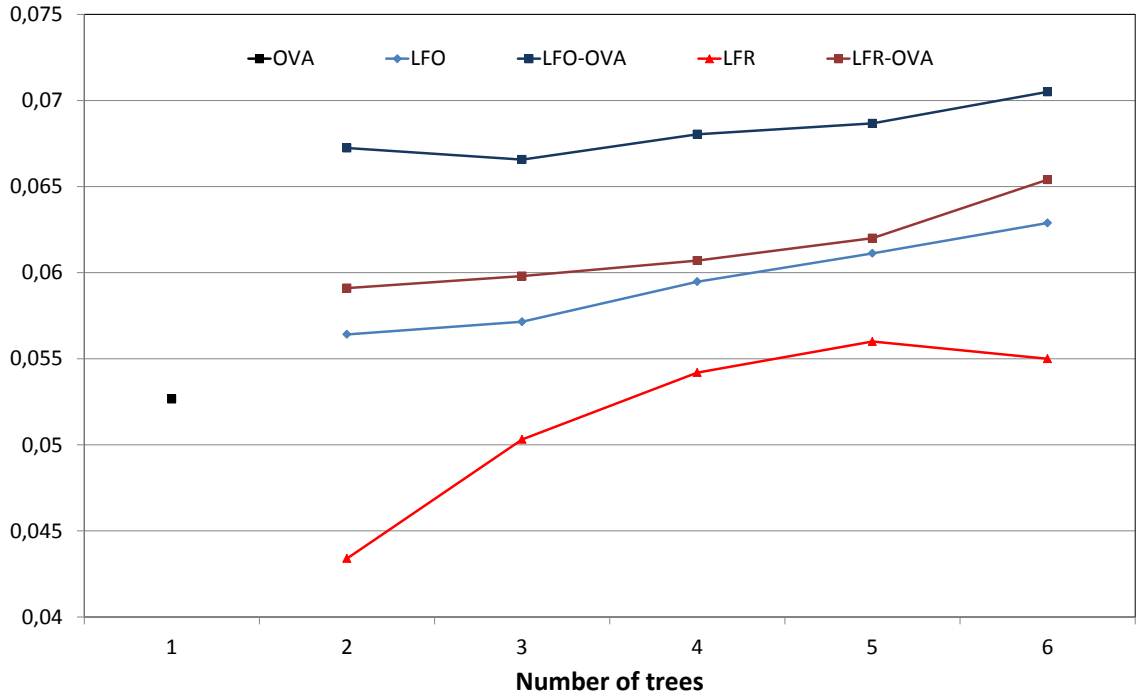


FIGURE 4.21: TRECVID 2010 (50 concepts). Performance (MAP) comparison of proposed label partitioning to random partitioning and single label classification, for various groups of trees.

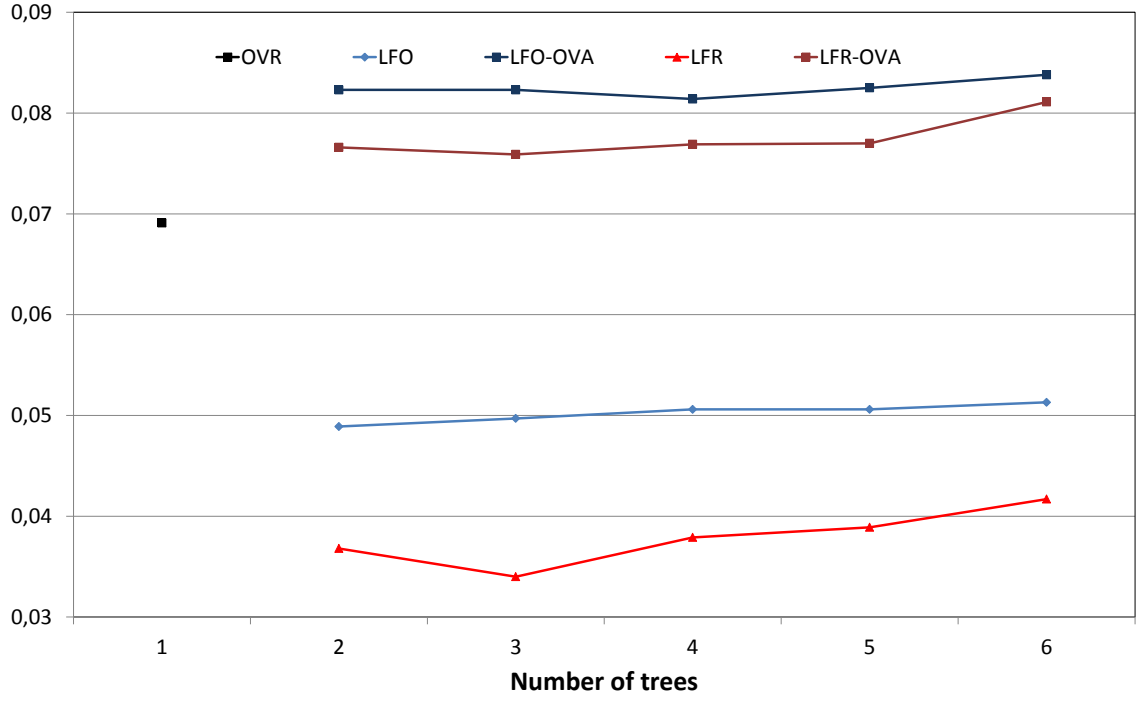


FIGURE 4.22: TREVID 2013 (38 concepts). Performance (MAP) comparison of proposed label partitioning to random partitioning and single label classification, for various groups of trees.

4.5 Conclusions

In this chapter we have seen how explicit label sharing can benefit the video concept detection, particularly for the concepts with very few positive examples.

We have devised a quick way of grouping concepts together based on their visual similarity to train them together for concept detection in internet videos. The group making criterion is intuitive, very fast and each concept is represented by a unique combination of groups. With the introduction of a little useful even random information we are able to improve concept detection performance on the TRECVID 2010 datasets for the list of 50 concepts. We improve 12% and 15% over the baseline with only 10 and 20 new group classifiers formed on the clever criterion. Using random grouping we further improve but at a cost of training a total of twice as many classifiers as compared to the intelligent criterion.

We feel that this group based classification can help ultimately reduce the number of classifiers to be trained if effective combination techniques can be found. So far for the grouping of concepts we have only used visual similarity while there are other options that may be fruitful to explore. Grouping only on visual similarity results in overfitting as the number of groups increases. We would also like to add diverse information in the group or create groups from negative information. This may be achieved with using mutual information principles on negative and positive annotations for concepts. This information is inherently provided with TRECVID style multi-label datasets.

We then grouped concepts in partitions based on the similarities as well as differences from others, and proposed two techniques to achieve this partitioning. The proposed label partitioning methods use effective measures of similarities generating meaningful partitions. This increases the learnability of the partitions which directly affects the performance. The techniques are complementary to single label classification, especially the tree-based method improves performance significantly with as little as only two trees in the forest when the two are fused. The error correcting capabilities of the tree increase as more labels are available.

During iterative tree generation to create a forest a part of validation data can be used to weight similarities between labels. Furthermore like [183] more nodes can be added to the already generated tree to further classify certain confusing labels. Since a label tree essentially results in unique codes for each category eventually other methods could be used to generate such unique codes.

Chapter 5

Leveraging from Unlabeled Data

In this thesis we are striving to improve video concept detection and we aspire to get a boost in performance by tapping into the vast resources of unlabeled data [109], as would be the case for any type of classification. Till this point we have only considered the annotated examples as positive and negative instances for a concept. We have also tried with borrowing examples from other concepts that are related or similar and receding from concepts that are different, but in those cases the examples from other concepts were annotated too. As stressed throughout the thesis there is always need for more annotated data to attempt to capture the variations with which a semantic concept occurs. Specifically for the TRECVID datasets we have used there is always a shortage of positive examples for most of the concepts, a good number of which is critical for the concept detection performance. Since the corpus in question is partly annotated we can always try to find useful examples to add to the training set for a concept for a better classification on the test set.

5.1 Semi-supervised Learning for Retrieval

Semi-supervised classification benefits from large number of unlabeled data where the tedious task of human intervention to annotate the data is minimized. These methods start with only a few labeled examples where the unlabeled data can be annotated iteratively to augment training resources. Learning from partially labeled data is gaining importance as unlabeled data is cheaply available and has proven to decrease classification error significantly [105, 106, 108, 191, 192].

A reasonable method is to find new training instances is to train on the provided training set and use this trained model to add the most confident predicted examples from the unlabeled set to the new training set and train the model again. This *selftraining* loop can be repeated until the performance on a validation set increases. Although it has worked before [105], the problem with this form of incremental learning is the potential lack of variability in the new examples added. The most confident examples added to the pool of positive instances have very little affect on the new decision boundary. Furthermore it may improve

on the validation set but this could lead to over-fitting as no diverse information is added. Dealing with videos give us the liberty to have a variety of representations and for a single keyframe we can extract a number of different features. This give us the opportunity to go beyond self learning and use the diverse information to find new labeled examples for the classifiers with different descriptors, like in cotraining.

Cotraining is a special case of using unlabeled examples which is useful when different feature representations of the data are available [191]. As the name suggests two learners trained on each data representation are used in unison to label training data for each other. This is done iteratively until training error is sufficiently reduced or all the data has been labeled. For final prediction both the classifiers are pooled together. It's a better way to find the new labels as diverse information is used from the representations and both the learners become powerful with time. There are however certain conditions that guarantee the successful working of the cotraining algorithm. Specifically it works when the different representations of data are different views and satisfy the cotraining properties. They should be conditionally independent given the class and each of them should be sufficient to learn, i.e. a learner on each view should predict the true class labels for the most part [191].

This comes naturally for some datasets where the features used are in fact complementary to each other and in combination reduce the classification error, however for others these conditions might be hard to satisfy. The text in the web pages and links to those pages are good examples of complementary features [191]. Another good example is multi-modal features where each feature represents a certain modality in learning from multimedia data. We benefit from the situation where more than two views of the data are available and we have the liberty of choosing the best complementary view for each classifier.

Cotraining provides convergence guarantees provided the two views are sufficient and conditionally independent which is not always the case for video data (specifically visual features). Video concept detection is a hard task and the current classifiers (views) are not sufficient [106, 107, 193–196], i.e. the top ranked predictions contain an abundance of miss-classifications. This leads to noisy examples added to the training set and the classifiers deteriorate over time. This deterioration increases if the size of unlabeled set is large. No matter how diverse the views are, if one view always performs poorly it will adversely affect the overall performance. It is therefore important to select the best view and to select the best examples from the most confident ones for cotraining. A human annotator can be added in the cotraining loop to keep a check on some of the new examples to be added, but this shifts the paradigm towards active learning. The annotator can only be presented with a few most ambiguous examples but here we are trying to minimize (rather eliminate) the human effort, while trying to find the best new labeled examples.

We briefly overview some of the work in the literature that address this problem and offer solutions to avoid adding too much noise during cotraining.

5.1.1 Cotraining for Video Analysis: A review

Yan and Naphade [197] achieve improvement in video concept detection performance by using manual human effort to select from the most confident predictions for each cotraining view. In another work [106] they build classifiers separately on the newly labeled data and include them in the final prediction only if it performs well on the validation set. This restrains the learners to learn from noisy data and stops the cotraining when the predictions worsen on the validation set.

Li et al. [107] validate the examples from different predictors before adding them to the training set for the learners. This is in fact multi-view learning in which predictions on unlabeled examples generate label candidate sets. These label sets are used in a Multiple Instance Learning (MIL) framework which identifies the best label for each example while training. They present three proposals to augment the label sets for each view improving upon each other. The first one simple uses the latest view predictions from each view while the second method uses predictions from all the previous iterations. Finally they add to the candidate label set by using classifiers with different biases for each view and adding predictions from all the different values of the bias. Authors in [198] also perform multi-view learning and select the new examples by an uncertainty criterion based on KL divergence. Furthermore the new examples are added with a weight that is found using a methodology measuring confidence of the examples.

Du et al. [108] identify feature splits for cotraining that satisfy the cotraining independence and sufficiency properties for tire wear classification from images. They use a set of 14 features with simple descriptors like mean and minimum values to more complex ones like DCT coefficients and KL divergence on features. They use clustering based on mutual information to identify pairs of features for cotraining and the confidence of other classifier to select unlabeled samples for the current one. [199] improve web page classification by introducing *Local Cotraining*. In this form of learning different local models are trained by dividing the instance space into partitions. This not only reduces the complexity of training by dividing and conquering but also gives the possibility of retraining only the dominant local models. Moreover they select different amount of unlabeled examples for each view.

Yasan and Cataltepe [200] propose methods for feature subspace selection and feature selection for cotraining. They use mutual information between class labels and features as a relevance measure to select the most relevant subspace and order the features in relevance to class labels. Li et al. [201] perform feature selection for cotraining (FESCOT) by discarding the most irrelevant ones using classification accuracy on validation dataset. Features are iteratively disregarded whereas we select one complementary feature at each iteration. Zhang et al. [195] find weights for Co-SVM [202] classifiers using error on the validation set.

[203] proposes to validate the cotraining views to see if single view learning is enough or does cotraining really bring improvement. Since the views are not always uncorrelated and

compatible cotraining may lead to disastrous results if one view is weak and always feeds in wrong predictions. They use simple statistics like the number of training errors made by the view classifiers, the number of correct common predictions and classifier complexities to identify the unnecessary view. Guz at al. [204] add newly labeled examples to the training set of classifiers based on two strategies: *Agreement*, where only the high confident examples from both the classifiers are considered and *Disagreement*, where examples with high confidence scores on the other view and low scores on the target view are added to the training set of the target view. Disagreement outperforms agreement based approach for the sentence boundary classification problem.

5.2 Selective Multi Co-training

All this research points to one direction: find the best samples from unlabeled examples and add them to the classifier to increase classification performance iteratively. Our proposals add useful information to the annotation set of a classifier based on the cotraining principle to augment the training resources [109]. This is achieved by adding positive examples that are expected (or ensure to some degree) to improve the final classification result. We propose two methods to select the most complementary view among a certain available possibilities based on some statistics calculated on the validation set. We take inspiration from [205] to add the most relevant examples that were the most confusing for the classifier. Instead of negatives we focus on adding only positive examples since we already have a large number of negative examples available. Among other possibilities a relevant positive example for a classifier is the one which was previously misclassified by the same classifier. We adapt this setting in the cotraining framework where the most relevant positive examples are added to the annotation set of the target classifier based on the predictions from another classifier. This will tame the target classifier especially for the categories with few positive examples and huge number of unlabeled examples available. The classifier which identifies the largest number of misclassifications of the target learner is considered the most complementary and this information is the basis for our two criteria presented later. One of the criteria proposed also selects automatically the number of new annotations to be added.

5.2.1 Proposed Approach

Contrary to cotraining where we start with two views of the data distribution assuming that we have very little data to start the training with, here we have a good number of negative examples to start with. The number of positive examples is however very low and is in the orders of magnitudes less than the number of negative samples for each concept. All the views contain descriptors that are classifiers built on powerful visual descriptions. It is important to stress here that the difference in views is only in terms of the descriptor used. The classification method is the same for all descriptors. So we try to add more annotations

to the positive examples of a view using information from another view which is a classifier on a different visual descriptor. This other descriptor is selected out of the available ones which is expected to bring the most information to the target view. The information brought is new positive annotations which are then used to re-train the classifier using the target descriptor.

5.2.1.1 Selecting the Most Complementary View

A certain number of descriptors are available to start with. A classifier is trained on a particular descriptor which makes one *view*. A view thus classifies or labels a video frame, by assigning a score value to the frame. Our development data is divided into training and validation part. The final predictions are done on the test part which is independent of the development set.

Since we are using a validation set to find the most complementary view a good technique would be to find the view with minimum correlation on the ranked validation set. This will pick the view that is the most complementary to the targeted one, but this would be a good idea in the case where the initial classifiers are strong enough in terms of prediction accuracy. In our case of video concept detection where we lag behind in terms of performance compared to other vision tasks, simply selecting the most complementary view would not do. We thus need to find the view that has minimum correlation to the targeted view and also identifies the mistakes (misclassifications) made by the targeted one. Two selection methods following these guidelines are presented later in the section.

As only the descriptor changes in a view, we call our views D. For the target view D_i we try to find the one that brings the most valuable information in terms of annotations, figure 5.1. This is judged by the classification performance of the descriptors on the validation set. In the next iteration of the cotraining the training and validation sets are re-labeled for each descriptor and thus we start over the selection process for every descriptor with the newly obtained data. We call this process as Selective Multi Cotraining. It is important to mention here that in this selective multi cotraining features do not work in pairs, rather for each feature the most complementary is selected at each iteration.

5.2.1.2 Selection Methods

We present two methods to select the source view which is used to add annotations to the target classifier. We add k new positive annotations to the target classifier's training set using the selected view. For the first method k is fixed while for the next one k is found automatically per descriptor. The two methods are detailed in the next two subsections.

Positive Disagreement

To explain this selection method let's look at the left side of the figure 5.2. Suppose we want to add k annotations to the examples for the view D_1 and we have an initial ranking of the validation set for all the views. The upper part of figure 5.2 contains validation ranked

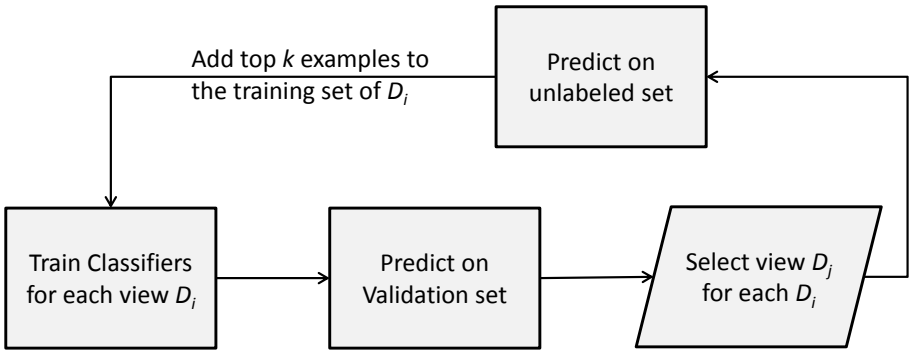


FIGURE 5.1: Selective Multi Cotraining: Identifying the best view at each iteration for *retraining* from a choice of views

lists for all the descriptors, V_D. The numbers represent the ids of the shots and the colors indicate the actual labels where green, red and black mean positive, negative and unlabeled respectively. Each shot S_i in the list has a rank; $rank^{V_D}(S_i)$ which is an integer value. The rank for the most confident prediction is 1 which is at the top of the ranked list. As the rank increases the prediction confidence decreases. So if a true positive example is predicted to have low confidence score for the concept, it would be ranked high somewhere at the bottom of the list and is treated as a misclassification. We define a function f returning 1 when the shot S_i is annotated positive.

$$f(S_i) = \begin{cases} 1 & sign(S_i) = + \\ 0 & otherwise \end{cases} \tag{5.1}$$

To select the most complementary view we look at the first k shots in the validation set of the target descriptor's classifier and the first k shots provided by the other views in a pairwise manner. Since we have the labels of the shots on the validation set we can calculate the disagreement between the pair: source D_s and the target D_t for a fixed value of k as:

$$DIS_{ts} = \sum_{i:rank^{V_{D_s}}(S_i)<k \,\wedge\, rank^{V_{D_t}}(S_i)>k} f(S_i) \tag{5.2}$$

The disagreement counts for the source view D_s how many labels are positive that did not appear in the first k shots for the validation shot list of the target view D_t. This is essentially the information that is missed by D_t for the top k shots. We select the D_s with maximum DIS_{ts} as it is understandable that the predictions done by this descriptor are the most complementary to D_t. For figure 5.2 if k is fixed as 3, then $DIS_{12} = 0$, $DIS_{13} = 1$ and $DIS_{14} = 0$ and D_3 is selected to add new labels for D_1 in the next iteration.

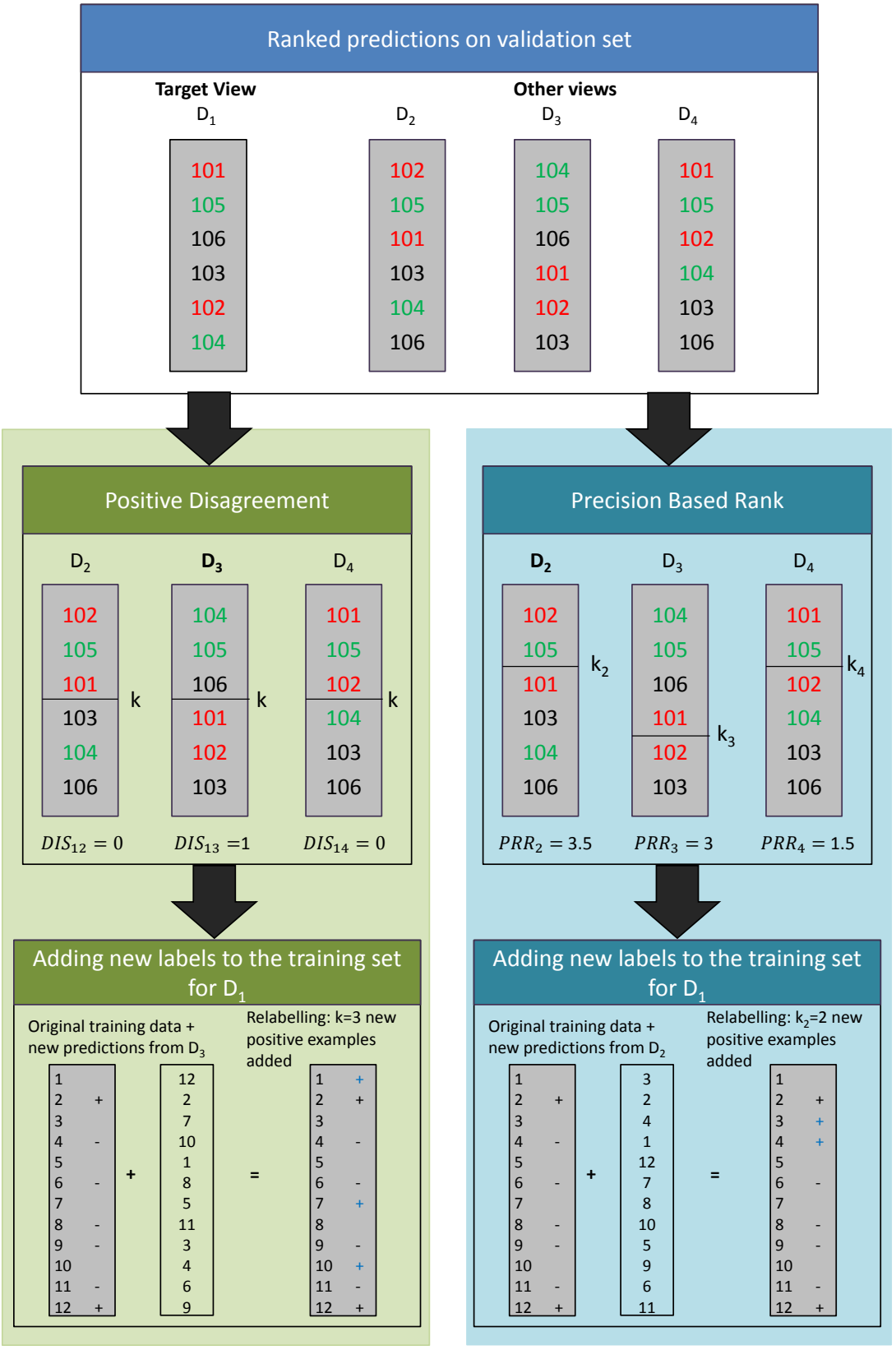


FIGURE 5.2: Selecting source view for Selective Multi Cotraining methods and adding new examples to the training set.

This view selection is done separately for each concept. That is to say that for each concept for D_t the best complementary view is chosen among the pool of available ones. The value of k here is fixed and we set it to the percentage of initial positive examples for every concept. We try 3 values of this percentage as $\{10\%, 20\%, 30\%\}$. Once D_s is found we use it to re-label the training set and then add k new positive labels to the training set for D_t as shown in the bottom part of figure 5.2.

Precision based Rank

The positive disagreement approach suffers from two setbacks. First is how to select the optimal value of k. For concepts with low initial positive examples, which is the case more often than not, 10% new labels are quite low while on the other hand for concepts with already abundant pool of positive examples 10% new examples may add noise. This noise will increase with k. The other predicament which is related to first one is how sure are we that the new labels added really bring valuable information. Furthermore and as more iterations are performed how useful the newly added labels are before they start to add wrong annotations and distort the data.

To cater these issues we need to find a source D_s for D_t that is expected to add up to a certain percentage of correct positive labels. So we add a **restrained number** of new examples to each view with a **certain confidence**. We use the precision on the classification results of the validation set for each of the source views to build this criterion. We restrict the classifiers to relabel only a certain number of examples some of which are expected to be correct with a certain confidence. Figure 5.2 is again used to explain this method. First for every ranked list we find k for a fixed precision value pr. That is to say using the true labels of the validation set we scroll down the ranked list calculating precision at every step and stop when the precision is lower than pr. The precision for the list V_D for a value of k is

$$P^k_{V_D} = \frac{\sum_{i=1}^{k} f(S_i)}{k} \tag{5.3}$$

and to find k for V_D we maximize equation 5.3 for k as:

$$\begin{aligned} k_s &= \underset{k}{\operatorname{argmax}}\, P^k_{V_{D_s}} \\ s.t. \quad & P^k_{V_{D_s}} >= pr \end{aligned} \tag{5.4}$$

So for the example in the figure 5.2 we find the values of k_s for precision of 50%. $k_3 = 4$ and for the other three descriptors the value of k is 2. Note that k_1 is not an important factor here as the k for target descriptor is not used in the criterion presented just after.

Once k is determined for each source descriptor, for D_t we simply select the D_s that maximizes the average rank of the first k_s shots on the validation set V_{D_t}. Maximum rank means that those shots that are ranked as positive by the source classifier are at the bottom of the list V_{D_t}. It means that this source classifier identifies the most serious misclassifications

for the target on the validation set. We define this average rank for the source descriptor D_s as:

$$PRR_s = \frac{\sum_{i=1}^{k_s} r^{V_{D_t}}(S_i)}{k_s} \tag{5.5}$$

where each source has a different PRR_s for a unique k_s and the D_s which maximizes equation 5.5 is selected for D_t. So to summarize we use the positive labels to determine the value of k_s but after we only used the average rank of shots ranked by D_t in the first k_s shots of V_{D_s}.

After determining D_s we use it to re-label the training set and add k_s new examples to the training set of D_t. Using k_s which had a precision greater than pr, it is expected that up to $pr\%$ correct labels are added to the training label set for D_t. As shown in the figure 5.2, for $pr = 50\%$, D_2 is selected with $k_2 = 2$ and then 2 new labels are added to the training set of D_1 with 1 unlabeled examples being labeled as positive and 1 negative's label flipped. Again as the positive disagreement method this selection is done separately for each concept. For further iterations of the process the modified training and validation lists are used from the previous iteration.

5.2.2 Results and Experiments

5.2.2.1 Experimental Setup

We have carried out experiments on the TRECVID 2013 [187] dataset where the development set consists of about 800 hours of internet videos of lengths varying divided into training and validation parts. The test part contains 600 hours of slightly longer videos. Training is done on a list of 60 concepts out of which NIST has evaluated 38 for the 2013 Semantic INdexing (SIN) task.

We have mainly extracted 2 kinds of descriptors from the video keyframes the SIFT [63] and the color SIFT [206] which are all densely extracted. These extracted descriptors are then used to build visual dictionaries using k-means clustering. Using the above extractions we build 5 types of descriptors of varying lengths (dictionary sizes). For dense SIFT, dictionaries of 4000 and 10,000 are built and we get two descriptors: dsift4K and dsift10K. For color dense SIFT we have cdsift1K, cdsift4K and cdsift10K from dictionaries of 1000, 4000 and 10,000 visual words.

All the classifiers used are 1 vs. all SVM classifiers using homogeneous kernel maps [137, 138] built on the input features. We have used Pegasos training [188] for speedy optimizations. We calculate the Average Precision (AP) for each concept and present the percentage Mean AP (MAP).

5.2.2.2 Results

We have conducted a certain number of experiments using the two proposed descriptor selection methods. For these experiments we have used all the labeled data provided by

Descriptor	Baseline	Selftraining				DIS				PRR			
		pr	1	2	3	k	1	2	3	pr	1	2	3
cdsift1K	8.11	**70**	8.41	7.85	7.74	**10**	8.04	8.05	7.33	**70**	7.85	8.65	8.32
		60	7.58	7.46	6.94	**20**	8.31	7.60	6.92	**60**	8.11	8.19	7.75
		50	7.88	7.85	7.01	**30**	7.67	6.54	5.60	**50**	8.23	8.21	7.98
cdsift4K	8.12	**70**	8.42	8.56	8.21	**10**	8.60	8.72	8.34	**70**	8.58	8.61	8.40
		60	8.45	8.13	8.14	**20**	8.58	8.58	7.75	**60**	8.33	8.33	8.17
		50	8.19	8.15	7.56	**30**	8.54	7.79	6.75	**50**	8.44	8.51	8.09
cdsift10K	8.18	**70**	8.28	7.88	7.69	**10**	8.54	8.49	9.19	**70**	8.60	8.68	8.99
		60	8.15	7.92	7.50	**20**	8.37	8.37	8.28	**60**	8.59	9.02	8.69
		50	8.05	7.88	7.25	**30**	8.50	8.10	7.13	**50**	8.81	8.80	8.53
dsift4K	7.52	**70**	7.87	7.76	7.50	**10**	7.68	7.91	8.16	**70**	7.59	7.77	7.40
		60	7.77	7.72	7.60	**20**	7.80	7.90	7.24	**60**	7.82	7.75	7.42
		50	7.63	7.29	7.09	**30**	7.79	7.59	6.32	**50**	7.68	7.39	7.49
dsift10K	7.84	**70**	7.94	7.86	7.98	**10**	8.31	8.27	8.40	**70**	8.02	7.96	7.75
		60	8.11	7.67	7.68	**20**	8.19	8.32	8.07	**60**	8.11	8.02	7.72
		50	8.15	7.98	7.44	**30**	7.99	7.71	6.67	**50**	8.15	8.28	8.03

TABLE 5.1: Mean Average Precision for various methods for 38 evaluated concepts. Results underlined show statistically significant improvements over the baseline.

NIST to train the initial classifiers. All the new labels are added either to the unlabeled examples or in some cases labels of negative examples are flipped. The results are detailed in the next few subsections.

Cotraining vs. 1 pass and selftraining

The two selection methods for multi-cotraining are compared with the single pass learning (Baseline) and also with selftraining or bootstrapping. Two kinds of selftrainings were done; adding fixed percentage of positive examples and adding positive annotations using the precision method, where we first calculate k_t for expected precision value using equation 5.3 and then add k_t new positive labels to the training set of D_t. Table 5.1 shows results for second kind of selftraining as it performs better among the two and compares it to the two proposed selective cotraining methods.

We show results for 3 iterations of relabeling and retraining in table 5.1 for all the semi-supervised methods. Results that are significantly better with randomization testing [190] are underlined. Selftraining and positive disagreement (DIS) methods are mostly outperformed by the Maximum rank on precision based selection (PRR). This is true for further iterations as for DIS noise is added with a fixed value of k, and selftraining lacks complementarity of using other descriptors. When k is 10% the performance of DIS is good as few noisy labels are added and it sometimes shows better results than the PRR. For the PRR criteria as the precision increases the value of k_s decreases and in many cases no new labels are added for certain categories, for example for $pr = 70\%$. Though we see an improvement for most of the descriptors for DIS and PRR the color SIFT descriptors seem to absorb more noise than others.

Cotraining vs. Linear Fusion

To check the complementarity of descriptors we fused every possible pair of the 5 descriptors

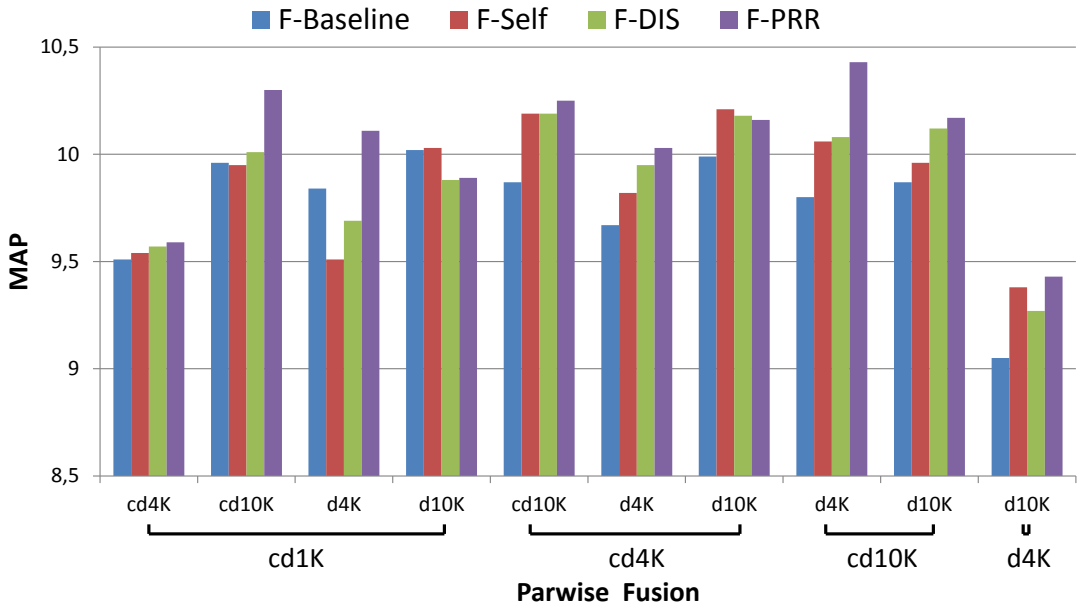


FIGURE 5.3: Linear fusion of every pair of descriptor for different methods

and compare the performance with the fusion after first iteration of each semi-supervised learning method. We have used weighted linear fusion to merge the classification scores with the weights optimized on the validation set. Results are compared in the figure 5.3 where PRR dominates and the best result of the fusion of baseline descriptors (10% MAP) is outperformed by a MAP of 10.43% for PRR.

Selected Views

Our goal was to select the most complementary view to label examples and PRR outperforms DIS for the major part. We have only used visual descriptors to generate our views with two slightly different categories of descriptors of varying sizes. Lets have a look at the views selected for the "cdsift10" and "dsift10k" that are the best descriptors in their categories. We look at the view selected for each concept off the 38 evaluated concept.

	Iteration 1			**Iteration 2**			**Iteration 3**		
Descriptor	**cdsift**	**dsift**	**None**	**cdsift**	**dsift**	**None**	**cdsift**	**dsift**	**None**
cdsift10k	12	22	4	11	25	2	6	27	5
dsift10k	32	2	4	32	4	2	26	6	6

TABLE 5.2: Views selected for PRR, precision = 50

	Iteration 1			**Iteration 2**			**Iteration 3**		
Descriptor	**cdsift**	**dsift**	**None**	**cdsift**	**dsift**	**None**	**cdsift**	**dsift**	**None**
cdsift10k	12	21	5	11	22	5	12	21	5
dsift10k	30	4	4	27	6	5	26	6	6

TABLE 5.3: Views selected for PRR, precision = 60

Descriptor	Iteration 1			Iteration 2			Iteration 3		
	cdsift	dsift	None	cdsift	dsift	None	cdsift	dsift	None
cdsift10k	11	19	8	11	18	9	13	17	8
dsift10k	27	3	8	27	2	9	30	4	4

TABLE 5.4: Views selected for PRR, precision = 70

Tables 5.2, 5.3 and 5.4 show the distribution of the views selected for the two descriptors for the values of precision at 50, 60 and 70. It is clearly visible that the selected descriptor is the different one most of the times. Moreover for "cdsift10k", "dsift10k" is selected the most out of the dsift's and also the vice versa is true. There are cases when no view is selected to do co-training when the precision criterion is not met on the validation set. This means that the particular descriptor for a concept did not achieve the required precision even for the first 2 shots on the validation set and re-training is not performed for that concept. Looking at the tables 5.2, 5.3 and 5.4, we can safely say that our method tends to select the best view to label new instances.

Adding a Different Descriptor

To further test our selection methods we add a more powerful descriptor, caffe1000 [207] in the selective multi-cotraining mechanism. caffe1000 is a 1000 dimensional descriptor containing scores of 1000 concepts trained on the ImageNet [164]. We use the same SVM settings for the

Descriptor	Baseline	DIS				PRR			
		k	1	2	3	pr	1	2	3
caffe1000	11,98	**10**	11.49	11.86	11.88	**70**	11.74	12.16	12.11
		20	12.13	11.91	11.89	**60**	11.9	12.07	12.08
		30	11.9	11.58	10.65	**50**	11.92	11.87	11.96
cdsift1K	8.11	**10**	7.85	7.98	8.23	**70**	7.93	8.56	8.15
		20	8.3	7.79	6.47	**60**	8.12	7.82	8.18
		30	7.69	6.97	6.12	**50**	7.84	8.13	8.12
cdsift4K	8.12	**10**	8.39	8.4	8.73	**70**	8.46	8.54	8.69
		20	8.33	8.31	7.79	**60**	8.62	8.51	8.31
		30	7.98	7.66	6.17	**50**	8.4	8.64	8.62
cdsift10K	8.18	**10**	8.25	8.63	8.79	**70**	8.89	8.95	9.08
		20	8.3	8.62	8.19	**60**	8.89	8.71	8.42
		30	8.68	7.6	6.89	**50**	8.45	8.63	9.19
dsift4K	7.52	**10**	7.73	7.69	8.05	**70**	7.81	7.8	7.79
		20	8.02	7.95	7.3	**60**	7.74	7.43	7.5
		30	8.17	7.41	6.55	**50**	7.63	7.81	7.8
dsift10K	7.84	**10**	8.12	8.24	8.26	**70**	7.99	8.06	8.01
		20	7.98	8.09	7.63	**60**	7.58	7.74	7.66
		30	8.13	7.25	5.96	**50**	7.94	8.36	7.78

TABLE 5.5: Mean Average Precision for the selective multi cotraining methods with the added view based on the *caffe1000* descriptor. Results underlined show statistically significant improvements over the baseline.

caffe1000 as our visual descriptors and achieve a baseline score of 0.1198 for the 38 evaluated concepts on TRECVID 2013 test set, table 5.5. Even though caffe1000 is the descriptor selected most of the times ($> 60\%$) for the other views based on visual descriptors the performance is not affected much. The best score achieved by PRR is 0.0919 for "cdsift10k", while "caffe1000" achieves a maximum of 0.121. The reason for this not-so-much increase in the performance is that though we do find the best view to re-label examples, our learners (and the descriptors) still lack the capability to capture the diversity in the examples.

5.3 Conclusions

We have demonstrated the effectiveness of selecting the appropriate feature to relabel a target learner for cotraining when different options are available. For each descriptor the best complementary descriptor is selected and the number of examples to label is also automatically selected which are expected to be correct up to a certain percentage. For future work it is required to find a successful weighing strategy to find relevance of the newly added labels. The relevance scores for a category can be attached to each example and will highlight its importance in training [205]. Furthermore it is important to (i) refine the k new positive examples added by an automatic or manual [197] selection, and (ii) automatically identify the number of cotraining iterations for each concept. Eventually the selective cotraining can be used to find negatives by reversing the selection criteria.

5.3.1 Restrained Multi-view Learning

Since we have a number of descriptors available and in the previous experiments we only discussed cotraining, we extend our conclusion by looking at what multi-view learning would give us with these descriptors. In multi-view learning all the available views (descriptors) are used to label examples. We perform this multi-view experiment to challenge our selective multi cotraining methods and test whether using all the information supersedes just selecting the best one.

The principle of multi-view learning is very simple. All the views predict the unlabeled set and the most confident predictions are added in order to extend the learning set of the target view with diverse examples. To add to the training set of a view we consider three possibilities of finding new labeled examples: *Intersection*, *Union* and *Weighted Combination*. **Intersection** only adds the common examples out of the k most confident predictions from all the other views. **Union** adds all of the most confident k predictions while in **Weighted Combination** we again take a union of the most confident predictions but each view is allowed to add new labels proportional to its performance in the previous iteration. So if k new examples are to be added performance wise ratio of k examples are added by each view.

Note that for Intersection the minimum number of new labels added is 0 when no common predictions are found among the views on the unlabeled set and the maximum number is

k if all of the views agree on the predictions. For Union the minimum number is k in the case all the views have the same examples in their top k predictions and the maximum new examples added are $|D| * k$, where $|D|$ is the number of views. For the third case of Weighted Combination the new examples added are always $\leq k$. Here we take the percentage of top k from each view, so in the case of total disagreement on the unlabeled set k examples are added. The value of k is again decided in the two ways as for the selective multi cotraining: **(i)** fixed k as the percentage of positive examples for each concept in the previous training iteration and **(i)** precision based k_s for each source descriptor, with different values of pr.

Multiview Results

Tables 5.6 (a) and (b) show multi-view learning results as the average for the 38 evaluated concepts on TRECVID 2013 dataset with the same SVM classifier settings as with the experiments done previously.

From the results in the tables 5.6 it is evident that intersection suffers from disagreement among the views and union is affected by the large number of noisy labels added. Adding new examples based on the classifiers' performance is however the saner choice. Moreover precision based selection of k outperforms its counterpart for most of the cases as was the case with selective cotraining. Analyzing and comparing the results to selective cotraining in table 5.5 we can safely say that, for this problem of concept detection on the TRECVID-type datasets, multi-view training performs more or less similar to the selective multi cotraining. The former performs lower on the sift descriptors but shows better performance, though not statistically significant, to the later on the caffe1000.

Descriptor	Baseline	k	Intersection			Union			Weighted by score		
			1	2	3	1	2	3	1	2	3
Caffe1000	11.98	10	12.06	12.2	11.9	12.24	12.01	11.86	11.73	12.04	11.71
		20	11.67	11.96	12	12.01	11.45	10.09	12.02	12.17	11.58
		30	11.71	11.87	11.89	12.06	10.18	6.84	12.06	11.76	11.99
cdsift1K	8.11	10	7.9	8.01	8.17	8.49	8.03	7.61	8.36	8.14	8.24
		20	8.11	8.02	7.93	7.74	7.32	5.71	8.07	7.94	8.07
		30	8.08	7.41	8.08	7.43	5.62	3.26	8.29	8.21	7.62
cdsift4K	8.12	10	8.16	7.85	8.04	8.72	8.72	8.09	8.66	8.71	8.55
		20	8.2	8.06	8.45	8.67	7.9	7.22	8.55	8.43	8.7
		30	8.32	8.32	8.18	8.18	6.94	4.23	8.48	8.7	8.27
cdsift10K	8.18	10	8	8.16	8.19	8.74	8.67	9.07	8.26	8.57	8.69
		20	7.78	8.13	8.04	8.65	8.36	7.44	8.49	8.51	8.75
		30	8.16	8.21	8.18	8.38	7.23	4.77	8.84	8.68	8.48
dsift4K	7.52	10	7.93	7.43	7.44	7.67	8.15	7.93	7.53	7.57	7.68
		20	7.58	7.38	7.74	8.04	7.57	6.28	7.4	8.14	8.07
		30	7.42	7.81	7.44	7.79	6.36	3.95	8.14	8.01	7.88
dsift10K	7.82	10	8.08	7.78	8.11	8.28	8.33	8.08	7.96	8.07	8.29
		20	8.09	8.04	8.08	8.44	8.05	6.76	8.12	8.26	8.05
		30	8.13	8.16	7.96	8.12	6.57	4.22	8.48	8.47	8.25

(A)

Descriptor	Baseline	pr	Intersection			Union			Weighted by score		
			1	2	3	1	2	3	1	2	3
Caffe1000	11.98	70	11.76	12.09	11.76	12.22	12.06	12.18	11.65	12.56	12.32
		60	11.8	12.03	11.84	12.47	11.85	11.55	12.1	12.26	12.5
		50	11.75	11.8	12.08	12.12	10.88	11.02	12.02	12.58	12.47
cdsift1K	8.11	70	7.75	7.72	8.04	8.34	8.12	7.6	8.03	8.17	8.55
		60	8.02	8.26	8.25	7.79	7.88	7.03	8.22	8.32	8.21
		50	8.14	7.86	7.73	7.8	6.65	6.3	8.22	8.14	8.08
cdsift4K	8.12	70	8.31	8.41	8.33	8.56	8.52	8.05	8.19	8.48	8.57
		60	8.45	8.18	8.5	8.36	8.12	7.82	8.41	8.68	8.58
		50	8.42	7.86	8.3	8.3	7.86	7.34	8.54	8.79	8.9
cdsift10K	8.18	70	8.16	8.37	8.41	8.95	8.83	8.62	8.33	8.31	9.07
		60	8.47	8.12	8.54	9.05	8.38	8.3	8.05	8.88	8.87
		50	8.32	8.18	8.2	8.63	8.22	7.84	8.54	9.05	8.63
dsift4K	7.52	70	7.53	7.56	7.93	7.4	7.09	7.16	7.51	7.66	7.99
		60	7.74	7.72	7.72	7.49	7.23	6.98	7.73	7.82	8.21
		50	7.42	7.6	7.03	7.29	6.77	6.45	7.76	7.68	7.97
dsift10K	7.82	70	8.04	7.59	8.13	8.13	7.88	7.38	8.09	8.18	8.32
		60	7.97	8.34	8.18	8.11	7.7	7.28	7.84	8.02	8.26
		50	7.95	7.93	7.87	7.88	7.35	6.68	8.47	8.26	8.21

(B)

TABLE 5.6: MAP scores for multiview learning techniques for various views (descriptors): (a) k is fixed as the percentage of positive examples of the concept, (b) k is decided based on the precision of the concept classifier on the validation set.

Chapter 6

Conclusions

Throughout this thesis we have looked at ways to improve video concept detection and developed methods to make the machine understand and recognize the visual content better. We have devised various methods to make the slice bigger incorporating extra information which is both important and useful from the Visual World when building classifiers for video concept detection. However the problem of automatically recognizing the content of videos is a tough one and is still far from being solved. The number of possible categories is limitless which leads to ever increasing video and image datasets. In addition trying to accommodate all the variations an object can be presented in a category makes the problem even harder. Nonetheless research is slowly progressing and machine's vision is improving every year [12, 17, 18, 25, 26, 187].

6.1 Key Contributions

In this dissertation we have made several contributions to address the problem of automatic detection of concepts in Internet videos. We have closely studied the steps involved in the concept detection pipeline ranging from information extraction from the raw image pixels to high level decision about the presence of the semantic concept in the video frame. After careful consideration and due to its importance in the state of the art we selected the challenging benchmark TRECVID video analysis task and have presented the detailed experimentation of our proposed methods for the Semantic Indexing task. To conclude, we first reiterate our key contributions in a few paragraphs below.

Improving Video Representation

We have used information loss principles to construct the Visual Dictionary in a supervised manner. After detailed analysis of the BOW framework we believe that incorporating category information while building the BOW representation somehow makes up for the semantic gap by explicitly including the semantics into the dictionary. Though this may lead to overfitting by building dictionaries that are specific to a concept, we have observed improved

performance when using the information about the whole concept distribution to build the dictionary.

Adding Refinement to the Video Representation

State of the art tells us that the orderless BOW histogram representation does not account for any specifics about the location of a keypoint inside the BOW clustering cell. This has certainly attracted some research attention and as a result we have seen methods proposing to include geometric, spatial and keypoint specific information that improved the discriminative power of the BOW model. We also focused our attention to this area and produced a method to add a descriptor's location specific information into the BOW representation. The extra information is fused into the BOW vector in terms of a very small vector capturing the difference of each keypoint from its cluster center which also resulted in better discrimination ability of the BOW model compared to the state of the art.

Using Multi-labeled Data: Similarity Among Concepts

This contribution challenged the normal practice when building the classification models for concepts where annotated data is used to construct the model for only that concept. We found *similar* concepts that are supposed to share certain visual properties, grouped their instances together and trained the classifiers on the combined set of examples. We demonstrated that the performance of these shared classifiers reached the performance of the single concept classifiers and also that this sharing does bring complementary information when fused with the single concept learning. We agree that the accomplishment was not as shiny as the idea but it was marred by many factors including the quality of the accumulated annotations, the method selected to find similarity, the overall low number of concepts and the increased complexity of the non-linear group classifier. However our *intelligent* grouping criterion proved to be better than random grouping of concepts and the RAKEL based grouping for video concept detection of TRECVID videos.

Using Multi-labeled Data: Similarity and Dissimilarity Among Concepts

Here we integrated *dissimilarity* in addition to similarity when grouping (rather partitioning) the set of concepts and observed that the shared classifiers matched and sometimes surpassed the performance of single concept learning with classifiers that were trained using considerably less training data than the norm. The overall number of classifiers trained were also less than the baseline which is important considering the complexity of machine learning methods.

Using Unlabeled Data: View Selection for Cotraining

Finally we turned our attention to the immense quantity of unlabeled video data that is readily available all over the internet and consists of a large part of TRECVID style partially annotated video datasets. We have studied the semi supervised learning methods and looked at the nifty method of cotraining to iteratively improve classifiers using unlabeled data but realized the sad reality of its disappointment for visual analysis methods, particularly video concept detection. With the gradual evolution of the field in terms of performance and

scalability, semi-supervised learning is also gaining importance as it relieves us of manually and painstakingly labeling each and every keyframe of unlabeled videos. We deem that it is important to find the most useful of the information among the unknown and put it to good use. To that end we developed methods to select the most complementary cotraining view to increase the size of the training set of a classifier with the most constructive and informative examples.

6.2 Promising Directions

We divide the future perspectives into the work that extends the presented contributions and some general perspectives for content based indexing and retrieval.

6.2.1 Immediate Future Work

In the context of our proposals we have presented some future directions throughout the dissertation but here we would like to stress certain points that could lead to immediate future work. Addressing chapter 3 different kernel design should be envisaged for the DBOW bins as it models different information from the BOW histogram. Alternatively an effective weighting mechanism [132] for the DBOW could be developed as well as finding the optimal DBOW size for a given visual dictionary. Since each high dimensional Voronoi cell has distinct dynamics a cell based quantization would result in more precise representation but at a higher cost. Concept wise DBOW construction is also a possible refinement by making a DBOW histogram separately for each concept using descriptors from images labeled only with that concept.

A requirement that is felt throughout the thesis is the need to go bigger. Since the real world is way bigger than a list of tens of concepts what would happen if larger dataset with considerably more concepts and increased set of examples is used? Considering direct application of the proposed methods in chapter 4 we believe that sharing among classes will always help and with more classes present better choices of grouping and partitioning will be available to increase learnability of the system. The group based classification will not scale well to the larger pool of information but the partitioning based methods are scalable. An example is to use the list of concepts from the full TRECVID Semantic Indexing task [18] to generate the partitions and train the classifiers, and then calculating the performance on the light task. As shown in the analysis in section 4.4.2 increasing the size of the list of concepts the performance of the label partitioning improves. Moreover the sections 4.3 and 4.4 show that the label space partitioning based methods generally use noticeably fewer examples with fewer classifiers to achieve the performance obtained by the single concept learning. So the presence of more information would actually benefit the system in terms of efficiency and performance.

Pursuing more work in semi-supervised learning is interesting especially for finding the most useful and a minimum number of new labeled examples. From the selective cotraining and multi-view results in chapter 5 we saw that overall the views only agree to a very small degree on the relevance of the unlabeled examples. Still we saw improved performance for all the views. We believe that adding a much stronger view (a good descriptor with a strong classifier) in the selective cotraining mechanism would improve the results further by pointing the others into the right direction.

In the two view selection criteria we devised we always fixed the values which determined the number of new examples to be added. This could be changed so that the criterion dynamically updates according to the performance of the classifier (view) during the course of the retraining iterations. An example is staring with 30% new examples to be added for a concept and then decreasing this percentage along the way. This also makes sense as we have used a fixed dataset where only few of the unlabeled examples would benefit the concept classifier if at all. Furthermore using such a methodology would automatically include a stopping criterion when the performance on the validation set starts to decline. Another extension is to revert back to a previous iteration if performance on the validation set declines, like the view validation in [106].

6.2.2 General Perspectives

There exists still a huge gap in the improving scientific research published year by year and the acceptance of state of the art methods in industry or real applications. This disconnect between the research and its applicability in real life has been felt by many researchers and various applications are now emerging. Some popular examples for image based retrieval systems are the INRIA's IKONA [208], LIRE [209], pixolu, and the SHIATSU system [210] for keyframe based video indexing. The importance of content cannot be denied and it is about time that industrial search engines replace keyword based search with content based indexing or at-least include it in their framework. Google and the Russian search engine Yandex led the pioneering work and offer online content based image search.

Scalability of the methods is also important to make them portable to be used on the numerous hand held devices. Smartphones and tablets are now equipped with high resolution cameras and sophisticated hardware with ample processing power. Applications like automatic recognition of commodities, etc. are already making wave in the online stores of various platforms.

Research work has continued to evolve and promising results are published on harder datasets every year. Lately the focus has been on de-correlating the training and the test set [154, 160, 161] (training models that are not specific to the dataset used). This means the test set is taken from a different source which is generally not the case for the databases used in scientific research these days.

Zero shot learning [152, 161, 177] is also gaining importance as it tackles the problem of annotating new emerging concepts. The content on the internet continues to grow and so does the list of semantic concepts. Moreover the social trends have to be taken into account as something that is relevant now may not be entered into the bar of a search engine for years to come.

Recently deep learning [211] has pushed the limits of performance on understanding the semantics for concept detection using Convolutional Neural Networks [102, 212]. They have the ability to automatically extract and devise meaningful features from the raw pixel values of the images. Nevertheless the deep belief networks are hard to train, require great amount of training data and in some cases special hardware. This may be regarded as another peek in the classification algorithms like Neural Networks back in the 80's but for now nothing matches their accuracy.

To conclude, it is safe to say that we are headed in the right direction to make our machines to be able to figure out the visual content. The information is there and is growing and so is the processing power of our computers. It is a matter of using it correctly with the correct methods. We also need to expand our horizons by exploring other cues that may be important in understanding the content. Making sense of the interaction among entities over the internet, the social activity, user modeling to get a glimpse of the knowledge and interests of the uploader of the material [213, 214], user's location and mutual interest among the users of content sharing services are some of the cues that could be mined to make further sense of the Visual World.

Appendix A

Our Contributions

Conference Proceedings

1. Q. Li, U. Niaz and B. Merialdo. An improved algorithm on viola-jones object detector. *In CBMI 2012*, June 27-29, 2012, Annecy, France

2. U. Niaz and B. Merialdo. Entropy based supervised merging for visual categorization. *In ACIVS 2012*, 4 September 2012, 7517/2012-Springer, Brno, Czech Republic

3. U. Niaz and B. Merialdo. Fusion methods for multimodal indexing of web data. *In WIAMIS 2013*, July 3-5, 2013, Paris, France

4. U. Niaz and B. Merialdo. Exploring intra-bow statistics for improving visual categorization. *In WIAMIS 2013*, July 3-5, 2013, Paris, France

5. U. Niaz and B. Merialdo. Improving video concept detection using uploader model. *In ICME 2013*, July 15-19, 2013, San Jose, California, USA

6. U. Niaz and B. Merialdo. Leveraging from group classification for video concept detection. *In CBMI 2013*, June 17-19 2013, Veszprem, Hungary

7. U. Niaz and B. Merialdo. Selective multi-cotraining for video concept detection. *In ICMR 2014*, April 1-4, 2014, Glasgow, Scotland.

8. B. Merialdo. and U. Niaz Uploader models for video concept detection. *In CBMI 2014*, 18-20 June 2014, Klagenfurt, Austria

9. U. Niaz and B. Merialdo. Improving video concept detection through label space partitioning. *In ICME 2014*, July 14-18, 2014, Chengdu, China

Workshop Proceedings

1. U. Niaz, M. Redi, C. Tanase and B. Merialdo. EURECOM at TRECVID 2011: The light semantic indexing task. *In TRECVID 2011*, National Institute of Standards and Technology, Gaithersburg, USA

2. U. Niaz, M. Redi, C. Tanase, B. Merialdo, G. Farinella and Q. Li. EURECOM at TRECVID 2012: The light semantic indexing task. *In TRECVID 2012*, National Institute of Standards and Technology, Gaithersburg, USA

Appendix B

Amélioration de la détection des concepts dans les vidéos par de plus grandes tranches du Monde Visuel

B.1 Résumé

Les documents visuels comprenant des images et des vidéos sont en croissance rapide sur Internet et dans nos collections personnelles. Cela nécessite une analyse automatique du contenu visuel qui fait appel à la conception de méthodes intelligentes pour correctement indexer, rechercher et récupérer des images et des vidéos.

Nous sommes entrés dans une époque où la dépendance de l'humanité sur l'informatique n'a jamais été si fortement ressentie avant. Cependant, nous ne sommes pas encore en mesure de rendre nos machines voir et comprendre le monde autour de nous comme nous, les humains. Les premiers succès de la vision par ordinateur après la conception du domaine il ya 50 ans ont encouragés la communauté scientifique à l'époque de prédire la résolution complète du problème de la compréhension automatique du matériel visuel dans quelques années. Des dizaines des années plus tard et nos ordinateurs atteindre qu'un succès partiel dans la détection du contenu représenté dans une image, et la performance e détection tombe encore pour une vidéo.

La tâche difficile de l'indexation automatique et la récupération des vidéos en fonction de leur contenu est abondamment traitée par l'exploration académique et industrielle. Des décennies de recherche sur des méthodes de l'indexation automatique basés sur le traitement textuelle, puis vers l'indexation des vidéos basés sur la compréhension automatique du contenu ont conduit à la mise au point de systèmes ingénieux. Ces systèmes complexes utilisent principalement l'analyse de l'image et des outils d'apprentissage statistique et plus ou moins comprennent plusieurs étapes complexes pour rendre la machine capable de reconnaître le contenu. Toutefois, contrairement les systèmes d'indexation textuelle, la compréhension

automatique du contenu vidéo est loin d'être résolu avec les machines d'aujourd'hui et en utilisant les méthodes actuelles. Une grande quantité d'information est extraite des vidéos dont certains sont utilisés efficacement et intelligemment pour atteindre l'objectif en étant conscient de l'extensibilité et de la complexité du système.

En comprenant l'importance de l'indexation et de la recherche par le contenu et après avoir fait des recherches approfondies sur l'état de l'art dans le domaine nous pensons que le système automatique peut être amélioré de nombreuses façons. Nous nous adressons principalement les informations qui ne sont pas utilisés lors de la construction d'un tel système, alors qu'ils sont déjà là. Cette thèse vise à améliorer la détection automatique des concepts dans les vidéos sur Internet en explorant toutes les informations disponibles et de profiter du plus bénéfique de cela. Nous la regardons comme faire une coupe pour avoir une tranche plus grande du monde visuelle. Cette grande tranche du monde visuelle est notre système de qui contient l'information plus utile et bénéfique.

Cette thèse vise à améliorer la détection automatique des concepts dans les vidéos sur Internet. Nos contributions portent sur des différents niveaux dans le cadre de détection de concept et peuvent être divisés en trois parties principales. La première partie se focalise sur l'amélioration du modèle de représentation des vidéos "Bag-of-Words (BOW)" en proposant un nouveau mécanisme de construction qui utilise des étiquettes de concepts et une autre technique qui ajoute un raffinement à la signature BOW basée sur la distribution de ses éléments. Nous élaborons ensuite des méthodes pour intégrer des entités semblables et dissemblables pour construire des modèles de reconnaissance améliorés dans la deuxième partie. A ce stade-là, nous observons l'information potentielle que les concepts partagent et construisons des modèles pour les méta-concepts dont sont dérivés les résultats spécifiques de concepts. Cela améliore la reconnaissance des concepts qui ont peu d'exemples annotés. Enfin, nous concevons certaines méthodes d'apprentissage semi-supervisé pour bénéficier de la quantité importante de données non étiquetées. Nous proposons des techniques pour améliorer l'algorithme de cotraining avec une sélection optimale des classifieurs utilisés.

B.2 Introduction

On dit qu'une image vaut mille mots, mais ce n'est qu'un euphémisme dans le domaine de vision par ordinateur. En effet, pour les scientifiques de la vision, une image peut contenir des millions de mots *Visuel*. Ces mots visuels sont construits en utilisant des divers éléments visuels qui capturent des mesures importantes à partir d'images (couleur, texture, des arrangements spéciale, les statistiques globaux, etc. ...). Toute cette multitude d'informations est utilisé pour aider les ordinateurs à capturer la compréhension du contenu des images. Il s'agit d'une tentative de faire des machines à voir et comprendre le monde comme nous le faisons. Cependant cette multi-méga-octet d'information attend un succès partiel et la reconnaissance de contenu dans les images continuent d'échapper à nos machines intelligentes.

L'analyse vidéo est une version plus complexe du même problème où une série d'images entrent en jeu. Ici nous avons les relations entre les images de vidéo et les dépendances temporelles. Le cœur de traitement constitue pourtant l'analyse au niveau de l'image. Analyse d'images / vidéo automatique se compose des tâches diverses, comme la catégorisation, la recherche, la détection du copie d'une vidéo, la détection d'événements, etc., à la tête de laquelle se trouve la capacité de reconnaître le contenu visuel basé sur la sémantique de l'image. Manipulation des images et des documents vidéo pour l'analyse automatique est l'un des défis les plus difficiles dans le domaine du vision par ordinateur [1]. Cependant, la nécessité d'une reconnaissance automatique du contenu visuel n'a jamais été plus fortement ressenti avant avec l'augmentation exponentielle de la quantité de l'information visuelle sur Internet. Environ 350 millions de nouvelles photos sont mis sur Facebook chaque jour [2], et le nombre pour Flickr est de 20 à 40 millions par jour [3]. En outre Facebook possède plus d'un quart de billion de photos sur leur site web. Environs 100 heures de nouvelle vidéo est mises en ligne chaque minute sur YouTube. Le nombre d'utilisateurs qui utilisent ces services est de plus en plus par jour et ce moyen de communication en ligne maintenant correspond l'importance de la diffusion de télévision. Cette énorme quantité de nouvelles informations et son importance pour les utilisateurs appels à des méthodes fiables et efficaces pour analyser le contenu visuel afin de développer des méthodes pour rechercher, indexer et parcourir automatiquement ces grandes bases de données.

Dans cette introduction, nous commençons par expliquer la tâche de la détection de concepts dans les vidéos qui est aussi appelé l'indexation sémantique. Dans la suite nous présentons un aperçu général des principales étapes dans la réalisation de détection de concept. En outre, nous analysons les possibilités d'amélioration dans le pipeline de détection des concepts et présentons nos motivations pour travailler dans les directions sélectionnées. Nous listons à la fin nos contributions pour améliorer la performance de détection des concepts dans les vidéos.

B.2.1 La détection des concepts dans les vidéos, Indexation Sémantique

Pendant des années, la recherche scientifique dans le domaine de l'indexation des l'image et les vidéos a été dominée par des approches basées sur le texte ou un concept basé où les *métadonnés* comprenant du texte comme titre, un article ou un narratif et les étiquettes, etc. sont utilisés pour récupérer du contenu multimédia à partir de l'Internet. Pratiquement, cela *est* la méthode populaire ces jours-ci pour obtenir des images des moteurs de recherche populaires et regarder des vidéos sur des services de partage vidéo tels que YouTube et Dailymotion. Depuis l'aube du siècle le thème de recherche a été déplacé à comprendre directement le contenu des documents multimédia et en utilisant les informations extraites pour construire des modèles de récupération. Des descriptions textuelles associées à ou autour du

contenu multimédia sur Internet ne sont pas toujours fiables. Ce sont généralement subjective de l'uploader, sont des fois personnalisé, trompeuse, incomplète et parfois n'existent pas du tout. Annoter ailleurs une grande base de données vidéo à partir de zéro avec un effort humain manuel est laborieuse et pourrait souffrir de l'incomplétude et du préjuge. Le contenu, autrement, a toujours raison et nous pouvons compter en toute sécurité sur son authenticité, à condition de son utilisation correcte.

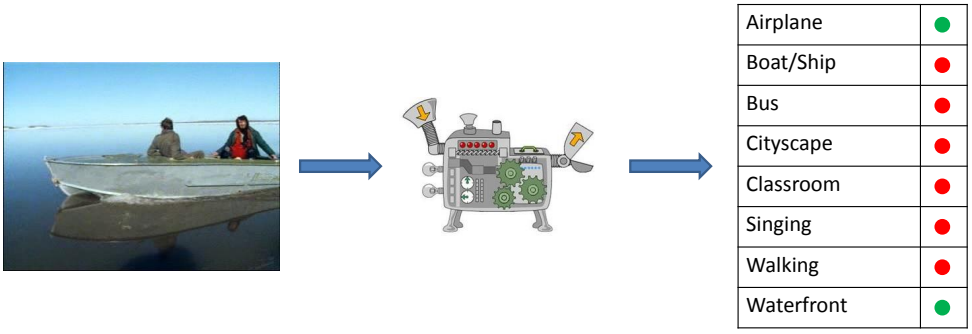


FIGURE B.1: Détection de concept dans une vidéo

La détection des concepts dans les vidéos ou bien la catégorisation des vidéos vise à décrire automatiquement une vidéo avec des concepts sémantiques qui correspondent au contenu de la vidéo, figure B.1. Ces concepts **sémantiques** sont des descriptions de haut niveau de la vidéo qui représentent directement les informations clés présents dans le contenu. Le concept sémantique peut consister simplement d'étiquettes comme des objets ou des personnes, peut être représentée par une scène ou peut comprendre une situation avec une interaction complexe de différentes entités. La solution attribue une probabilité de présence d'un concept ou d'une étiquette dans le cadre de la vidéo (par exemple, la vidéo contient le concept "Bus" et est moins susceptible de contenir le concept "l'homme dansant"). La probabilité est assignée par un classificateur qui est fabriqué pour ce concept. Ce modèle de classification est construit pour chaque concept séparément. Le classement n'est pas fait pour chaque image de la vidéo, mais plutôt un ensemble de *images-clés* sont extraites pour chaque vidéo [4, 5]. Ces **images-clés** sont des *représentant* du contenu de la vidéo. Une vidéo est d'abord segmenté en *des morceaux* ou *shots* où un nouvel emplacement de shot peut être administré avec les métadonnées de la vidéo ou peut être détecté automatiquement [5]. Habituellement, une seule image-clé est alors extrait de chaque shot de vidéo qui est le plus informatif des cadres de shots. Comme l'analyse des vidéos avec des images clés offrent une alternative pratique et efficace pour l'analyse vidéo dans son ensemble [4–6] nous optons pour travailler avec des images clés tout au long de cette thèse.

Un concept sémantique ne possède pas une description fixé ou unique la plupart du temps et il existe de nombreuses variations au sein d'une classe, qu'on appelle *les variations intra-classe* . Le cadre de détection doit capturer cette variabilité intra-classe lors de la construction des modèles de classification pour les concepts. Le système de détection de

concept de vidéo qui commence à travailler avec les données brutes de pixel à partir d'images et attribue une probabilité d'existence d'un concept à une image de teste est très complexe et se compose de quelques étapes de traitement. Ces étapes sont détaillées dans la section suivante.

B.2.2 Le *Pipeline* du détection

Un système de la catégorisation des vidéos comprend une structure complexe d'éléments plus ou moins séquentielles qui s'étendent sur un certain nombre de disciplines scientifiques ; y compris le traitement du signal et de l'image, l'analyse statistique, l'extraction des données et de l'apprentissage pour n'en nommer que quelques-uns. Figure B.2 présente les principaux éléments du pipeline où nous commençons avec des images (images-clés) en supposant que la vidéo a déjà été segmentée en shots. La construction de différents modèles de détection emploie plus ou moins toutes les étapes du pipline et ne diffèrent que par la fonctionnalité de certains des étages.

Le système extrait un certain nombre de caractéristiques ou des traits visuels des images à l'étape 1 du cadre. Ces caractéristiques brutes sont généralement pas utilisées directement et sont plutôt transformées en une représentation spécifique de l'image agrégée à l'étape 2 de la canalisation pour un traitement ultérieur. Un certain nombre d'images annotées est généralement disponible pour apprendre le modèle qui classifie les images du concept du reste. Cela fait la 3ème étape où le modèle de classification est utilisée pour trouver des prédictions sur les images de teste qui décrire les perspectives de l'existence de ce concept dans la vidéo correspondante. La quatrième étape est une étape supplémentaire mais très utile lorsque les prédictions de différents modèles sont combinés pour unir les puissances des classificateurs individuels. Nous décrivons brièvement chacun des étages du pipeline en dessous avant la recherche dans les zones possibles pour l'amélioration des composantes du pipeline.

Etape 1: L'extraction et la description des caractéristiques

Une image dans une vidéo typique se compose de dizaines de milliers de pixels qui sont des chiffres simples, mais une énorme quantité d'informations 'disant les caractéristiques ou les traits visuels' peut être extrait à partir d'une seule image. Cette étape est à la base du système de détection des concepts car ici on s'attend de capturer l'essence de ce que l'image représente. L'homme peut, la plupart du temps, comprendre le concept représenté dans l'image simplement en prenant un coup d'oeil. Les caractéristiques sont attendues pour encapsuler cette essence de l'image pour les machines. Ils capturent la sémantique de l'image pour rendre la machine comprendre la similitude ou de dissemblance entre les images.

Une image clé d'une vidéo peut être susceptible à extraction des caractéristiques globales ou peut être coupé en des petites régions avant de construire des descriptions pour chaque

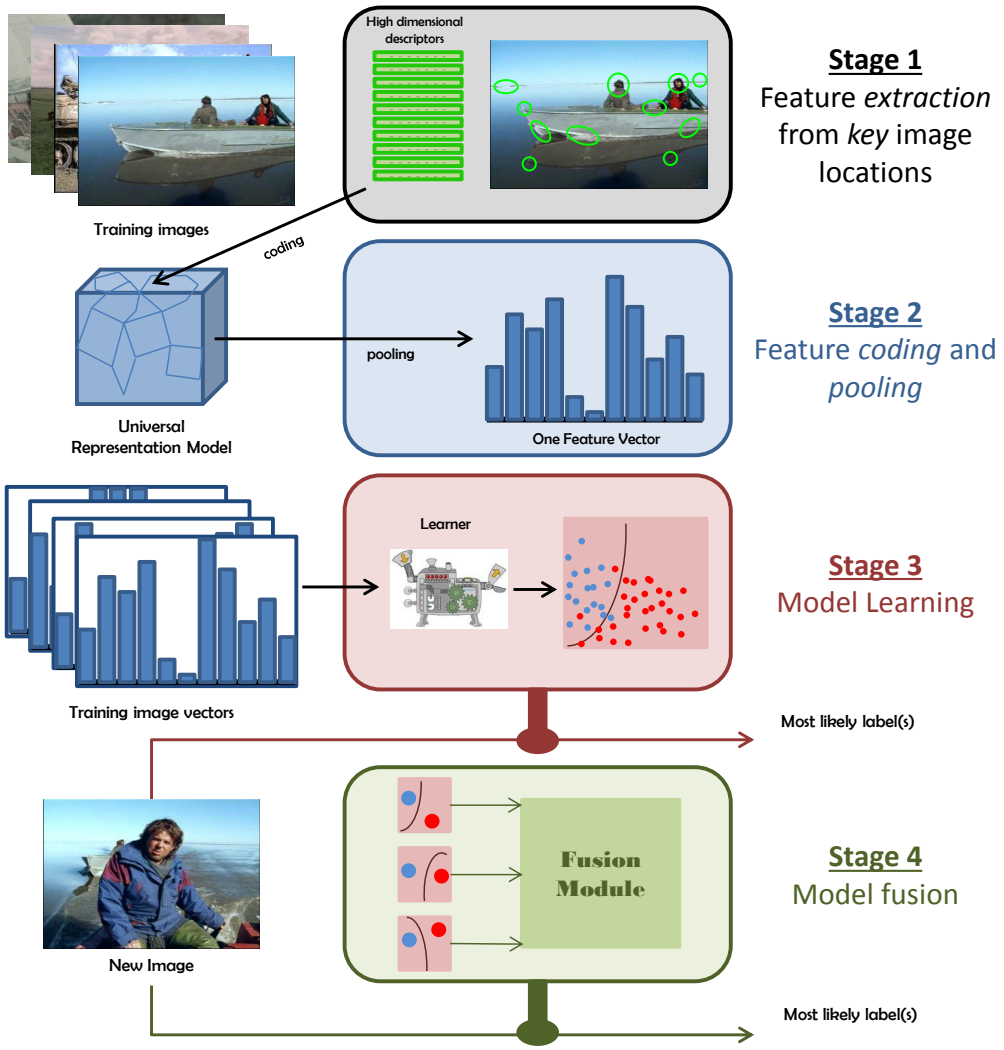


FIGURE B.2: Le pipeline de détection.

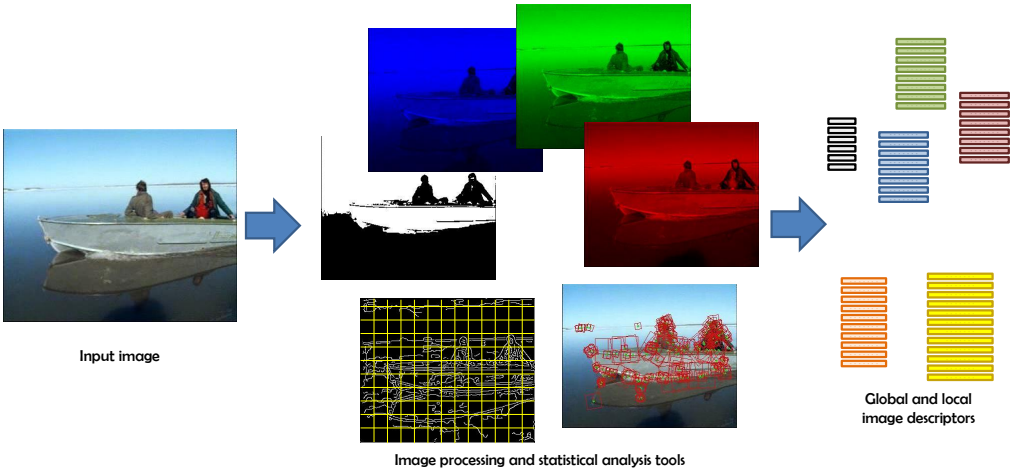


FIGURE B.3: L'extraction des caractéristiques locales et globales.

région. Ces descripteurs sont appelés les caractéristiques *visuels* de l'image. Alors que **caractéristiques globales** capturent les qualités ou les propriétés sur l'ensemble de l'image, les **caractéristiques locales** travaille sur un voisinage local de l'image pour recueillir des informations pour le pixel traité. De l'autre coté les caractéristiques globales peuvent être calculées pour l'image entière ou une partie de l'image en segmentant l'image en plusieurs parties prédéfinies, puis la construction de la caractéristique pour chaque segment. Ils décrivent les propriétés globales de l'image (segment) comme la couleur, le gist ou de l'information spatiale. Caractéristiques locales peuvent être décrites pour les points prédéfinies dans les images et sont parfois extraites des points spécifiques de l'image. Ces points sont censés contenir des informations riches par rapport au reste de l'image. Il y a des méthodes qui sont utilisées pour trouver ces points critiques fondées sur un critère de maximisation. Le nombre de caractéristiques peut donc varier d'une image à l'autre si ce type d'extraction des *points-clés* est fait. Les statistiques locales incluent des propriétés comme les informations sur la forme et la géométrie, la structure locale, texture, etc .. Figure B.3 représente l'idée d'extraire des caractéristiques globales et locales à partir d'une image par exemple. Notez que la taille et le nombre de caractéristiques extraites sont différents pour les types de traits différents.

Les caractéristiques extraites doivent être pertinentes à la tâche de détection de concept et devraient être robuste. Donc, si par exemple l'apparition de un objet change dans l'image les caractéristiques doivent toujours être en mesure de saisir l'information pertinente et nécessaire. Elles doivent également résister à la déformation, changement accidentel d'images, de variation d'illumination, le mouvement de la caméra et d'autres formes de distorsions. Une bonne connaissance du traitement de l'image est nécessaire pour construire de telles caractéristiques.

Habituellement, les caractéristiques d'un type spécifique sont utilisées pour construire le reste du système, mais parfois différents types d'éléments peuvent être fusionnés pour effectuer la tâche. Plus de discussion sur l'utilisation des caractéristiques est à suivre.

Etape 2: Codage et regroupment des descripteurs visuels

Les différents types de caractéristiques extraites à l'étape précédente peuvent être directement utilisés pour représenter une image mais leur utilisation est prohibitive en raison de leur taille et le nombre énorme. C'est plus souvent le cas pour les descripteurs d'images locales qui sont extraites à de nombreux endroits de l'image. En outre, le nombre de ces caractéristiques extraites à partir des images pourrait être différente aussi qui interdit une représentation unifiée pour toutes les images. C'est pourquoi les caractéristiques sont généralement *résumée* ou *agrégées* en un seul vecteur de haute dimensions et de longueur fixe, figure B.4.

Pour construire la représentation unifiée d'abord les caractéristiques de l'image sont transformées ou **codées** dans un ensemble de valeurs. Cette projection de la fonction sur la nouvelle représentation a certaines propriétés: deux caractéristiques qui sont proches sont codés dans la même représentation, les représentations sont compacts et la plupart sont de zéro pour

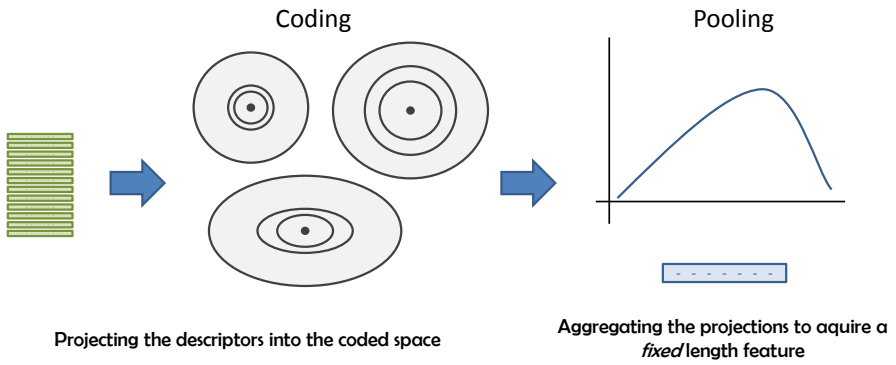


FIGURE B.4: Codage et agrégation des caractéristiques de bas niveau pour généré un descripteur resumé de l'image.

une image [7]. Quelques exemples de codage comprennent la quantification vectorielle [8], quantification 'soft' [9, 10] et estimation de la probabilité de GMM pour générer des vecteurs Fisher [11]. La nouvelle représentation ou le dictionnaire est construit en utilisant les caractéristiques de quelques images dans le base d'entraînement. Après chaque caractéristique d'image est codée, **le pooling** ou bien **le regroupment** regroupe les fonctions codées en une valeur unique pour chacun des codes. Il en résulte un vecteur de valeurs de longueur égale au nombre de représentations codées. L'opération de pooling peut varier par exemple de prenant la moyenne à prenant la valeur maximale de paramétrer l'adaptation dans le cas des vecteurs Fisher [11, 12].

L'avantage de ces vecteurs de longueur fixe est qu'ils peuvent être directement utilisés pour construire un classificateur discriminant et comme ca un modèle pour un concept peut être construit. En outre, la taille réduite permet l'apprentissage rapide et rend la prédiction des exemples de teste aussi rapide. De l'autre côté la capacité des caractéristiques qui capturent les dynamiques de l'image à partir des endroits spécifiques est quelque peu perdu maintenant, car les caractéristiques locales sont agrégées. Les valeurs extrêmes peuvent être diminuées en raison de prendre la moyenne ou la maximisation et la structure géométrique ne peut être plus utilisé. Des raffinements peuvent être faits pour inclure quelques informations utiles qui récupèrent une partie de la dynamique perdus de l'image originale. La taille de la représentation codée peut être augmentée afin de mieux distinguer les caractéristiques de l'image mais à la charge d'augmenter le temps d'entraînement et de prédiction et le risque de surapprentissage.

Etape 3: Modèle d'apprentissage

Cette étape est le cerveau du pipeline où la machine apprend à identifier correctement des exemples d'un concept. Cela nécessite la présence de certaines images annotées qui doivent être utilisés pour l'apprentissage du modèle d'un concept sémantique. Ces images sont traitées comme des exemples positifs et sont utilisés pour former le modèle de classification pour les distinguer du reste du monde. Pour être en mesure d'atteindre ce comportement intelligent

beaucoup de calculs est nécessaire et généralement cette étape prend la plupart du temps dans le pipeline. Ce temps d'entraînement augmente de manière linéaire avec le nombre de concepts dans la pratique comme un modèle distinct est appris pour chaque concept.

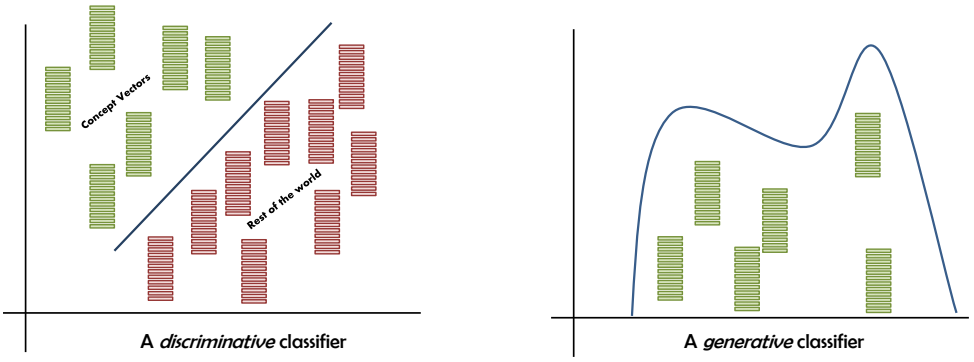


FIGURE B.5: (a) Un classificateur discriminatif distinguant les exemples positifs et négatifs. (b) Un classificateur generatif adapté à la distribution des exemples positifs.

L'etat de l'art dans l'apprentissage de ce type de modèle est principalement divisé en deux parallèles: l'apprentissage discriminatif et d'apprentissage génératif, figure B.5. Les classificateurs **discriminatifs** apprennent à associer les exemples de la base d'entraînement x à l'étiquette de la classe y [13]. Ce résultat est obtenu par la modélisation de la probabilité conditionnelle $P(y|x)$ directement à partir de l'exemples d'entraînements x et leurs étiquettes respectives y. D'autre part, les classificateurs **Generatif** apprennent la distribution de probabilité joint $P(x, y)$. La règle de Bayes est utilisée pour trouver la probabilité conditionnelle $P(y|x)$ de prédire l'étiquette la plus probable. Donc, pour reprendre un modèle génératif tente de savoir comment les données ont été générées, et peut être utilisé par exemple pour générer plusieurs échantillons de données, tandis que le modèle discriminant ne se soucie pas de la génération de données et apprend à classer directement.

La qualité du classificateur appris dépend non seulement du type de méthode d'apprentissage utilisé et de ses paramètres, mais aussi sur le type de caractéristique sélectionnée et de la qualité des exemples positifs et négatifs acquises pour entraînement. Dans cette thèse, nous avons principalement mis l'accent sur les classificateur discriminatifs.

Stage 4: Fusion

Bien que le pipeline de détection concept à peu près est complété à l'étape précédente, combinant une variété d'apprenants sur différents descripteurs est toujours une option fructueuse. Fusion permet aux classificateurs d'être construits pour différents types de caractéristiques de façon indépendante et seulement alors être combinés lors de la génération de la probabilité finale du concept. Les différents types de caractéristiques peuvent inclure d'autres modalités comme l'audio et l'information textuelle avec des traits visuelles. Ce type de fusion est appelé fusion de décisions ou fusion tardive comme cela se fait au niveau de la décision, comme indiqué dans la figure B.6. Une fusion précoce au niveau des caractéristiques peut également

être envisagé et apparaît dans la littérature de temps en temps mais il est moins populaire en raison de la combinaison hétérogène et l'augmentation de la taille de la caractéristique combinée.

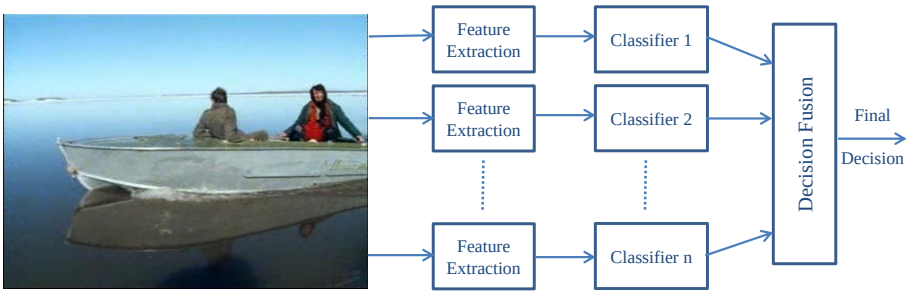


FIGURE B.6: La fusion précoce et tardive

B.2.3 Motivation

Les informations extraites des images vidéo au niveau du pixel sont des descriptions de bas niveau de l'image, ce qui ne se traduit pas directement à la sémantique de l'image. Il existe donc un *gap sémantique* [14] entre les caractéristiques ou des descripteurs de bas niveau que le système de vision utilise, pour en apprendre sur la sémantique de haut niveau que l'image représente. Il ya plusieurs raisons pour l'existence de cet écart. Le plus important est la différence dans la perception humaine d'un concept et la représentation appropriée pour nos ordinateurs à comprendre. Ensuite, il ya la subjectivité [15, 16], c'est à dire lorsque différents utilisateurs perçoivent différemment le contenu qui est similaire. Enfin, le gap sémantique est élargi par la diversité intra-classe, et cela arrive pour de nombreux concepts. Pour souligner ce problème envisager une image d'une chaise placée à l'intérieur d'un bureau et une autre image d'une chaise à l'extérieur dans une pelouse. Bien que les deux images contiennent le concept *Chaise*, les pixels de bas niveau ne seraient pas d'accord pour la plus grande partie des deux images. La recherche en interprétation automatique de contenu vidéo a été portée à réduire ce gap sémantique.

La performance d'un système de catégorisation dépend de tous les éléments qui constituent le pipeline de détection. Pour chacune de ces étapes, nous énumérons ci-dessous les principales possibilités d'amélioration.

- **Les caractéristiques extraites au niveau du pixel**
  Comme la base du pipeline de détection de cette phase reçoit une attention considérable de recherche afin de capturer l'information la plus utile avec une taille raisonnable à partir des images.

- **La sélection et la construction de la plus informative représentation (mi-niveau) agregée**
  Résumer ou agréger les caractéristiques visuelles est l'étape la plus sensible dans le pipeline comme la plupart des informations de bas niveau brut est perdu ici. La recherche dans ce domaine se concentre sur la construction des représentations de niveau intermédiaire détaillées qui capturent avec succès la sémantique de l'image.

- **Raffinement de la représentation agregée**
  Recherche a également été fait pour affiner la représentation mi-niveau pour essayer de réduire l'écart sémantique en ajoutant par exemple contextuelle, l'information spatiale ou l'étiquette spécifique.

- **Qualité des annotations / sélection des meilleurs exemples**
  Le travail fastidieux d'extraction des caractéristiques et de les résumant n'est d'aucune utilité si des exemples représentatifs ne sont pas utilisés pour le modèle de l'apprentissage des concepts. Sélection des meilleurs exemples pour faire de l'entraînement et de les affiner avec le *feedback* est importante pour la performance du système. En outre, il existe toujours une grande quantité d'exemples non étiquetés à découvrir. Nouvelles instances devraient être ajoutées qui rehaussera la diversité tout en augmentant les performances.

- **La recherche du meilleur hyper-plan de décision**
  Finalement le classificateur distingue entre les cas positifs et négatifs. La recherche continue à évoluer dans ce domaine a fin de construire des classificateurs généralisant bien sur des exemples de tests et avec moins de complexité.

- **La fusion efficace des classificateurs**
  La recherche continue pour trouver de nouvelles méthodes pour sélectionner les meilleurs classificateurs entre le pool des classificateurs multimodales et les combiner de la façon la plus efficace.

B.2.4 Elargissant la tranche

Avec l'énorme quantité d'informations visuelles extraites des images, les différentes méthodes de codage et de pooling et la variété des techniques de classification disponibles, tout n'est pas utilisé pour construire le système de catégorisation. Si trop des caractéristiques ou des paramètres sont utilisés pour former les modèles de concepts, il ya un risque de sur-apprentissage sur les exemples d'entraînement. De plus cela nécessiterait des tas de puissance et le temps de calcul car les représentations visuelles sont de grande dimension et la puissance de traitement actuel limite l'utilisation de l'extravagance de cette information visuelle.

Nous pouvons regarder toutes ces informations comme formant un *Monde Visuel*, figure B.7, et le plus souvent un système de détection de concept dans les vidéo ne prend qu'une

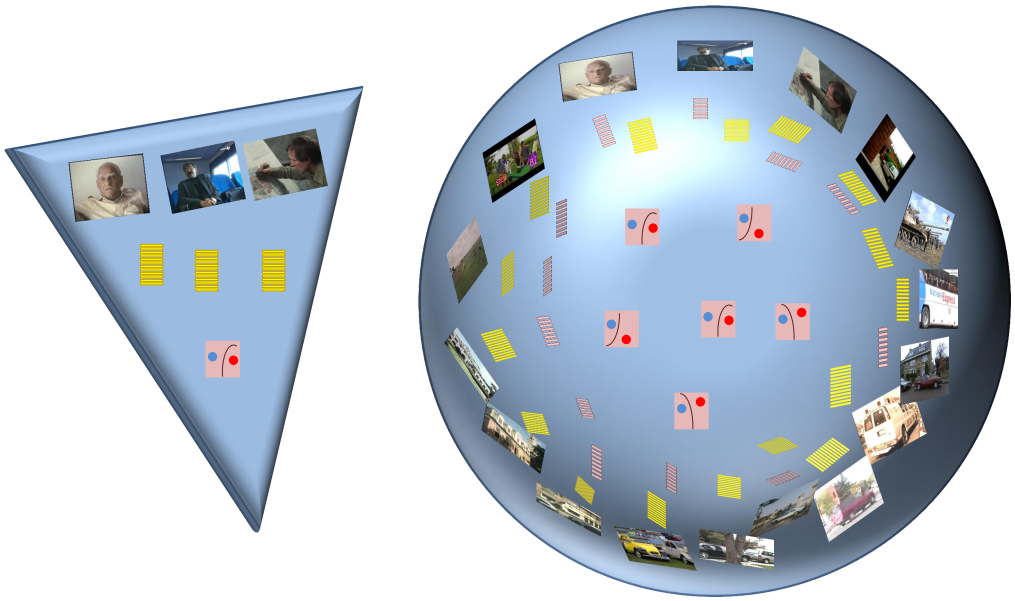


FIGURE B.7: Le monde visuel avec de nombreuses images, descripteurs et classificateurs.

tranche de ce monde avec quelques d'images annotées, certaines caractéristiques et un algorithme de classification qui apprend un modèle à l'aide de cette information. La quantité d'informations allant dans la tranche va de l'information brute au niveau de l'image de la décision de classification qui justifie la structure pyramidale. Ce que nous visons dans cette thèse est de savoir comment faire de cette tranche plus grosse, en explorant le monde visuel afin d'ajouter des informations supplémentaires et utiles avec peu d'effet sur la complexité de la tâche d'améliorer la catégorisation des vidéos. Donc, au lieu de couper la tranche normale du monde visuel nous avons coupé une plus grande part où nous incluons des informations supplémentaires à différents niveaux du pipeline de détection B.2. Notez que dans les termes de ce monde visuel, la fusion des classificateur peut être considérée comme combinant diverses tranches.

B.2.5 Nos Contributions

Nos contributions visent deux grands domaines du pipeline de détection des concepts: le codage et de pooling des caractéristiques visuelles, et l'apprentissage du modèle où nous nous concentrons sur la recherche d'exemples de formation utiles. Les lignes qui suivent résument nos contributions.

- Nous utilisons des caractéristiques extraites des images et travaillons sur leur transformation dans la représentation de niveau intermédiaire. Nous utilisons les informations

de l'étiquette lors de la construction de cette représentation pour ajouter des informations spécifique de l'image et du concept dans le trait finale.

- Nous améliorons aussi la représentation standard d'image en codant les caractéristiques d'informations spécifiques de transformation.
- Pour améliorer les performances de classification pour un concept nous explorons d'autres exemples de concepts connexes, afin d'ajouter des informations utiles et diversifiée. L'idée est que les concepts communs ou proches partagent des informations entre eux et nous pouvons utiliser cette information lors de l'apprentissage des modèles de classification.
- Nous étendons l'idée ci-dessus pour regarder concepts qui ne sont pas si proche, plutôt que sont différentes, afin d'apprendre de la dissemblance entre les concepts.
- Enfin, nous essayons d'explorer les exemples non étiquetés dans une manière d'entraînement semi- supervisée. Nous augmentons la quantité de données annotées et utilisons les meilleures informations sur elles pour apprendre les modèles des concepts.

B.3 Nos Contributions

Compte tenu des propriétés et des éléments essentiels des systèmes de détection des concepts dans les vidéos, dans cette thèse, nous proposons un ensemble de nouvelles contributions pour augmenter la taille de la tranche coupée du monde visuel. Nous travaillons aux differentes étapes du pipeline de détection avec l'objectif d'améliorer l'analyse visuelle globale et la précision de détection.

B.3.1 Pour une meilleure représentation de la vidéo

Nous commençons nos contributions avec le chapitre 3 dans lequel nous essayons d'améliorer les **modèles de représentation** de vidéo standard qui se compose de la deuxième étape du pipeline de catégorisation des vidéo présenté dans l'introduction. Plus précisément, nous améliorons le Dictionnaire Visuel ou le modèle de sac de mots **Bag of Words (BOW)** [8, 20], qui attribue des points-clés (descripteurs) aux clusters spécifiques et regroupe l'ensemble des descripteurs assignés à la même cellule de clustering.

La description de l'image dans le cadre de BOW est généralement confrontée à **deux** questions importantes. Premiere est la sélection de la **taille du dictionnaire** approprié et la seconde est la **perte d'informations** comme les caractéristiques de toutes les images dans la base d'entraînement sont considérées égales pour la construction du dictionnaire. La taille du dictionnaire est directement liée à la performance de catégorisation ainsi que de l'efficacité de recherche. Un modèle de BOW plus grand est plus *discriminant* mais l'efficacité d'entraînement de recherche du système est compromise avec un plus grand modèle. En

outre, lorsque la construction du modèle à partir des descripteurs nous n'incluons pas les informations sur la catégorie de l'image. Principalement dans le chapitre 3 nous proposons deux approches afin de rendre une représentation de BOW plus discriminantes et également plus courte.

Notre première proposition [125] aborde les deux problèmes viennent d'être présentés par la construction d'un relativement petit dictionnaire d'une plus grande en utilisant de l'information de la catégorie. Puisque la construction du dictionnaire réalisée par classification non supervisée ne prend pas avantage de la relation entre les descripteurs qui viennent à partir d'images appartenant aux concepts similaire, cela agrandit l'écart sémantique. Nous utilisons explicitement les informations sur les *étiquettes* d'image pour construire un modèle de BOW supervisé en fusionnant des clusters voisins en utilisant le critère de minimisation de la perte de l'information.

Nous améliorons également le modèle de BOW en codant la *position* de chacun des descripteurs à l'intérieur de la cellule de cluster quantifié en fonction de son centroide [126]. Ici un nouveau petit BOW est construit par la quantification des différences entre les descripteurs et de leurs centroides de clusters. L'intégration de ce nouveau petit descripteur, que nous apellons **DBOW** avec l'original pour représenter des images augmente la précision de la détection de concept dans les vidéos.

En testant les méthodes nous avons vu que la capacité discriminative du modèle de sac de mots (BOW) augmente lors qu'on le construit utilisant la méthode de fusion supervisé proposé. La performance d'un dictionnaire beaucoup plus petit obtenu par fusion supervisé atteint la performance de détection des concepts avec un dictionnaire grand obtenu par k-means. L'introduction d'un nouveau descripteur DBOW basé sur la quantification des différences également rend un petit modèle de la représentation des vidéos plus discriminant.

B.3.2 S'appuyant sur des données multi-étiquetés

Dans le chapitre 4 nous travaillons sur un nouveau paradigme d'apprentissage dans lequel nous essayons de combiner les concepts ou les étiquettes et d'apprendre ensemble. Dans ce paradigme tous les concepts font partie de certains groupes ou des méta-concepts. Les classificateurs sont formés pour ces nouveaux groupes ou des méta-concepts et les résultats pour chaque concept sont en quelque sorte dérivés.

Les Images et vidéos typiques sont multi-étiquettes dans la nature en ce que les classes (concepts) ne sont pas mutuellement exclusives. Le contenu visuel est très riche et comprend habituellement plusieurs objets ou concepts, dans un sens plus large, en une seule image ou une image-clé de vidéo. Une image de *Skyline* d'une ville, par exemple, contient de nombreux objets et une vidéo est généralement étiqueté avec plus d'un des concepts sémantiques. Nous utilisons cette multi-étiquetage intrinsèque avec un double objectif; **(i)** d'augmenter les ressources dans la base d'entraînement pour chaque étiquette (catégorie) et **(ii)** d'augmenter

le nombre de classificateurs par étiquette qui en fin de compte être combinés pour essayer de maximiser la performance de détection.

Considérant les concepts ensemble signifie que si elles sont très similaires, ils doivent être entraînés ensemble pour augmenter les ressources de formation et en même temps ils doivent être classés les uns contre les autres afin de mettre en évidence leurs différences. Un exemple pourrait être de fusionner les exemples de *la voiture* et *la route* dans un ensemble pour en entraîner un classificateur *groupe* qui soit fort et en même temps organiser au moins un de ceux à tomber dans un autre *groupe* de sorte qu'ils peuvent être distingués les uns contre les autres pour les cas où la voiture est dans le garage par exemple, ou la route est vide ou est bondé que par des bus. Nous y proposons deux différents types d'approches. La **première** groupe les concepts ou les class qui sont similaires et la **deuxième** fait les groupe opposant avec les concepts qui sont différent. Ici pour le deuxième type les concepts dans les mêmes groupes doit être similaire.

Nous proposons **group-SVM** [139], où nous devisons un critère pour grouper intelligemment les concepts basés sur la similarité entre eux. La similarité entre les concepts peut être calculée de différentes façons, y compris la distance entre les descripteurs, des informations de web sémantique, les règles de l'ontologie ou des relations prédéfinies entre les concepts. Nous ensuite essayons de regarder la similarité ainsi que les différences entre les concepts et les divisons dans des partitions [140]. Un classificateur est puis développé utilisant les exemples des deux partitions opposées. Nous regardons ce deuxième type d'approche dans le cadre du **Error Correcting Output Codes (ECOC)** où un ensemble de classificateurs sous-optimales peut atteindre les performances d'un classificateur complexe. Nous proposons un décodage de l'ECOC où nous pondérons la contribution de chaque classificateur basé sur la dynamique d'ECOC. Nous ensuite créons les ensembles d'ECOC afin de faire des prédictions plus fort.

En réalisation du test, nous avons vu comment le partage explicite des étiquettes peut bénéficier de la détection de concept dans les vidéos, en particulier pour les concepts avec très peu d'exemples positifs. Nous avons vu également que les techniques sont complémentaires à la classification simple binaire quand nous avons fait la fusion tardive des classificateurs.

B.3.3 Classification semi supervisé

Dans cette thèse, nous nous efforçons d'améliorer la détection des concepts dans les vidéo et nous aspirons à obtenir un coup de pouce à la performance en puisant dans les vastes ressources de données non étiquetées, comme ce serait le cas pour n'importe quel type de classification. Jusqu'à ce point, nous n'avons considéré que les exemples annotés comme des instances positives et négatives pour un concept. Nous avons également essayé avec des exemples empruntés d'autres concepts qui sont similaire ou connexe et le recul de concepts qui sont différents, mais dans ces cas, les exemples d'autres concepts ont été également

annotés. Comme l'a souligné tout au long de la thèse, il ya toujours besoin de plus de données annotées pour tenter de capturer les variations avec laquelle un concept sémantique se produit. Spécifiquement pour les jeux de données TRECVID que nous avons utilisé, il ya toujours un manque d'exemples positifs pour la plupart des concepts, dont un nombre important qui est essentiel pour la performance de détection de concept. Puisque le corpus en question est annoté en partie nous pouvons toujours essayer de trouver des **exemples utiles** à ajouter à la base d'entraînement d'un concept pour une meilleure classification sur les exemples de test.

Une méthode raisonnable pour trouver de nouvelles instances d'entraînement est de former un classificateur sur l'ensemble des exemples dans la base d'entraînement dispensée et d'utiliser ce modèle pour trouver les nouveaux exemples. Ce sont des exemples les plus confiants à la sortie du classificateur dans l'ensemble sans étiquette et ils sont ajoutés à la base d'entraînement pour ré-entraîner le classificateur. Cette **selftraining** boucle peut être répétée jusqu'à ce que la performance sur un ensemble de validation augmente. Bien qu'il ait travaillé avant [105], le problème avec cette forme d'apprentissage incrémental est le manque potentiel de variabilité dans les nouveaux exemples ajoutés. Les exemples les plus confiants ajoutés aux instances positives ont très peu d'effet sur la nouvelle frontière de décision. En outre, il peut améliorer sur l'ensemble de validation, mais cela pourrait conduire à un sur-ajustement, aucune information diverse est ajouté. Quand on trait des vidéos nous avons la liberté d'avoir une variété de représentations d'une seule image-clé, nous pouvons donc extraire un certain nombre de caractéristiques différentes.

Cotraining est un cas particulier d'apprentissage semi supervisé, qui ajoute d'exemples non étiquetés, et est utile lorsque les différentes représentations de données sont disponibles [191]. Comme son nom l'indique deux classificateurs formés sur chaque représentation de données sont utilisées à l'unisson pour étiqueter les données d'entraînement pour l'autre. Cela se fait de manière itérative jusqu'à ce que l'erreur de l'entraînement est suffisamment réduite ou toutes les données ont été étiquetées. La catégorisation des vidéos est une tâche difficile et les classificateurs actuels (les vus de cotraining) ne sont pas suffisantes [106, 107, 193–196], c'est à dire les prédictions les mieux classés contiennent une abondance de mal-classifications.

Nos propositions dans le chapitre 5 ajoutent des informations utiles à la base d'entraînement d'un classificateur basé sur le principe de cotraining à but pour augmenter les ressources d'entraînement annotés [109]. Ce résultat est obtenu en ajoutant des exemples positifs qui devraient (ou assurer dans une certaine mesure) pour améliorer le résultat de la classification finale. Nous proposons **deux méthodes** pour sélectionner la vue la plus complémentaire entre certain possibilités disponibles en fonction des statistiques calculées sur la base de validation. Nous ajoutons k nouvelles annotations positives pour entraîner le classificateur cible à l'aide de la vue sélectionnée. Pour la première méthode k est fixe tandis que pour le prochain k est trouvé automatiquement par descripteur (vue).

Nous avons démontré utilisant une base de test l'efficacité de la sélection de la vue approprié à ré-étiqueter un apprenant cible pour cotraining lorsque différentes options sont disponibles. Pour chaque descripteur le meilleur descripteur complémentaire est sélectionné et le nombre d'exemples d'étiquette est automatiquement sélectionné qui devraient être correcte jusqu'à un certain pourcentage.

B.4 Conclusions et Perspectives d'Avenirs

Tout au long de cette thèse, nous avons examiné les moyens d'améliorer la détection de concept de la vidéo et développé des méthodes pour rendre la machine à comprendre et de mieux reconnaître le contenu visuel. Nous avons mis au point différentes méthodes pour rendre la tranche plus grande en intégrant des informations supplémentaires qui est à la fois important et utile du monde visuel lors de la construction des classificateurs pour la détection de concept de vidéo. Toutefois, le problème de la reconnaissance automatique du contenu des vidéos est une question difficile et est encore loin d'être résolu. Le nombre de catégories possibles est illimité qui entraîne un accroissement incessant des bases de données vidéos et images. En outre essayer d'accueillir toutes les variations d'un objet qui peut être présent dans une catégorie rend le problème encore plus difficile. Néanmoins la recherche progresse lentement et la vision de la machine s'améliore chaque année [12, 17, 18, 25, 26, 187].

B.4.1 Les contributions clés

Dans ce manuscrit, nous avons apporté plusieurs contributions à résoudre le problème de la détection automatique des concepts dans des vidéos sur Internet. Nous avons étudié de près les étapes du pipeline de détection de concepts allant de l'extraction d'informations de bas niveau à partir des images brutes à la décision de haut niveau sur la présence du concept sémantique dans l'image de vidéo. Après un examen attentif et en raison de son importance dans l'état de l'art, nous avons choisi comme référence la tâche d'analyse vidéo, le TRECVID, et avons présenté l'expérimentation détaillée de nos méthodes proposées pour la tâche d'indexation sémantique. Pour conclure, nous réitérons d'abord nos principaux apports en quelques paragraphes ci-dessous.

Améliorer la représentation de la vidéo

Nous avons utilisé les principes de perte de l'information pour construire le Dictionnaire Visuel de manière supervisée. Après une analyse détaillée du modèle de BOW nous croyons que l'intégration des informations de catégorie tout en construisant la représentation de BOW fait en quelque sorte pour l'écart sémantique en incluant explicitement la sémantique dans le dictionnaire. Bien que cela puisse conduire à sur-apprentissage par la construction de dictionnaires qui sont spécifiques à un concept, nous avons observé une amélioration dans la performance lors de l'utilisation des informations sur la distribution de tous les concepts pour construire le dictionnaire.

Ajout de raffinement à la Représentation de la vidéo

L'état de l'art nous dit que la représentation de l'histogramme désordonnée (BOW) ne tient pas compte des détails sur l'emplacement d'un point-clé à l'intérieur de la cellule de clustering de BOW. Cela a certainement attiré une certaine attention de la recherche et, par conséquent, nous avons vu des méthodes proposant d'inclure des informations spécifiques géométrique, spatiale et le point-clé que l'amélioration du pouvoir discriminant du modèle de BOW. Nous avons également porté notre attention sur ce domaine et produit une méthode pour ajouter des informations de localisation précises d'un descripteur dans la représentation de BOW. L'information supplémentaire est fusionnée dans le vecteur de BOW en termes d'un très petit vecteur qui capture la différence de chaque point-clé de son centre de cluster qui a également abouti à une meilleure capacité de discrimination du modèle de BOW par rapport à l'état de l'art.

Utilisation les données multi-étiqueté: considérant la similarité entre concepts

Cette contribution a contesté la pratique normale lors de la construction des modèles de classification des concepts où les données annotée sont utilisé pour construire le modèle pour ce concept. Nous avons trouvé concepts *similaires* qui sont censés partager certaines propriétés visuelles, regroupé leurs instances ensemble et formé les classificateurs sur l'ensemble combiné des exemples. Nous avons démontré que la performance de ces classificateurs partagés atteint la performance des classificateurs de concept seule et aussi que ce partage apport des informations complémentaires lorsqu'il est fusionné avec l'apprentissage du concept seul (l'apprentissage binaire). Nous sommes d'accord que la réalisation n'était pas aussi brillante que l'idée, mais il a été marqué par de nombreux facteurs, notamment la qualité des annotations accumulées, la méthode choisie pour trouver la similitude, le faible nombre de concepts globale et la complexité accrue du classificateur non-linéaire de groupe. Cependant, notre critère *intelligente* de regroupement s'est avéré être mieux que le regroupement aléatoire de concepts et le regroupement de RAKEL pour la détection de concept de vidéo TRECVID.

Utilisation des données multi-étiquetés: considérant la similarité et la dissemblance entre les concepts

Ici, nous avons intégré *dissemblance* en plus de la similarité lors du regroupement (plutôt partitionnement) l'ensemble des concepts et observé que les classificateurs partagés correspondent et parfois dépassent la performance d'apprentissage de concepts seule avec les classificateurs qui ont été formés à partir des données d'entraînement beaucoup moins que la norme. Le nombre total de classificateurs formés étaient également moins du *baseline* qui est important compte tenu de la complexité des méthodes d'apprentissage artificiel.

Utilisation des données sans étiquette: la sélection de vue pour le Cotraining

Enfin, nous avons porté notre attention sur l'immense quantité de données vidéo non étiqueté qui est facilement disponible partout sur l'internet et se compose d'une grande partie des base de donnés vidéos comme TRECVID qui sont partiellement annotées. Nous avons étudié les méthodes d'apprentissage semi- supervisé et regardé la méthode astucieuse de cotraining

d'améliorer itérativement classificateurs utilisant des données non marquées, mais a réalisé la triste réalité de sa déception pour les méthodes d'analyse visuelle, en particulier la détection du concept dans vidéos. Avec l'évolution progressive du domaine en termes de performances et d'évolutivité, l'apprentissage semi-supervisé est également gagne en importance car il nous soulage de la main pour soigneusement étiqueter chaque image-clé de vidéos non étiquetés. Nous estimons qu'il est important de trouver le plus utile de l'information entre l'inconnu et le mettre à bon usage. À cette fin, nous avons développé des méthodes pour sélectionner les vue de cotraining le plus complémentaire pour augmenter la taille de l'ensemble d'entraînement d'un classificateur avec les exemples les plus constructifs et instructifs.

B.4.2 Des directions prometteuses

Nous divisons les perspectives d'avenir dans le travail qui étend les contributions présentées dans cette thèse et des perspectives générales pour l'indexation et la recherche basé sur le conténu.

B.4.2.1 Les perspectives d'avenir immédiat

Dans le contexte de nos propositions, nous avons présenté des orientations futures à travers de la thèse, mais ici, nous tenons à souligner certains points qui pourraient conduire à un travail immédiat de l'avenir. S'adressant chapitre 3 la conception de noyau différent doit être envisagé pour le descripteur de DBOW comme il modélise l'information différent de l'histogramme de BOW. En variante, un efficace mécanisme de pondération [132] pour la DBOW pourrait être développé ainsi que de trouver la taille optimale de DBOW pour un dictionnaire visuel donné. Étant donné que chaque cellule de Voronoï de grande dimension a dynamiques distinctes une quantification sur la base de la cellule se traduirait par une représentation plus précise, mais à un coût plus élevé. Une construction DBOW différemment pour chaque concept est également une amélioration possible en faisant un histogramme DBOW séparément pour chaque concept en utilisant des descripteurs à partir d'images marquées seulement avec ce concept.

Une exigence qui se fait sentir tout au long de la thèse est la nécessité d'aller plus. Puisque le monde réel est beaucoup plus grand que d'une liste de dizaines de concepts, qu'est-ce qui se passerait si une grande base de données avec beaucoup plus de concepts et un ensemble d'exemples augmenté est utilisé? Considérant l'application directe des méthodes proposées dans le chapitre 4 nous croyons que le partage entre les classes aidera toujours et avec plus de classes présentes meilleurs choix de regroupement et de partitionnement sera disponible pour augmenter la *capacité d'apprendre* du système. La classification de groupe ne se traduira pas bien pour le plus grand base d'entraînement, mais les méthodes de partitionnement s'adaptent. Un exemple est d'utiliser la grande liste des concepts de la tâche d'indexation sémantique *full* de TRECVID [18] pour générer les partitions et de former les classificateurs,

puis calcul de la performance sur la tâche *light* avec moins de concepts. Comme le montre l'analyse de la section 4.4.2 augmenter la taille de la liste des concepts, la performance de label partitioning s'améliore. En outre, les sections 4.3 et 4.4 montrent que les méthodes basées sur le partitionnement de l'espace de l'étiquette utilisent généralement nettement moins d'exemples avec moins de classificateurs pour atteindre les performances obtenues par l'apprentissage de chaque concept séparément. Donc la présence de plus d'informations serait effectivement bénéficier du système en termes d'efficacité et de performance.

Poursuivant plus de travail dans l'apprentissage semi-supervisé est intéressant surtout pour trouver un nombre minimum de nouveaux exemples étiquetés qui seront les plus utiles. En voyant les résultats de cotraining sélectif et d'apprentissage dans le chapitre 5, nous avons vu que dans l'ensemble les différents vues ne conviennent d'un très petit degré sur la pertinence des exemples non-étiquetés. Pourtant, nous avons vu une amélioration des performances pour tous les vues de cotraining. Nous croyons que l'ajout d'une vue beaucoup plus forte (un bon descripteur avec un classificateur fort) dans le mécanisme de ce cotraining sélectif permettrait d'améliorer davantage les résultats en pointant les autres dans la bonne direction.

Ici nous devrons faire des explorations dans le domaine multimodales pour bénéficier de la complémentarité des donnés très variés. Dans la thèse nous étions focalisé sur l'utilisation des descripteurs visuelles and nous avons testé les méthodes de sélection de vue pour le cotraining que pour ces descripteur visuelles. Nous admettons qu'ici il manqué de la variété car les descripteurs on étés plus ou moins de mêmes type. Mais le problème avec des base de donnés de type TRECVID est que d'autre modalités (notamment l'audio) ne fonctionnent pas bien pour la catégorisation. Pour conclure, nous devons utiliser très différents types de descripteurs avec des classificateurs forts (audio ou visuel) que ceux utilisés pour tester plus les méthodes de sélection.

B.4.2.2 Les perspectives d'avenir générale

Il existe toujours un écart énorme dans la recherche scientifique publiée chaque année qui s'améliore également et l'acceptation des méthodes de l'état de l'art dans l'industrie ou des applications réelles. Ce décalage entre la recherche et son application dans la vie réelle a été ressenti par de nombreux chercheurs et diverses applications sont en train d'émerger. Quelques exemples populaires pour les systèmes de récupération à base d'image sont l'IKONA de l'INRIA [208], LIRE [209], pixolu, et le système de SHIATSU [210] pour l'indexation des vidéos en fonction d'images clés. L'importance du contenu ne peut pas être nié et il est temps que les moteurs de recherche industrielle remplacent recherche par mot clé en fonction de l'indexation de contenu sur la base ou au-moins l'inclure dans leur système. Google et le moteur de recherche russe Yandex a conduit le travail de pionnier et offrent la recherche en ligne l'image basée sur le contenu.

L'évolutivité des méthodes est également important de les endre portables pour être utilisé sur les nombreux appareils portables. Les smartphones et tablettes sont maintenant équipés de caméras à haute définition et du matériel sophistiqué avec une grande puissance de traitement. Les applications telles que la reconnaissance automatique des produits, etc. ont été allouées dans les magasins en ligne de différentes plates-formes.

Les travaux de recherche ont continué à évoluer et des résultats prometteurs sont publiés sur ensembles de données plus difficiles chaque année. Récemment, l'accent a été mis sur la de-corrélation de la base d'entraînement et celui de teste [154, 160, 161] (modèles d'entraînement qui ne sont pas spécifiques à l'ensemble de données d'entraînement). Cela signifie que l'ensemble de test est tirée d'une autre source qui n'est généralement pas le cas pour les bases de données utilisées dans la recherche scientifique de nos jours.

L'apprentissage *zero-shot* [152, 161, 177] gagne également une importance car il aborde le problème de l'annotation de nouveaux concepts émergents. Le contenu de l'Internet continue de croître et c'est le même cas pour la liste des concepts sémantiques. En outre, les tendances sociales doivent être prises en compte comme quelque chose qui est pertinent aujourd'hui ne peut être inscrit dans le bar d'un moteur de recherche pour les années à venir.

Récemment l'apprentissage en profondeur *deep-learning* [211] a repoussé les limites de la performance sur la compréhension de la sémantique pour la détection de concept à l'aide de réseaux de neurones à convolution [102, 212]. Ils ont la capacité à extraire automatiquement et concevoir des caractéristiques significatives des valeurs brutes de pixels des images. Néanmoins, les réseaux de croyances profondes sont difficiles à former, nécessitent une grande quantité de données d'entraînement et, dans certains cas, un matériel spécial. Cela peut être considéré comme un autre *peek* dans les algorithmes de classification comme les réseaux de neurones de retour dans les années 80, mais pour l'instant rien ne correspond à leur exactitude et performance.

Pour conclure, il est sûr de dire que nous nous dirigeons dans la bonne direction pour rendre nos machines soient en mesure de comprendre le contenu visuel. L'information est là et il augment de plus en plus et est aussi la puissance de traitement de nos ordinateurs. Il s'agit de l'utiliser correctement avec les bonnes méthodes. Nous avons également besoin d'élargir nos horizons en explorant d'autres indices qui peuvent être importants dans la compréhension du contenu. Donner un sens à l'interaction entre les entités sur l'Internet, l'activité sociale, modélisation de l'utilisateur pour obtenir un aperçu des connaissances et des intérêts de l'uploader de la matière [213, 214], la location de l'utilisateur et de l'intérêt mutuel entre les utilisateurs d'un services de partage de contenu sont quelques-uns des indices qui pourraient être exploitées pour faire plus de sens du monde visuel.

Bibliography

[1] Y. G. Jiang and C. W. Ngo. Visual word proximity and linguistics for semantic video indexing and near-duplicate retrieval. *Computer Vision and Image Understanding*, 113 (3):405–414, 2009.

[2] C. smith. 350m new photos each day, September 2013. URL http://www.businessinsider.com/facebook-350-million-photos-each-day-2013-9.

[3] flickr forum. General flickr stats, October 2013. URL https://www.flickr.com/help/forum/en-us/72157636817319655/.

[4] A. Hanjalic, R.L. Lagendijk, and J. Biemond. A new method for key frame based video content representation. *Image Databases and Multi-Media Search*, 1998.

[5] M. S. Lew, N. Sebe, C. Djeraba, and R. Jain. Content-based multimedia information retrieval: State of the art and challenges. *ACM Trans. Multimedia Comput. Commun. Appl.*, 2(1):1–19, 2006.

[6] L. Zhao, W. Qi, S. Li, S. Q. Yang, and H. J. Zhang. Key-frame extraction and shot retrieval using nearest feature line (nfl). In *Proceedings of the 2000 ACM Workshops on Multimedia*, MULTIMEDIA '00, pages 217–220, New York, NY, USA, 2000. ACM.

[7] Y. L. Boureau, F. Bach, Y. LeCun, and J. Ponce. Learning mid-level features for recognition. In *CVPR*, pages 2559–2566. IEEE, 2010.

[8] G. Csurka, C. R. Dance, L. Fan, J. Willamowski, and C. Bray. Visual categorization with bags of keypoints. In *ECCV*, 2004.

[9] Y. G. Jiang, C. W. Ngo, and J. Yang. Towards optimal bag-of-features for object categorization and semantic video retrieval. In *Proceedings of the 6th ACM International Conference on Image and Video Retrieval*, CIVR '07, pages 494–501, New York, NY, USA, 2007. ACM.

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

[11] F. Perronnin and C. R. Dance. Fisher kernels on visual vocabularies for image categorization. In *CVPR*, 2007.

[12] H Jégou, F. Perronnin, M. Douze, J. Sánchez, P. Perez, and C. Schmid. Aggregating local image descriptors into compact codes. *IEEE Trans. Pattern Anal. Mach. Intell.*, 34(9):1704–1716, September 2012.

[13] A. Ng and M. Jordan. On Discriminative vs. Generative Classifiers: A Comparison of Logistic Regression and Naive Bayes. In *Advances in Neural Information Processing Systems (NIPS)*, volume 14, 2001.

[14] A. Smeulders, M. Worring, S. Santini, A. Gupta, and R. Jain. Content-based image retrieval at the end of the early years. *IEEE Trans. Pattern Anal. Mach. Intell.*, 22 (12):1349–1380, December 2000.

[15] A. Hanjalic and L. Q. Xu. Affective video content representation and modeling. *IEEE Transactions on Multimedia*, 7(1):143–154, 2005.

[16] C. Snoek and M. Worring. Concept-based video retrieval. *Found. Trends Inf. Retr.*, 2 (4):215–322, April 2009.

[17] L. Zhang and Y. Rui. Image search—from thousands to billions in 20 years. *ACM Trans. Multimedia Comput. Commun. Appl.*, 9(1s):36:1–36:20, October 2013.

[18] A. F. Smeaton, P. Over, and W. Kraaij. Evaluation campaigns and trecvid. In *MIR '06*, pages 321–330, 2006.

[19] M. Flickner, H. Sawhney, W. Niblack, J. Ashley, Q. Huang, B. Dom, M. Gorkani, J. Hafner, D. Lee, D. Petkovic, D. Steele, and P. Yanker. Query by image and video content: The qbic system. *Computer*, 28(9):23–32, September 1995.

[20] J. Sivic and A. Zisserman. Video Google: A text retrieval approach to object matching in videos. In *ICCV*, 2003.

[21] A. Joly, C. Frélicot, and O. Buisson. Robust content-based video copy identification in a large reference database. In *CIVR*, pages 414–424, 2003.

[22] T. Quack, U. Mönich, L. Thiele, and B. S. Manjunath. Cortina: A system for large-scale, content-based web image retrieval. In *Proceedings of the 12th Annual ACM International Conference on Multimedia*, MULTIMEDIA '04, pages 508–511, New York, NY, USA, 2004. ACM.

[23] A. Torralba, R. Fergus, and Y. Weiss. Small codes and large image databases for recognition. In *CVPR*. IEEE Computer Society, 2008.

[24] M. Douze, H. Jégou, H. Sandhawalia, L. Amsaleg, and C. Schmid. Evaluation of gist descriptors for web-scale image search. In *Proceedings of the ACM International Conference on Image and Video Retrieval*, CIVR '09, pages 19:1–19:8, New York, NY, USA, 2009. ACM.

[25] H. Jegou, M. Douze, C. Schmid, and P. Pérez. Aggregating local descriptors into a compact image representation. In *CVPR*, pages 3304–3311. IEEE, 2010.

[26] X. J. Wang, L. Zhang, M. Liu, Y. Li, and W. Y. Ma. Arista - image search to annotation on billions of web photos. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 2987–2994, June 2010.

[27] Y. Wang, T. Mei, S. Gong, and X. S. Hua. Combining global, regional and contextual features for automatic image annotation. *Pattern Recogn.*, 42(2):259–266, February 2009.

[28] J. Han and K. K. Ma. Fuzzy color histogram and its use in color image retrieval. *Trans. Img. Proc.*, 11(8):944–952, August 2002.

[29] R. Chakravarti and X. Meng. A study of color histogram based image retrieval. In *Information Technology: New Generations, 2009. ITNG '09. Sixth International Conference on*, pages 1323–1328, April 2009.

[30] T. Gevers, J. Weijer, and H. Stokman. *Color image processing: methods and applications: color feature detection: an overview*, chapter 9, pages 203–226. CRC press, 2006.

[31] K. van de Sande, T. Gevers, and C. Snoek. Evaluating color descriptors for object and scene recognition. *IEEE Trans. Pattern Anal. Mach. Intell.*, 32(9):1582–1596, September 2010.

[32] J. Weijer, T. Gevers, and A. D. Bagdanov. Boosting color saliency in image feature detection. *IEEE TRANS. PATTERN ANALYSIS AND MACHINE INTELLIGENCE*, 28:150–156, 2005.

[33] J. Hays and A. Efros. im2gps: estimating geographic information from a single image. In *Proceedings of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2008.

[34] E. Chang, K. Goh, G. Sychay, and G. Wu. Cbsa: Content-based soft annotation for multimodal image retrieval using bayes point machines. *IEEE Transactions on Circuits and Systems for Video Technology*, 13:26–38, 2003.

[35] J. Li and J.Z. Wang. Automatic linguistic indexing of pictures by a statistical modeling approach. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 25(9): 1075–1088, Sept 2003.

[36] M. Stricker and M. Orengo. Similarity of color images. pages 381–392, 1995.

[37] J. Li and J. Z. Wang. Automatic linguistic indexing of pictures by a statistical modeling approach. *IEEE Trans. Pattern Anal. Mach. Intell.*, 25(9):1075–1088, September 2003.

[38] F. Mindru, T. Tuytelaars, L. V. Gool, and T. Moons. Moment invariants for recognition under changing viewpoint and illumination. *Comput. Vis. Image Underst.*, 94(1-3):3–27, April 2004.

[39] J. Wang and X. S. Hua. Interactive image search by color map. *ACM Trans. Intell. Syst. Technol.*, 3(1):12:1–12:23, October 2011.

[40] A. Oliva and A. Torralba. Modeling the shape of the scene: A holistic representation of the spatial envelope. *International Journal of Computer Vision*, 42:145–175, 2001.

[41] Y. Weiss, A. Torralba, and R. Fergus. Spectral hashing. In *NIPS*, pages 1753–1760. Curran Associates, Inc., 2008.

[42] J. Hays and A. Efros. Scene completion using millions of photographs. *ACM Transactions on Graphics (SIGGRAPH 2007)*, 26(3), 2007.

[43] H. Tamura, S. Mori, and T. Yamawaki. Textural features corresponding to visual perception. *Systems, Man and Cybernetics, IEEE Transactions on*, 8(6):460–473, June 1978.

[44] D. Martin, C. Fowlkes, D. Tal, and J. Malik. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *in Proc. 8th Int'l Conf. Computer Vision*, pages 416–423, 2001.

[45] G. H. Liu, L. Zhang, Y. K. Hou, Z. Y. Li, and J. Y. Yang. Image retrieval based on multi-texton histogram. *Pattern Recognition*, 43(7):2380 – 2389, 2010.

[46] T. Ahonen, A. Hadid, and M. Pietikäinen. Face recognition with local binary patterns. In *In Proc. of 9th Euro15 We*, pages 469–481, 2011.

[47] M. Subrahmanyam, R.P. Maheshwari, and R. Balasubramanian. Local maximum edge binary patterns: A new descriptor for image retrieval and object tracking. *Signal Processing*, 92(6):1467 – 1479, 2012.

[48] A. Sinha, S. Banerji, and C. Liu. Novel color gabor-lbp-phog (glp) descriptors for object and scene image classification. In *ICVGIP*, page 58, 2012.

[49] C. S. Won, D. K. Park, and S. J. Park. Efficient use of MPEG-7 edge histogram descriptor. *Electronics and Telecommunications Research Institute (ETRI) in Daejeon, South Korea*, 24(1):23–30, February 2002.

[50] M. Wang, X. H. Hua, Y. Song, X. Yuan, S. Li, and H. J. Zhang. Automatic video annotation by semi-supervised learning with kernel density estimation. In *Proceedings of the 14th Annual ACM International Conference on Multimedia*, MULTIMEDIA '06, pages 967–976, New York, NY, USA, 2006. ACM.

[51] T. Tuytelaars and K. Mikolajczyk. Local invariant feature detectors: A survey. *Foundations and Trends in Computer Graphics and Vision*, 3(3):177–280, 2007.

[52] F. F. Li and P. Perona. A bayesian hierarchical model for learning natural scene categories. In *Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Volume 2 - Volume 02*, CVPR '05, pages 524–531, Washington, DC, USA, 2005. IEEE Computer Society.

[53] J. Winn, A. Criminisi, and T. Minka. Object categorization by learned universal visual dictionary. In *ICCV '05*, pages 1800–1807, 2005.

[54] 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.

[55] F. Moosmann, B. Triggs, and F. Jurie. Fast discriminative visual codebooks using randomized clustering forests. In *NIPS*, 2006.

[56] A. Gordoa, J. A. Rodriguez-Serrano, F. Perronnin, and E. Valveny. Leveraging category-level labels for instance-level image retrieval. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 3045–3052, June 2012.

[57] J. Delhumeau, P. H. Gosselin, H. Jégou, and P. Pérez. Revisiting the vlad image representation. In *Proceedings of the 21st ACM International Conference on Multimedia*, MM '13, pages 653–656, New York, NY, USA, 2013. ACM.

[58] K. Mikolajczyk and C. Schmid. A performance evaluation of local descriptors. *IEEE Trans. Pattern Anal. Mach. Intell.*, 27(10):1615–1630, October 2005.

[59] J. Matas, O. Chum, M. Urban, and T. Pajdla. Robust wide baseline stereo from maximally stable extremal regions. In *BMVC*. British Machine Vision Association, 2002.

[60] T. Tuytelaars and L. Van Gool. Matching widely separated views based on affine invariant regions. *International Journal of Computer Vision*, 59(1):61–85, 2004.

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

[62] D. 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 vol.2, 1999.

[63] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vision*, 60:91–110, 2004.

[64] Y. Ke and R. Sukthankar. Pca-sift: A more distinctive representation for local image descriptors. In *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, CVPR'04, pages 506–513, Washington, DC, USA, 2004. IEEE Computer Society.

[65] M. Jain, R. Benmokhtar, H. Jegou, and P. Gros. Hamming embedding similarity-based image classification. In *ICMR*, 2012.

[66] R. Arandjelović and A. Zisserman. Three things everyone should know to improve object retrieval. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2012.

[67] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, pages 886–893 vol. 1, June 2005.

[68] Qiang Z., M. C. Yeh, K. T. Cheng, and S. Avidan. Fast human detection using a cascade of histograms of oriented gradients. In *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, volume 2, pages 1491–1498, 2006.

[69] P. Ott and M. Everingham. Implicit color segmentation features for pedestrian and object detection. In *Computer Vision, 2009 IEEE 12th International Conference on*, pages 723–730, Sept 2009.

[70] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part-based models. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 32(9):1627–1645, Sept 2010.

[71] X. Wang, T. X. Han, and S. Yan. An hog-lbp human detector with partial occlusion handling. In *ICCV*, pages 32–39. IEEE, 2009.

[72] F. Jing, M. Li, L. Zhang, H. J. Zhang, and B. Zhang. Learning in region-based image retrieval. In *Image and Video Retrieval*, pages 206–215. Springer, 2003.

[73] G. Schindler, M. Brown, and R. Szeliski. City-scale location recognition. In *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on*, pages 1–7. IEEE, 2007.

[74] J. Wang, J. Yang, K. Yu, F. Lv, T. S. Huang, and Y. Gong. Locality-constrained linear coding for image classification. In *CVPR*, pages 3360–3367. IEEE, 2010.

[75] K. Chatfield, V. S. Lempitsky, A. Vedaldi, and A. Zisserman. The devil is in the details: an evaluation of recent feature encoding methods. In *British Machine Vision Conference, BMVC 2011, Dundee, UK, August 29 - September 2, 2011. Proceedings*, pages 1–12. BMVA Press, 2011.

[76] J. C. van Gemert, C. J. Veenman, A. W. M. Smeulders, and J. M. Geusebroek. Visual word ambiguity. *IEEE TPAMI*, 32(7), 2010.

[77] S. A. Chatzichristofis, C. Iakovidou, Y. S. Boutalis, and O. Marques. Co.vi.wo.: Color visual words based on non-predefined size codebooks. *IEEE T. Cybernetics*, 43(1): 192–205, 2013.

[78] K. Grauman and T. Darrell. The pyramid match kernel: Discriminative classification with sets of image features. In *ICCV*, pages 1458–1465. IEEE Computer Society, 2005.

[79] J. Yang, K. Yu, Y. Gong, and T. S. Huang. Linear spatial pyramid matching using sparse coding for image classification. In *CVPR*, pages 1794–1801. IEEE, 2009.

[80] Y. L. Boureau, N. Roux, F. Bach, J. Ponce, and Y. LeCun. Ask the locals: Multi-way local pooling for image recognition. In *ICCV*, pages 2651–2658. IEEE, 2011.

[81] J. Yuan, Y. Wu, and M. Yang. Discovery of collocation patterns: from visual words to visual phrases. In *Computer Vision and Pattern Recognition, 2007. CVPR '07. IEEE Conference on*, pages 1–8, June 2007.

[82] Y. T. Zheng, S. Y. Neo, T. S. Chua, and Q. Tian. Toward a higher-level visual representation for object-based image retrieval. *The Visual Computer*, 25(1):13–23, 2009.

[83] D. Picard and P. H. Gosselin. Improving image similarity with vectors of locally aggregated tensors. In *ICIP*, pages 669–672. IEEE, 2011.

[84] X. Zhou, K. Yu, T. Zhang, T. S. Huang, and T. S. Huang. Image classification using super-vector coding of local image descriptors. In *ECCV (5)*, pages 141–154, 2010.

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

[86] J. Luo and A. E. Savakis. Indoor vs outdoor classification of consumer photographs using low-level and semantic features. In *ICIP (2)*, pages 745–748, 2001.

[87] Y. Mori, H. Takahashi, and R. Oka. Image-to-word transformation based on dividing and vector quantizing images with words. In *MISRM'99 First International Workshop on Multimedia Intelligent Storage and Retrieval Management*, 1999.

[88] J. Jeon, V. Lavrenko, and R. Manmatha. Automatic image annotation and retrieval using cross-media relevance models. In *Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Informaion Retrieval*, SIGIR '03, pages 119–126, New York, NY, USA, 2003. ACM.

[89] P. Huang, J. Bu, C. Chen, K. Liu, and G. Qiu. Improve image annotation by combining multiple models. In *SITIS*, pages 3–9. IEEE Computer Society, 2007.

[90] A. Bosch, A. Zisserman, and X. Muñoz. Scene classification using a hybrid generative/discriminative approach. *IEEE Trans. Pattern Anal. Mach. Intell.*, 30(4):712–727, April 2008.

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

[92] H. H. Permuter, J. M. Francos, and I. Jermyn. A study of gaussian mixture models of color and texture features for image classification and segmentation. *Pattern Recognition*, 39(4):695–706, 2006.

[93] R. Zhang, Z. Zhang, M Li, W. Y. Ma, and H. Zhang. A probabilistic semantic model for image annotation and multi-modal image retrieval. *Multimedia Syst.*, 12(1):27–33, 2006.

[94] D. M. Blei and M. I. Jordan. Modeling annotated data. In *Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Informaion Retrieval*, SIGIR '03, pages 127–134, New York, NY, USA, 2003. ACM.

[95] X. Zhang, Z. Li, L. Zhang, W. Y. Ma, and H. Y. Shum. Efficient indexing for large scale visual search. In *Computer Vision, 2009 IEEE 12th International Conference on*, pages 1103–1110, Sept 2009.

[96] N.K. Alham, Maozhen L., S. Hammoud, and Hao Q. Evaluating machine learning techniques for automatic image annotations. In *Fuzzy Systems and Knowledge Discovery, 2009. FSKD '09. Sixth International Conference on*, volume 7, pages 245–249, Aug 2009.

[97] U. Niaz, M. Redi, C. Tanase, G. Merialdo, B. Farinella, and Q. Li. EURECOM at TRECVID 2011: The light semantic indexing task. In *TRECVID 2011, 15th International Workshop on Video Retrieval Evaluation, 2011, National Institute of Standards and Technology*, Gaithersburg, USA, 11 2011.

[98] U. Niaz, M. Redi, C. Tanase, and B. Merialdo. EURECOM at TrecVid 2012: The light semantic indexing task. In *TRECVID 2012, 16th International Workshop on Video Retrieval Evaluation, October 29, 2012, National Institute of Standards and Technology*, Gaithersburg, USA, 10 2012.

[99] S. Maji, A. C. Berg, and J. 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, June 2008.

[100] P. Vincent and Y. Bengio. K-local hyperplane and convex distance nearest neighbor algorithms. In *NIPS*, pages 985–992. MIT Press, 2001.

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

[102] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems 25*, pages 1097–1105. Curran Associates, Inc., 2012.

[103] H. Zhang, A. C. Berg, M. Maire, and J. Malik. Svm-knn: Discriminative nearest neighbor classification for visual category recognition. In *Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Volume 2*, CVPR '06, pages 2126–2136, Washington, DC, USA, 2006. IEEE Computer Society.

[104] T. Tuytelaars, M. Fritz, K. Saenko, and T. Darrell. The nbnn kernel. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 1824–1831, Nov 2011.

[105] K. Nigam and R. Ghani. Analyzing the effectiveness and applicability of co-training. In *CIKM*, 2000.

[106] R. Yan and M. R. Naphade. Semi-supervised cross feature learning for semantic concept detection in videos. In *CVPR (1)*, pages 657–663. IEEE Computer Society, 2005.

[107] W. Li, L. Duan, I. Tsang, and D. Xu. Co-labeling: A new multi-view learning approach for ambiguous problems. In *ICDM*, pages 419–428. IEEE Computer Society, 2012.

[108] W. Du, R. Phlypo, and T. Adali. Adaptive feature split selection for co-training: Application to tire irregular wear classification. In *ICASSP*, 2013.

[109] U. Niaz and B. Merialdo. Selective multi-cotraining for video concept detection. In *ICMR 2014, 4th ACM International Conference on Multimedia Retrieval, April 1-4, 2014*, Glasgow, United Kingdom, 04 2014.

[110] P. K. Atrey, M. A. Hossain, A. El Saddik, and M. S. Kankanhalli. Multimodal fusion for multimedia analysis: A survey. *Multimedia Systems*, 16(6):345–379, 2010.

[111] C. Snoek and M. Worring. Multimodal video indexing: A review of the state-of-the-art. *Multimedia Tools and Applications*, 25:5–35, 2003.

[112] U. Niaz and B. Merialdo. Fusion methods for multimodal indexing of web data. In *WIAMIS 2013, 14th International Workshop on Image and Audio Analysis for Multimedia Interactive Sercices, July 3-5, 2013*, Paris, Frace, 07 2013.

[113] C. Snoek, M. Worring, and A. Smeulders. Early versus late fusion in semantic video analysis. ACM MM, 2005.

[114] S. T. Strat, A. Benoit, H. Bredin, G. Quénot, and P. Lambert. Hierarchical late fusion for concept detection in videos. ECCV, 2012.

[115] M. Law, N. Thome, and M. Cord. Hybrid pooling fusion in the bow pipeline. ECCV, 2012.

[116] M. Guillaumin, J. Verbeek, and C. Schmid. Multimodal semi-supervised learning for image classification. In *CVPR*, 2010.

[117] M. S. Kankanhalli, J. Wang, and R. Jain. Experiential sampling in multimedia systems. *Trans. Multi.*, 8(5):937–946, October 2006.

[118] Y. G. Jiang, G. Ye, S. F. Chang, D. Ellis, and A. C. Loui. Consumer video understanding: A benchmark database and an evaluation of human and machine performance. ICMR, 2011.

[119] X. Zou and B. Bhanu. Tracking humans using multi-modal fusion. CVPR, 2005.

[120] K. McDonald and A. F. Smeaton. A comparison of score, rank and probability-based fusion methods for video shot retrieval. CIVR, 2005.

[121] X. S. Hua and H. J. Zhang. An attention-based decision fusion scheme for multimedia information retrieval. PRCM, 2004.

[122] N. Liu, E. Dellandrea, C. Zhu, C. E. Bichot, and L. Chen. A selective weighted late fusion for visual concept recognition. ECCV, 2012.

[123] W. H. Adams, G. Iyengar, M. R. Naphade, C. Neti, H. J. Nock, and J. R. Smith. Semantic indexing of multimedia content using visual, audio and text cues. *EURASIP*, 2:170–185, 2003.

[124] R. Yan, J. Yang, and A. G. Hauptmann. Learning query-class dependent weights in automatic video retrieval. MULTIMEDIA '04, pages 548–555. ACM, 2004.

[125] U. Niaz and B. Mérialdo. Entropy based supervised merging for visual categorization. In *ACIVS 2012, Advanced Concepts for Intelligent Vision Systems, 4 September 2012, Brno University of Technology, Also published in LNCS, Volume 7517/2012, Springer*, Brno, Czech Republic, 09 2012.

[126] U. Niaz and B. Merialdo. Exploring intra-bow statistics for improving visual categorization. In *WIAMIS 2013, 14th International Workshop on Image and Audio Analysis for Multimedia Interactive Sercices, July 3-5, 2013*, Paris, France, 07 2013.

[127] L. Wang. Toward a discriminative codebook: Codeword selection across multi-resolution. In *CVPR*, 2007.

[128] F. Perronnin, C. R. Dance, G. Csurka, and M. Bressan. Adapted vocabularies for generic visual categorization. In *ECCV (4)*, pages 464–475, 2006.

[129] C. Lin, S. Li, and S. Su. Image classification using adapted codebook. In *ITIME '09*, volume 1, pages 1307 –1312, 2009.

[130] J. Hao and X. Jie. Improved bags-of-words algorithm for scene recognition. In *Signal Processing Systems (ICSPS), 2010 2nd International Conference on*, volume 2, pages V2–279 –V2–282, 2010.

[131] Y. Su and F. Jurie. Visual word disambiguation by semantic contexts. In *ICCV*, 2011.

[132] F. Wang and B. Merialdo. Weighting informativeness of bag-of-visual-words by kernel optimization for video concept detection. In *VLS-MCMR*, 2010.

[133] H. Jegou, M. Douze, and C. Schmid. Hamming embedding and weak geometric consistency for large scale image search. In *ECCV*, 2008.

[134] D. Arthur and S. Vassilvitskii. k-means++: the advantages of careful seeding. In *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*, SODA '07, pages 1027–1035, Philadelphia, PA, USA, 2007.

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

[136] C. C. Chang and C. J. Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27, 2011.

[137] Andrea Vedaldi and Andrew Zisserman. Efficient additive kernels via explicit feature maps. *IEEE Trans. Pattern Anal. Mach. Intell.*, 34(3):480–492, March 2012.

[138] A. Vedaldi and B. Fulkerson. VLFeat: An open and portable library of computer vision algorithms, 2008. URL http://www.vlfeat.org/.

[139] U. Niaz and B. Merialdo. Leveraging from group classification for video concept detection. In *CBMI 2013, 11th International Workshop on Content-Based Multimedia Indexing, June 17-19 2013*, Veszprem, Hungary, 06 2013.

[140] U. Niaz and B. Merialdo. Improving video concept detection through label space partitioning. In *ICME 2014, IEEE International Conference on Multimedia and Expo, 14-18 July 2014*, Chengdu, China, 07 2014.

[141] G. Tsoumakas and I. Katakis. Multi-label classification: An overview. *Int J Data Warehousing and Mining*, 2007:1–13, 2007.

[142] G. Tsoumakas, I. Katakis, and I. Vlahavas. A review of multi-label classification methods. *Proceedings of the 2nd ADBIS Workshop on Data Mining and Knowledge Discovery (ADMKD 2006)*, pages 99–109, 2006.

[143] G. Tsoumakas, I. Katakis, and I. P. Vlahavas. Mining multi-label data. In *Data Mining and Knowledge Discovery Handbook*, pages 667–685. Springer, 2010.

[144] G. Tsoumakas, I. Katakis, and I. P. Vlahavas. Random k-labelsets for multilabel classification. *IEEE Trans. Knowl. Data Eng.*, 23(7):1079–1089, 2011.

[145] G. Tsoumakas and I. Vlahavas. Random k-labelsets: An ensemble method for multilabel classification. In *Proceedings of the 18th European Conference on Machine Learning*, ECML '07, pages 406–417, Berlin, Heidelberg, 2007. Springer-Verlag.

[146] E. Bart and S. Ullman. Single-example learning of novel classes using representation by similarity. In *BMVC*. British Machine Vision Association, 2005.

[147] E. Bart and S. Ullman. Cross-generalization: Learning novel classes from a single example by feature replacement. In *CVPR (1)*, pages 672–679. IEEE Computer Society, 2005.

[148] R. Fergus, H. Bernal, Y. Weiss, and A. Torralba. Semantic label sharing for learning with many categories. In *ECCV (1)*, volume 6311 of *Lecture Notes in Computer Science*, pages 762–775. Springer, 2010.

[149] G. A. Miller. Wordnet: A lexical database for english. *Commun. ACM*, 38(11):39–41, November 1995.

[150] M. Rohrbach, M. Stark, G. Szarvas, I. Gurevych, and B. Schiele. What helps where - and why? semantic relatedness for knowledge transfer. In *CVPR*, pages 910–917. IEEE, 2010.

[151] V. Ferrari and A. Zisserman. Learning visual attributes. In *Advances in Neural Information Processing Systems*, December 2007.

[152] A. Farhadi, I. Endres, D. Hoiem, and D. Forsyth. Describing objects by their attributes. In *CVPR*, 2009.

[153] G. Wang and D. A. Forsyth. Joint learning of visual attributes, object classes and visual saliency. In *ICCV*, pages 537–544. IEEE, 2009.

[154] C. H. Lampert, H. Nickisch, and S. Harmeling. Learning to detect unseen object classes by betweenclass attribute transfer. In *CVPR*, 2009.

[155] G. Wang, D. Hoiem, and D. Forsyth. Building text features for object image classification. In *CVPR*, pages 1367–1374. IEEE, 2009.

[156] N. Kumar, A. C. Berg, P. N. Belhumeur, and S. K. Nayar. Attribute and simile classifiers for face verification. In *ICCV*, pages 365–372. IEEE, 2009.

[157] T. L. Berg, A. C. Berg, and J. Shih. Automatic attribute discovery and characterization from noisy web data. In *ECCV (1)*, volume 6311 of *Lecture Notes in Computer Science*, pages 663–676. Springer, 2010.

[158] D. Parikh and K. Grauman. Interactively building a discriminative vocabulary of nameable attributes. In *CVPR*, pages 1681–1688. IEEE, 2011.

[159] M. Rastegari, A. Farhadi, and D. Forsyth. Attribute discovery via predictable discriminative binary codes. In *ECCV*, pages 876–889, 2012.

[160] A. Farhadi, I. Endres, and D. Hoiem. Attribute-centric recognition for cross-category generalization. In *CVPR*, pages 2352–2359. IEEE, 2010.

[161] D. Parikh and K. Grauman. Relative attributes. In *ICCV*, pages 503–510. IEEE, 2011.

[162] Y. Wang and G. Mori. A discriminative latent model of object classes and attributes. In *ECCV (5)*, volume 6315 of *Lecture Notes in Computer Science*, pages 155–168. Springer, 2010.

[163] O. Russakovsky and F. F. Li. Attribute learning in large-scale datasets. In *ECCV Workshops (1)*, volume 6553 of *Lecture Notes in Computer Science*, pages 1–14. Springer, 2010.

[164] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR09*, 2009.

[165] J. Cai, Z. J. Zha, W. Zhou, and Q. Tian. Attribute-assisted reranking for web image retrieval. In *ACM Multimedia*, pages 873–876. ACM, 2012.

[166] Hanwang Zhang, Zheng-Jun Zha, Yang Yang, Shuicheng Yan, Yue Gao, and Tat-Seng Chua. Attribute-augmented semantic hierarchy: towards bridging semantic gap and intention gap in image retrieval. In *ACM Multimedia*, pages 33–42. ACM, 2013.

[167] F. Yu, R. Ji, M. H. Tsai, G. Ye, and F. F. Chang. Weak attributes for large-scale image retrieval. In *CVPR*, 2012.

[168] R. Salakhutdinov and G. E. Hinton. Learning a nonlinear embedding by preserving class neighbourhood structure. In *AISTATS*, volume 2 of *JMLR Proceedings*, pages 412–419. JMLR.org, 2007.

[169] R. R. Salakhutdinov and G. E. Hinton. Semantic hashing. In *Proceedings of the SIGIR Workshop on Information Retrieval and Applications of Graphical Models*, Amsterdam, 2007. Elsevier.

[170] M. Jain, H. Jégou, and P. Gros. Asymmetric hamming embedding: Taking the best of our bits for large scale image search. In *Proceedings of the 19th ACM International Conference on Multimedia*, MM '11, pages 1441–1444, New York, NY, USA, 2011. ACM.

[171] J. P. Heo, Y. Lee, J. He, S. F. Chang, and S. E. Yoon. Spherical hashing. In *CVPR*, pages 2957–2964, 2012.

[172] M. Raginsky and S. Lazebnik. Locality-sensitive binary codes from shift-invariant kernels. In *NIPS*, pages 1509–1517. Curran Associates, Inc., 2009.

[173] T. Hastie and R. Tibshirani. Classification by pairwise coupling. In *Proceedings of the 1997 Conference on Advances in Neural Information Processing Systems 10*, NIPS '97, pages 507–513, Cambridge, MA, USA, 1998. MIT Press.

[174] T. G. Dietterich and G. Bakiri. Solving multiclass learning problems via error-correcting output codes. *Journal of Artificial Intelligence Research*, 2:263–286, 1995.

[175] S. Escalera, O. Pujol, and P. Radeva. Loss-weighted decoding for error-correcting output coding. In *International Conference on Computer Vision Theory and Applications*, pages 117–122, 2008.

[176] G. Armano, C. Chira, and N. Hatami. Error-correcting output codes for multi-label text categorization. In *IIR*, CEUR, 2012.

[177] M. Cissé, T. Artières, and P. Gallinari. Learning compact class codes for fast inference in large multi class classification. In *ECML/PKDD (1)*, volume 7523 of *Lecture Notes in Computer Science*, pages 506–520. Springer, 2012.

[178] E. L. Allwein, R. E. Schapire, and Y. Singer. Reducing multiclass to binary: A unifying approach for margin classifiers. *J. Mach. Learn. Res.*, 1:113–141, September 2001.

[179] S. Escalera, O. Pujol, and P. Radeva. Separability of ternary codes for sparse designs of error-correcting output codes. *Pattern Recognition Letters*, 30(3), 2009.

[180] T. Kajdanowicz and P. Kazienko. Multi-label classification using error correcting output codes. *Applied Mathematics and Computer Science*, 22, 2012.

[181] M. Mirza-Mohammadi, F. Ciompi, S. Escalera, O. Pujol, and P. Radeva. Ranking error-correcting output codes for class retrieval. In *CVC-IIR*, 2009.

[182] C. S. Ferng and H. T. Lin. Multi-label classification with error-correcting codes. In *ACML*, volume 20 of *Journal of Machine Learning Research*, 2011.

[183] O. Pujol, S. Escalera, and P. Radeva. An incremental node embedding technique for error correcting output codes. *Pattern Recognition*, 41, 2008.

[184] Y. Wang and D. A. Forsyth. Large multi-class image categorization with ensembles of label trees. In *ICME*, 2013.

[185] A. Rocha and S. K. Goldenstein. Multiclass from binary: Expanding one-versus-all, one-versus-one and ecoc-based approaches. *IEEE Trans. Neural Netw. Learning Syst.*, 25(2):289–302, 2014.

[186] B. C. Russell, A. Torralba, K. P. Murphy, and W. T. Freeman. Labelme: A database and web-based tool for image annotation. *Int. J. Comput. Vision*, 77:157–173, May 2008.

[187] P. Over, G. Awad, M. Michel, J. Fiscus, G. Sanders, W. Kraaij, A. F. Smeaton, and G. Quéenot. Trecvid 2013 – an overview of the goals, tasks, data, evaluation mechanisms and metrics. In *Proceedings of TRECVID 2013*. NIST, USA, 2013.

[188] S. Shalev-Shwartz, Y. Singer, and N. Srebro. Pegasos: Primal estimated sub-gradient solver for svm. ICML, 2007.

[189] S. Bengio, J. Weston, and D. Grangier. Label embedding trees for large multi-class tasks. In *NIPS*, pages 163–171. Curran Associates, Inc., 2010.

[190] M. D. Smucker, J. Allan, and B. Carterette. A comparison of statistical significance tests for information retrieval evaluation. In *ACM-IKM*, 2007.

[191] A. Blum and T. Mitchell. Combining labeled and unlabeled data with co-training. In *COLT*, 1998.

[192] C. Xu, D. Tao, and C. Xu. A survey on multi-view learning. *CoRR*, abs/1304.5634, 2013.

[193] X. Zhang, J. Cheng, H. Lu, and S. Ma. Weighted co-svm for image retrieval with mvb strategy. In *ICIP (4)*, pages 517–520. IEEE, 2007.

[194] R. Ewerth and B. Freisleben. Semi-supervised learning for semantic video retrieval. In *Proceedings of the 6th ACM International Conference on Image and Video Retrieval*, CIVR '07, pages 154–161, New York, NY, USA, 2007. ACM.

[195] X. Zhang, J. Cheng, H. Lu, and S. Ma. Weighted co-svm for image retrieval with mvb strategy. In *ICIP (4)*, pages 517–520, 2007.

[196] C. M. Christoudias, R. Urtasun, A. Kapoor, and T. Darrell. Co-training with noisy perceptual observations. In *CVPR*, pages 2844–2851, 2009.

[197] R. Yan and M. R. Naphade. Co-training non-robust classifiers for video semantic concept detection. In *ICIP (1)*, pages 1205–1208. IEEE, 2005.

[198] P.Q. Gu, Qingsheng Zhu, and C. Zhang. A multi-view approach to semi-supervised document classification with incremental naive bayes. *Computers and Mathematics with Applications*, 57(6):1030–1036, 2009.

[199] Y. Du, X. Guan, and Z. Cai. Enhancing web page classification via local co-training. In *Proceedings of the 2010 20th International Conference on Pattern Recognition*, ICPR '10, pages 2905–2908, Washington, DC, USA, 2010. IEEE Computer Society.

[200] Y. Yaslan and Z. Cataltepe. Random relevant and non-redundant feature subspaces for co-training. In *IDEAL*, volume 5788, 2009.

[201] G. Z. Li, D. Li, W. C. Lu, J. Y. Yang, and M. Q. Yang. Feature selection for co-training: A qsar study. In *IC-AI*, 2007.

[202] J. Cheng and K. Wang. Active learning for image retrieval with co-svm. *Pattern Recognition*, 40(1):330 – 334, 2007.

[203] I. Muslea, C. Kloblock, and S. Minton. Adaptive view validation: A first step towards automatic view detection. pages 443–450, 2002.

[204] U. Güz, S. Cuendet, D. Hakkani-Tür, and G. Tür. Multi-view semi-supervised learning for dialog act segmentation of speech. *IEEE Transactions on Audio, Speech and Language Processing*, 18(2):320–329, 2010.

[205] X. Li and C. Snoek. Classifying tag relevance with relevant positive and negative examples. In *ACM International Conference on Multimedia*, 2013.

[206] K. van de Sande, T. Gevers, and C. Snoek. Empowering visual categorization with the gpu. *IEEE Transactions on Multimedia*, 13(1), 2011.

[207] Yangqing J. Caffe: An open source convolutional architecture for fast feature embedding. http://caffe.berkeleyvision.org, 2013.

[208] Ikona project web demo. URL https://www.rocq.inria.fr/cgi-bin/imedia/circario.cgi/demos.

[209] M. Lux. Content based image retrieval with lire. In *Proceedings of the 19th ACM International Conference on Multimedia*, MM '11, pages 735–738, New York, NY, USA, 2011. ACM.

[210] I. Bartolini, M. Patella, and C. Romani. Shiatsu: tagging and retrieving videos without worries. *Multimedia Tools and Applications*, 63(2):357–385, 2013.

[211] J. Ngiam, A. Khosla, M. Kim, J. Nam, H. Lee, and A. Y. Ng. Multimodal deep learning. In *International Conference on Machine Learning (ICML)*, Bellevue, USA, June 2011.

[212] R. Socher and B. Huval and B. Bhat and C. D. Manning and A. Y. Ng. Convolutional-Recursive Deep Learning for 3D Object Classification. In *Advances in Neural Information Processing Systems 25*. 2012.

[213] U. Niaz and B. Merialdo. Improving video concept detection using uploader model. In *ICME 2013, IEEE International Conference on Multimedia and Expo, July 15-19, 2013*, San Jose, USA, 07 2013.

[214] U. Merialdo, B. Niaz. Uploader models for video concept detection. In *CBMI 2014, 12th International Workshop on Content-Based Multimedia Indexing, 18-20 June 2014*, Klagenfurt, Austria, 06 2014.