IMAGE ANNOTATION WITH SEMI-SUPERVISED CLUSTERING

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

AHMET SAYAR

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF DOCTOR OF PHILOSOPHY
IN
COMPUTER ENGINEERING

DECEMBER 2009

Approval of the thesis:

IMAGE ANNOTATION WITH SEMI-SUPERVISED CLUSTERING

submitted by **AHMET SAYAR** in partial fulfillment of the requirements for the degree of **Doctor of Philosophy in Computer Engineering Department, Middle East Technical University** by,

Prof. Dr. Canan Özgen
Dean, Graduate School of **Natural and Applied Sciences** _____

Prof. Dr. Müslim Bozyiğit
Head of Department, **Computer Engineering** _____

Prof. Dr. Fatoş Tünay Yarman Vural
Supervisor, **Computer Engineering Department, METU** _____

Examining Committee Members:

Prof. Dr. Faruk Polat
Computer Engineering, METU _____

Prof. Dr. Fatoş Tünay Yarman Vural
Computer Engineering, METU _____

Prof. Dr. Gözde Bozdağı Akar
Electrical and Electronics Engineering, METU _____

Assist. Prof. Dr. İlkay Ulusoy
Electrical and Electronics Engineering, METU _____

Assist. Prof. Dr. Pınar Duygulu Şahin
Computer Engineering, Bilkent University _____

Date: _____

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last Name: AHMET SAYAR

Signature :

ABSTRACT

IMAGE ANNOTATION WITH SEMI-SUPERVISED CLUSTERING

Sayar, Ahmet
M.S., Department of Computer Engineering
Supervisor : Prof. Dr. Fatoş Tünay Yarman Vural

December 2009, 144 pages

Image annotation is defined as generating a set of textual words for a given image, learning from the available training data consisting of visual image content and annotation words.

Methods developed for image annotation usually make use of region clustering algorithms to quantize the visual information. Visual codebooks are generated from the region clusters of low level visual features. These codebooks are then, matched with the words of the text document related to the image, in various ways.

In this thesis, we propose a new image annotation technique, which improves the representation and quantization of the visual information by employing the available but unused information, called side information, which is hidden in the system. This side information is used to semi-supervise the clustering process which creates the visterms. The selection of side information depends on the visual image content, the annotation words and the relationship between them. Although there may be many different ways of defining and selecting side information, in this thesis, three types of side information are proposed. The first one is the hidden topic probability information obtained automatically from the text document associated with the image. The second one is the orientation and the third one is the color

information around interest points that correspond to critical locations in the image. The side information provides a set of constraints in a semi-supervised K-means region clustering algorithm. Consequently, in generation of the visual terms from the regions, not only low level features are clustered, but also side information is used to complement the visual information, called visterms. This complementary information is expected to close the semantic gap between the low level features extracted from each region and the high level textual information. Therefore, a better match between visual codebook and the annotation words is obtained. Moreover, a speedup is obtained in the modified K-means algorithm because of the constraints brought by the side information. The proposed algorithm is implemented in a high performance parallel computation environment.

Keywords: image annotation, semi-supervised clustering, K-means, SIFT, MPI, visterm, document

ÖZ

YARI DENETİMLİ KÜMELEME İLE GÖRÜNTÜ ETİKETLEME

Sayar, Ahmet
Yüksek Lisans, Bilgisayar Mühendisliği Bölümü
Tez Yöneticisi : Prof. Dr. Fatoş Tünay Yarman Vural

Aralık 2009, 144 sayfa

Görüntü etiketleme, mevcut etiketlenmiş görüntü eğitim kümelerinden öğrenerek, verilen bir resim için bir dizi kelime üretilmesi olarak tanımlanabilir.

Otomatik görüntü etiketleme yöntemlerinde görsel bilgiyi nicelemek için genelde bölge kümeleme algoritmaları kullanılmaktadır. Görsel kod tabloları, bölgelerden elde edilen düşük düzeyli görsel özniteliklerin kümelenmesiyle elde edilir. Bu kod tabloları görüntü etiketleriyle değişik yöntemler kullanılarak eşleştirilmektedir.

Bu tezde, etiketlenmiş görüntülerde mevcut ancak, kullanılmayan bilgileri kullanarak kümeleme işlemini iyileştiren yeni bir görüntü etiketleme tekniği önerilmektedir. ”Ek bilgi” adı verilen bazı öznitelikler kümeleme işlemini denetlemek için kullanılmaktadır. Bu tezde, üç tip ek bilgi önerilmektedir. İlki, görüntü etiketlerini kapsayan metin dokümanınından otomatik olarak elde edilen gizli konu olasılıkları bilgisidir. Diğer ikisi görüntünün önemli yerlerini işaret eden ilgi noktaları etrafından elde edilen yön ve renk bilgileridir. Bu ek bilgiler, yarı denetimli k-ortalama bölge kümeleme algoritmasına bir dizi kısıt sağlamak amacı ile değerlendirilirler. Böylece, bölgelerin kümelemesinde sadece düşük seviyeli görsel öznitelik-

ler değil, aynı zamanda bu ek bilgiler de kullanılmış olur. Bu tamamlayıcı ek bilginin görüntü bölgelerinden elde edilen düşük seviyeli öznitelikler ile yüksek seviyeli metin bilgisi arasına anlambilimsel açığı kapatması beklenir.

Sonuç olarak, görsel kod tabloları ve görüntü etiket kelimeleri arasında daha iyi bir ilişki elde edilmiş olur. Ayrıca, uyarlanan K-ortalama algoritmasında kullanılan kısıtlar nedeniyle algoritma performansında hızlanma sağlanmıştır. Önerilen algoritma yüksek performanslı paralel hesaplama ortamında gerçeklenmiştir.

Anahtar Kelimeler: görüntü etiketleme, yarı-denetimli kümeleme, K-ortalama, SIFT, MPI, görsel terim, doküman

To my wife and son

ACKNOWLEDGMENTS

This thesis would not have been possible without a number of people.

Firstly, I would like to thank my supervisor, Prof. Fatoş Tünay Yarman Vural for her support, motivation and encouragement throughout my studies. I have learned a lot from her both academically and intellectually.

I would also like to thank other committee members, Prof. Faruk Polat, and Dr. Pınar Duygulu. They have provided a lot of helpful suggestions and fruitful discussions.

Thanks to Image Processing and Pattern Recognition Laboratory current and former members for helpful discussions and friendly meetings on weekends.

I am also thankful to Florent Monay for answering questions through e-mail and Kobus Barnard for providing the dataset.

And finally, I thank to my parents, my wife and son for their love and support. Thanks to my wife and son for their patience, support and tolerance for times, I had to spend away from them to finish this thesis.

This research was supported by National High Performance Computing Center of Istanbul Technical University under grant number 10182007. This research was supported in part by TUBITAK through TR-Grid e-Infrastructure Project. TR-Grid systems are hosted by TUBITAK ULAKBIM, Middle East Technical University, Pamukkale University, Cukurova University, Erciyes University, Bogazici University and Istanbul Technical University. Visit http://www.grid.org.tr for more information.

TABLE OF CONTENTS

LIST OF TABLES

TABLES

LIST OF FIGURES

FIGURES

CHAPTER 1

INTRODUCTION

With the recent developments in digital image acquisition and storage technologies, amount of collections that carry images continue to increase; World Wide Web leading the way. Managing large amount of such collections is an important task that requires searching these collections with high accuracy and efficiency. An intuitive way of searching through these collections is the Query-By-Example (QBE) method, which is also known as Content Based Image Retrieval (CBIR). This method has been the subject of considerable amount of research in the last decade, surveys of which can be found in [1] and [2]. In CBIR, a sample image is given as a query, and the retrieval engine is expected to find the most resembling image(s) in the collection based on visual content of the query image. Although implemented in the early image retrieval systems [3], [4], [5], [6], [7], [8], [9], [10]; this method did not find its way in recent retrieval architectures. The reason for this result is two-folded. First, it is not easy for users to find sample query images to retrieve similar ones. Second, it is difficult to design a retrieval system, which models the visual content of the images and similarity metrics, especially when the background of the image contains objects that are not of interest to the user.

Annotating images with textual keywords and performing queries through these keywords has recently emerged as a better alternative.

Image annotation can be defined as the process of assigning keywords to a given image. Since manual annotation of images is expensive, automatically performing this process by a computer system is of significant importance. Not only, using textual keywords instead of providing similar images is more convenient, but also querying an image in a database with textual keywords gives more satisfactory results compared to low-level visual features,

(a)
Cape Town, South Africa, Londolozi

(b)
Bee, omaraenero, purple purple flowers, flowers, flower, purples, photoshop, border, yellow and black bee, black bee, yellow bee, green, green stem, adobe, adobe photoshop

Figure 1.1: Sample images and their annotations from the Flickr web site.

such as, color and texture, used in CBIR systems. This fact is, mostly, attributed to the large semantic gap between the low-level features and semantic content of images.

There is a variety of image collections that could benefit from automatic image annotation. Some of them include museum collections, satellite imagery, medical image databases, astrophysics images and general-purpose collections on the World Wide Web such as Flickr [11] and Video Google [12]. Considering the well-known Flickr web site [11], which contains several billion photos, searching through these images is a daunting task. Unfortunately, some of the images have no annotation labels; some images are annotated subjectively without reflecting the content of the image as shown in Figure 1 (a); and some of the images are annotated in detail as in Figure 1(b) but requiring substantial manual effort.

The available image annotation approaches can be categorized in two groups. First approach is to construct a statistical model that correlates image features with textual keyword annotations [13], [14], [15], [16], [17], [18], [19], [20], [21], [22]. Second approach is based on finding the most similar images to a given query image extracting the visual features and using the annotation of those images as the query result [23], [24]. Both of these approaches require extraction of low level visual features from images, either to be used in the construction of a statistical model or in direct comparison of images with each other.

There are many ways of extracting the low level visual features from images [25]. The available methods can be categorized in three different groups. First group of methods involve dividing the image into a grid of rectangles with a predefined size and extracting features from each of the rectangles. Second group of methods employ image segmentation algorithms to find regions and extract features in these locations. Third group of methods extract a set of interest points to find critical locations in the image and extracting features around these points. Low level features, extracted from images are mainly based on either color or texture information. Some methods cluster low level visual features into so called ”visterms” to obtain a discrete representation of visual properties of images to enable the match between visual and textual information. This approach also simplifies the computation and reduces complexity.

Most of the studies that use statistical models in automatic image annotation have been inspired from the research, related to text information retrieval. One of the pioneer works proposed by Mori et al. [13] use a co-occurrence model between the words and visterms obtained from low-level features of grid rectangles. Another work proposed by Duygulu et. al. [14] describes images using a vocabulary of visterms. First, a segmentation algorithm is employed to create regions for each image. Then, the visual features are extracted from the regions. The crucial point of this approach is to represent the visual information by a set of visterms, which are created using a clustering algorithm. Then, a statistical machine translation model is used to translate the visterms to keywords. In [15] , Blei and Jordan develop a model, called Latent Dirichlet Allocation (LDA) to model joint distribution of image and text data. This model finds conditional relationships between latent variable representations of image region and word sets. In [16], Jeon et. al. describe image regions using a vocabulary of visterms, as in [14]. However, they use cross-media relevance models (CMRM) to annotate images. Continuous-space relevance model of Lavrenko [17] is quite different than CMRM model where low level visual features are not clustered into visterms, but using continuous features results in better annotation at the expense of increased computational complexity. In [18], Li et. al. use two dimensional Hidden Markov Models obtained from rectangular grid of images to correlate with concepts. This model has been improved to come up with a real time image annotation system in a recent study [19]. Monay et. al. [20] model an image and it’s associated text captions as a mixture of latent aspects. They use latent aspects learned from text captions to train visual feature probabilities, so that latent aspects can be assigned

to a new image. Based on these latent aspects most likely text captions are output as annotation result. Carneiro et. al. [21] learn Gaussian mixture models from images to compute a density estimate for each word that are used to annotate images using minimum probability of error rule. Liu et. al. [22] propose graph learning to learn image-to-image, image-to-word and word-to-word relationships. Word-to-word relationships are used to refine annotations obtained from the other two types of relations.

Recently, there have been attempts to attack image annotation problem by directly finding the visually nearest neighbors of an image in an annotated set of images and using the annotation results of the corresponding similar images. In [23], Oztimur and Yarman Vural propose a two layered architecture to compute keyword probabilities. In the first layer, keyword probabilities are computed based on the distance of the specific low level visual features of the query image to those of the training set individually. In the second layer, these probabilities are combined, for obtaining the majority voting decision. As opposed to previous models, this method extracts low level features from the whole image instead of a specific region. In [24], Wang propose a similar approach where image annotation is based on finding the visually similar images to a query image. In this model, for partially annotated query images, existing annotation keywords is used to limit the search space, and to cope with the increased computational complexity hash codes are used in comparison of visual features.

All of the approaches to image annotation, mentioned above are quite far from the requirements of the end user. Therefore, one can claim that the methods developed for automatic image annotation are still in their infancy and there is a long way to reach the ultimate goal to automatically annotate large image databases for a specific application domain.

There are two major contributions in this thesis:

First, we propose a new method to partially close the semantic gap, which can be explained as the huge difference between complex high level semantic knowledge and low level visual features such as color, texture and shape. For this purpose, our goal is to improve the information content of the system. This task is achieved by introducing ”side information” to the system. The side information is simply defined as the already available, but unused information in the annotation system. One may use the side information to improve the visual features extracted from the image regions. This improvement comes from guiding the clustering process with side information that co-exists with the visual features. Clustering of low level visual features

is performed in such a way that features with the same side information are constrained to fall in the same clusters. By elevating the information content of visual features by the complementary side information, we expect to close the semantic gap between low level visual features and the high level annotation text.

There are many ways of defining and using side information. We use three different types of side information in this thesis. The first one is based on hidden topic probabilities obtained from annotation keywords associated with images. The hidden topic probabilities are computed by the PLSA algorithm [26]. This side information is associated with visual features extracted from image regions obtained from N-Cut region segmentation algorithm [27]. The other two side information are visual, namely orientation and color information both of which are extracted from interest points that correspond to critical locations in images. The orientation information is the dominant direction obtained from the peaks in a histogram of gradient orientations of points around interest points. The color information is based on LUV color features [28] around interest points. Both of these side information are used in clustering of SIFT features [29] extracted from images.

The definition of side information is not unique and depends on the visual and textual content of the image. For example, if one needs to train the data set for the word ”zebra”, the side information should somehow support the detection of stripes, whereas if the word is ”bus”, one needs to support the low level shape features. From the above argument, one can see that the definition of side information is critical. If supportive side information is not available, then the use of other inappropriate side information may spoil the training stage, resulting in even a poorer performance.

The benefits we obtain by using ”side information” available in the annotated images besides the visual features that are clustered, are two-folded. First, clusters become more homogeneous with respect to the provided side information. Hence, they have sharper probability density functions, which reduce the overall entropy of the system. Since visual features become less random, we improve the annotation performance. Second, we can complete clustering in less time, since we compare a visual feature with not all of the cluster centers but with only a subset of it, depending on the constraints provided by the side information. We further reduce the time to cluster visual features by using a parallel version of both the standard K-means and the proposed algorithm.

The second major contribution in this thesis deals with the lack of a detailed comparison measure that compares two image annotation algorithms based on their per-word performances. Two annotation algorithms may differ in such a way that, some words are estimated better by one of the algorithms, while some words are poorly estimated. To be able to compare two image annotation algorithms based on their per-word performances, we introduce a new curve that enables one to see the distribution of relative per-word performances of two different annotation algorithms, by plotting per-word average precision difference values, sorted from highest to lowest. Moreover, we introduce three new metrics based on this curve that show the percentage of words that are better estimated by any of the two algorithms and the total average performances of words that are estimated better/worse than the other algorithm.

1.1 Outline of the Thesis

The rest of this thesis is organized as follows. Chapter 2 provides background knowledge for state of the art techniques in image annotation and quantization of visual features related to this thesis. First, image representations for low level visual features including color and texture are discussed. Next, state of the art image annotation algorithms are discussed under two headings: image annotation algorithms using quantized image regions and image annotation algorithms using continuous features. For visual feature quantization, semi-supervised clustering algorithms are explained under search based and distance metric based categories.

In Chapter 3, the proposed system, Semi Supervised Annotation (SSA) is discussed in detail. First, image annotation problem is introduced formally. Next, detection and extraction of low level visual features are given. Then, Side Information concept is introduced and defined. Next, the rationale behind the semi-supervised clustering of visual features, instead of plain clustering is explained and the algorithm that employs the side information in semi-supervised clustering of low level visual features is described. Finally, we discuss parallel version of the algorithm and computational complexity for both serial and parallel versions of the algorithm.

Chapter 4, presents thorough experiments, to test the performance of SSA and compare it to the state of the art annotation algorithm PLSA-Words [20]. First, data set used in the experiments is explained. Next, the performance metrics are discussed and a new per-word performance comparison curve and three metrics based on this curve are introduced. Then,

estimation of system parameters by cross validation is discussed. Next, overall and per-word performance of the system are given. Finally, we show that overall entropy of the system is reduced by making use of side information. The conclusion and future directions are presented in Chapter 5.

CHAPTER 2

STATE OF THE ART TECHNIQUES FOR IMAGE ANNOTATION AND SEMI-SUPERVISED CLUSTERING

This chapter aims to discuss the state of the art image annotation techniques, their superiorities and weaknesses. The major approaches used in image annotation are overviewed and compared. For this purpose; first, the techniques for image representations used in annotation algorithms are presented.

It is well known that one of the major steps of image annotation is to represent the visual information of the image content. Therefore, we start by explaining the major visual features used in image annotation problem, together with their representation. Considering the large variety of the features and their large variances, one needs to quantize the feature space in order to make the annotation of visual information by finite number of words.

As it will be seen in the subsequent chapter, the major contribution of this thesis is to close the semantic gap between the visual and textual representation of images. We propose to semi-supervise the quantization of the visual features. In order to support our approach, we review the major semi-supervised clustering algorithms in this chapter.

2.1 Representation of Visual Information for Annotation

State of the art image annotation algorithms extract usually visual features either from the whole image [23], [24], or select regions of interest from the image first, and then extract low level features from these regions separately. There are three major approaches for region selection. The first approach is to divide the image into regions by a region segmentation

algorithm [14] such as Normalized Cut [27]. The second approach is to divide the image into regions by using a grid of rectangles of fixed size [13], [18], [20], [21]. The third way is to automatically find out regions of interest by an algorithm such as difference of Gaussians (DoG) point detector [29] and extracting features from these regions where interest points are taken as maxima/minima of the Difference of Gaussians (DoG) that occur at multiple scales.

Extracting features from the whole image is a global approach that is a simple method and it may work well for data sets where very similar images are present. If there is an annotated image in the training set, which is very similar to the query image, the same annotation is attributed to the unlabeled images. Despite of its simplicity the major disadvantage of this method is its inability to generalize. Since images are defined by the overall content, the method is unable to learn the objects that can be present in an image individually. Another problem is its inability to recognize an image if some of the objects are occluded or displayed in a different way.

Finding out regions of interest is a local approach. This method has the advantage of being able to better generalize. As the number of image labels increase, local approaches are more advantageous because of their ability to recognize at the object level. Another advantage of local approaches is its robustness to occlusion. Dividing the image into regions or performing segmentation lies somewhere between global and local approaches. Since segmentation is an ill-posed problem, one may prefer using grids instead of using a segmentation method.

Visual information is represented as a set of low level visual features extracted from the whole image, or regions selected as explained above. In the following section, we discuss common low level visual features that are used in the state of the art image annotation algorithms.

2.1.1 Feature Spaces for Image Annotation

In most common features for image annotation, color and texture information are utilized. The other common features are some shape features, such as ratio of the region area to the perimeter squared, the moment of inertia, the ratio of the region area to that of its convex hull, region size and region position [14]. Blob feature consisting of a mixture of color, texture, and aforementioned other information employed by Duygulu is used in [16], [30], [15] [17], [20] as well as others for comparison purposes. It is an open problem to close the semantic

gap that is indicated by the difficulty of reaching high level semantic knowledge represented by annotation keywords through low level features such as color, texture and shape.

Low level features used in the state of the art image annotation algorithms can be classified into two groups: color features and texture features. In the following sections, we discuss common visual features based on color and texture.

2.1.1.1 Color Features

Colors are represented in a variety of color spaces. Common ones are RGB [13], [14], LAB [14], HSV [20], YBR [21] and LUV [18]. RGB Color Space is most commonly employed color space for digital images and general storage format for cameras. Unlike RGB, LAB is designed to approximate human vision. HSV is good for high intensity white lights, and different surface orientations relative to the light source. YBR has the ability to reduce the redundancy present in RGB color channels and can separate luminance and chrominance components. LUV provides perceptual uniformity, approximates human vision, but has the disadvantage of being computationally expensive.

Frequently used color features are color histogram [13], [20], color average and standard deviation [14], [18], pixel color concatenation [21], Color Layout, Scalable Color, Color Structure MPEG-7 features [23]. Color histograms are computed in two or three dimensional formats depending on whether all three (RGB in [13]) components of the color space are used or just only two (Hue-Saturation (HS) in [20]). Each color component corresponding to the pixels of an image or a region is quantized into some fixed number of values and accumulated in the corresponding bins. Since images with the same color content distribution, but with a different physical layout end up with the same histogram, this feature has difficulty to discriminate especially in large datasets. Color average and standard deviation features are calculated by averaging and finding out the standard deviation of all the pixels for each color component. Since this feature is a summary of image content, it can be used for small image patches and is not suitable for a global image representation. Pixel color concatenation corresponds to simple concatenation of color component values of all the pixels. This feature requires extensive storage and processing power; because of the space requirements and the incurred high dimensionality of the feature space. Color Layout represents the spatial layout of color images. Scalable Color is basically a color histogram in the HSV Color Space that is encoded

by a Haar transform. Color Structure is a feature capturing both color content and information about the spatial arrangement of the colors. Color Layout, Scalable Color, and Color Structure features use spatial information with the aim of more discriminative power.

2.1.1.2 Texture Features

Texture feature refers to repeating pattern of spatial variations in image intensity that can be identified with descriptions such as fine, coarse, grained and smooth. Various texture features used for annotation are edge histogram [13], [23], mean oriented energy [14], SIFT [20], wavelet transforms [18] and Homogeneous Texture [23].

In edge histogram feature, edge orientation value of each pixel of an image is quantized into some fixed number of values and accumulated in the corresponding bins. Edge histogram feature captures spatial distribution of edges. It is mainly used to identify non-homogeneous texture regions. Mean orientation energy, Gabor filters and Homogeneous texture are all based on a series of multi-scale and multi-orientation cosine modulated Gaussian kernels.

Since we compare our method with the state of the art image annotation algorithm of [20], we employ Scale Invariant Feature Transform (SIFT) features as in [20], which will be explained in the following subsection.

Scale Invariant Feature Transform (SIFT) SIFT features are extracted using the local histograms of edge orientation from a local interest area [29]. The most widely used local interest area selection method is Difference of Gaussians (DOG) [29]. Some other mostly used interest point detectors are Harris Corner Detector [31], Fast Hessian [32] and Features from Accelerated Segment Test (FAST) [33], Saliency Detector [34] and Maximally Stable Extremum Regions [35].

Difference of Gaussians (DOG) In this method, area of interest is selected based on the maxima and minima of the difference of Gaussian (DOG) operator. It is scale, orientation and illumination invariant. Different scales can be represented by scale-space function defined as:

$$L(x, y, \sigma) = G(x, y, \sigma) * I(x, y) \, , \tag{2.1}$$

where $*$ is the convolution operator, $G(x, y, \sigma)$ is a variable-scale Gaussian function, σ is the Gaussian parameter and $I(x, y)$ is the input image. Stable interest points are identified using the Difference of Gaussians operator which is defined as:

$$D(x, y, \sigma) = L(x, y, k\sigma) - L(x, y, \sigma) \ , \tag{2.2}$$

where k corresponds to the smoothing factor. A pyramid of Difference of Gaussians is generated from the input image. Each layer of the pyramid consists of difference of Gaussians obtained by taking the difference of successively blurred images for a given scale. Successive layers of the pyramid are obtained by downsampling the input image by a factor of two and further obtaining the difference of Gaussians for the corresponding scale. If the number of scale space levels is given as s, the smoothing factor k can be computed as $k = 2^{1/s}$.

The interest points are found by comparing each pixel with its immediate 8 neighbors, 9 neighbors in the preceding scale space level and 9 neighbors in the following scale space level for a total of 26 neighbors. All pixels corresponding to maxima or minima among all its neighbors are considered as interest points.

The detection process is scale, illumination and orientation invariant.

SIFT Feature Extraction Before computing the interest point descriptor, an orientation is assigned to each interest point. The interest point descriptor is then represented relative to this orientation, resulting in invariance to rotation. Orientation assignment is performed as follows:

First, the scale of the interest point is used to select the Gaussian smoothed image L. Next gradient magnitude is computed as follows:

$$m(x, y) = \sqrt{(L(x+1, y) - L(x-1, y))^2 + (L(x, y+1) - L(x, y-1))^2} \ . \tag{2.3}$$

The orientation is computed using:

$$\theta(x, y) = \arctan\left(L(x, y+1) - L(x, y-1)\right)/\left(L(x+1, y) - L(x-1, y)\right) \ . \tag{2.4}$$

An orientation histogram with 36 bins is constructed each bin spanning 10 degrees. A neighboring window is formed for each interest point using a Gaussian-weighted circular window

with a σ which is 1.5 times that of the scale of the interest point Each pixel in the window is then added to the histogram bin weighted by its gradient magnitude and by its Gaussian weight within the window . The peaks in the histogram correspond to dominant orientations. The orientations corresponding to the highest peak and peaks that are within 80% of the highest peaks are assigned to the interest point. In the case of multiple orientations , an additional interest point is created having the same location and scale as the original interest point for each additional orientation.

To compute SIFT descriptor, the neighborhood of 16x16 pixels around the found interest point is divided into a grid of 4x4 blocks and a gradient orientation histogram of each 4x4 block is computed. Since there are 16 histograms each having 8 bins corresponding to each orientation, the final SIFT feature ends up in a 128 element vector.

Since SIFT features are local, they are robust to occlusion and clutter and have the ability to generalize to a large number of objects. One shortcoming of SIFT is the added complexity compared to global features.

The above mentioned visual features are only a few of tremendous amount of other features. The reason that we focus on these features is two folded: Firstly, in our experiments we employ the defacto standard data base Corel, which is used in demonstrations of most of the image annotation systems. This database is heavily characterized by color and texture. Secondly, the selected visual features are also employed in state of the art image annotation systems. Therefore, the above features enable us to make fair performance comparisons between the proposed work of this thesis and the other available algorithms.

2.2 Automatic Image Annotation Techniques

In this section, techniques for automatic image annotation are discussed. Let us start by formally defining image annotation problem. Suppose that the training set S , consists of n images in set $I = \{I_j\}_{j=1}^{n}$ and associated text documents in set $D = \{D_i\}_{i=1}^{n}$ pairs, where $S = \{(I_1, D_1), (I_2, D_2), ..., (I_n, D_n)\}$. Suppose also that, each image I_j consists of regions and represented by $I_j = \{r_{j1}, r_{j2}, ..., r_{jN(j)}\}$, where r_{jm} is the feature vector associated with region m of image I_j and $N(j)$ is the number of regions for image I_j. Let,

$$R = \{r_{11}, ..., r_{1N(1)}, r_{21}, ..., r_{2N(2)}, r_{n1}, ..., r_{nN(n)}\} .$$

Each text document D_i consists of words obtained from a dictionary, W, where

$$D_i = \{w_{i1}, w_{i2}, ..., w_{iK(i)}\} ,$$

w_{ij} is j-th word of text document D_i, $w_{ij} \in W$,

$$W = \{word_1, word_2, ..., word_L\} ,$$

L is the size of the dictionary W, and $K(i)$ is the number of words for text document D_i.

Given a query image Q where $Q = \{r_{Q,1}, r_{Q,2}, ..., r_{Q,N(Q)}\}$, $N(Q)$ is the number of regions in image Q, image annotation can defined as finding a function $F(Q) = A$ where $A = \{w_{A,1}, w_{A,2}, ..., w_{A,K(A)}\}$, $K(A)$ is the number of words in annotation A and $w_{A,i}$ is obtained from dictionary W.

Over the past decades, there is a vast amount of work on image annotation problem. A good source of references can be found in [2] and [36].

There are many problems with the currently available image annotation techniques. In order to develop a working, real life image annotation system, the researcher on this field should attack three major obstacles:

First of all, as in all of the computer vision applications, semantic gap problem still remains as an unsolved issue. Although the low level feature extraction techniques are well studied, it is still very difficult today for automated high level semantic concept understanding based on these low level features. This is due to the so called ”semantic gap” problem, which can be explained as the difficulty of representing complex high level semantic knowledge through low level visual features such as color, texture and shape. This is still an open problem and under research from a variety of disciplines involving pattern recognition, image processing, cognitive science and computer vision.

Second problem of the image annotation literature is that there is not a consistent way of measuring the performance to evaluate the image annotation techniques. Currently, the performance of the image annotation algorithms is measured by a variety of metrics used in CBIR systems. In most of the systems, [13], [14], [16], [17], [21] precision and recall have been adopted. Liu et. al. [22] use precision and recall as well as number of words with non-zero recall. Monay et. al. [20] uses mean average precision claiming that it is more important to use such a metric since main purpose of annotation is retrieval. Blei and Jordan, [15] used

annotation perplexity. Barnard et. al. [30] defined three different scores. First measure is Kullback-Leibler divergence between predictive and target word distributions. Second measure is so called normalized score that penalizes incorrect keyword predictions. Third measure is the coverage, which is based on the number of correct annotations divided by the number of annotation words for each image in the ground truth. In the annotations, some words might be more important than others and some words could be accepted as correct even though they are not in the ground truth, but if they are semantically similar. These considerations should be taken into account for measuring the performance of image annotation algorithms. Moreover, the available metrics do not compare the image annotation algorithms based on their per-word performances.

Third problem is the lack of a standard image data set, which covers statistically meaningful and stable images with reasonably many text annotations. Mori et. al., [13], used a multimedia encyclopedia, in [14], [16], [17], [21], [22], [23] part of the Corel data set have been used. They have used 4500 images for training, 500 images for testing purposes. In [30] and [20] a bigger part Of Corel photos have been used. The dataset they use consist of 10 subsets collected from 16000 photos, each set on the average having 5244 images for training and 1750 images for testing. Recently there have been attempts to use images from world wide web [19], [24], [37], [22] but the number of images used is very low compared to what is needed in a real practical image annotation system. In [19], 54,700 images collected from Flickr web site have been used. In [24], there are 450 images collected from Google image search engine and in [37], there are 5260 images collected from Yahoo image search engine. In [22] 9046 web images on 48 topics have been used. Unfortunately, none of the above mentioned data sets contain sufficient number of samples, which are consistently annotated to yield an appropriate training set.

Corel data set is criticized of having visually very similar images in the set [37]. This property of Corel makes it easy to find a similar image to a given query image and use its annotation. Although, it is not unrealistic to find images with very similar content in real, large data bases, such as world wide web consisting of billions of pages, the same technique cannot be used since there would be many images matching the same global features but with possibly quite different content. Other data set collections obtained from the web have the problem of possible noise, since annotations might not be correct and can be done differently from person to person. Some annotated sample images from Flickr web site are shown in Figure

Blue, Green, Cloud, Drive-by, Forest, Meadow, Motion, Tree, Yellow, out of the car, Blue Green, Horse

brunswick, maine, snow, sun, light, trees, scenic, landscape, Seen in Explore

purple, flower, grass

Figure 2.1: Sample images and their annotations from the Flickr web site.

2.2. Annotations ”Drive-by, Motion, out of the car, Seen in Explore, brunswick, maine” are very subjective and quite difficult to learn from the attached images.

In the following sections, we give an overview for the major annotation algorithms discussing the pros and cons. First; algorithms, where low level image features are quantized using a clustering algorithm are explained. Then, we present algorithms, where low level visual features are used as they are, without any quantization. Reported performance results are based on two different datasets obtained from Corel data set. The first one uses 5000 images [14] and the second dataset consists of 10 subsets collected from 16000 images [30], that we refer to as Corel2002 and Corel2003 data sets, respectively.

2.2.1 Image Annotation Using Quantized Image Regions

Automatic annotation of images, depending on their context requires learning of image content. In this sense, annotation problem can be considered as a mixture of classification and CBIR problem. Therefore, the techniques are similar to that of learning the visual content of images and associating the visual content to a set of words that can be considered as classes.

One of the approaches to automatic image annotation involves segmenting the image into regions and representing these regions by low level features. The low level features are then quantized by clustering to obtain visterms. Therefore, annotation problem is reduced to finding the correlation between annotated words and visterms. First, low level visual features such as color and texture are computed for each region. Next, usually a standard clustering

method, such as the K-means algorithm [38] is used to cluster visual features obtained from image regions. By assigning the cluster label to each region, a discrete representation for image is obtained. The clustering process reduces the computational cost of automatic image annotation, since we use just a cluster label called visterm to represent a region instead of a multidimensional low level visual features vector. This approach opens the door to the annotation problem using text based methods [14], [16], [20].

2.2.1.1 Co-occurrence Model

Work by Mori et al. [13] is one of the first attempts at image annotation, where the images are first tiled into grids of rectangular regions. Next, a co-occurrence model is applied to words and low-level features of grid rectangles. Visual features extracted from each grid rectangle are clustered to obtain visual terms by using the cluster labels that are briefly called visterms. Using Bayes rule, the conditional multinomial probability $P(word_i|visterm_j)$ of keyword $word_i$ for a given cluster $visterm_j$ is estimated by:

$$P(word_i|visterm_j) = \frac{P(visterm_j|word_i)P(word_i)}{\sum_{k=1}^{L} P(visterm_j|w_k)P(w_k)} . \tag{2.5}$$

The conditional multinomial probability $P(visterm_j|word_i)$ of cluster $visterm_j$ for a given keyword $word_i$ is approximated by dividing the total number of words m_{ji} in cluster $visterm_j$ for word $word_i$ by the total number of instances of $word_i$ in the data set, n_i; and approximating the multinomial probability $P(word_i)$ of word $word_i$ by dividing the total number of instances of $word_i$ in the data set, n_i; by the total number of words in data, N; Note that, although a word can be related to only one cluster (visterm), all the conditional visterm probabilities are updated given a word. Hence, the approximation of conditional multinomial probability of a cluster, by dividing the total number of words in that cluster to the total number of instances of that word may not be accurate. So, the conditional probability becomes

$$\approx \frac{(m_{ji}|n_i)(n_i|N)}{\sum_{k=1}^{L} P(m_{jk}|n_k)(n_k|N)} = \frac{m_{ji}}{M_j} , \tag{2.6}$$

where, m_{ji} is the total number of words $word_i$ in cluster $visterm_j$,

$$M_j = \sum_{k=1}^{L} m_{jk}$$

the total of all words in cluster $visterm_j$, n_i the total number of instances of $word_i$ in the data set, and

$$N = \sum_{k=1}^{L} n_k$$

and L is the size of the dictionary. Next, an image can be annotated in the testing stage as follows: First, the test image is tiled into grid of rectangles as in training images. Next, the corresponding cluster is computed for each such rectangle. Third, an average of the likelihoods of the nearest cluster is computed. Finally, keywords that have largest average of the likelihoods are output as the annotation result.

This model has a reported precision of 0.03 and recall of 0.02 on Corel2002 data set reported by [17]. The main reason for this low performance is the assumption that each keyword is associated with a cluster, although it is likely that more than one cluster determines one of the keywords associated with an image. Another drawback is that frequent words are mapped to almost every visterm. In addition, many training examples are needed to correctly approximate the conditional visterm probabilities.

2.2.1.2 Translation Model

In this model [14], the annotation of images is considered as a translation of visual information to text words similar to translating an English text to French. The lexicon of the visual language is the visual terms obtained by clustering image regions. Although in the original paper, these visual terms are called blobs, we call them visterms to maintain the consistency among all models. Let us assume $visterm_{im}$ is the visterm associated with region m of image I_i. In this model, it is assumed that each visterm is assigned to a word. Assignment probability of region r_{ik} to word w_{ij} is shown by $P(a_{ij} = k)$. Translation probability of $visterm_{ik}$ into w_{ij} is shown by $P(t_{ij} = k)$. Given an image I_i and an annotation D_i, the probability of annotating I_i with D_i is computed as follows:

$$P(D_i|I_i) = \prod_{j=1}^{K(i)} P(w_{ij}|I_i) = \prod_{j=1}^{K(i)} \sum_{k=1}^{N(i)} P(t_{ij} = k)P(a_{ij} = k) \ , \tag{2.7}$$

where $P(t_{ij} = k)$ is the probability of translating $visterm_{ik}$ into w_{ij} and $P(a_{ij} = k)$ is the probability of assigning r_{ik} region to w_{ij} By maximizing the likelihood of the training images, these translation probabilities can be computed:

$$l(S) = \prod_{i=1}^{n} P(D_i|I_i) = \prod_{i=1}^{n} \prod_{j=1}^{K(i)} \sum_{k=1}^{N(i)} P(t_{ij} = k)P(a_{ij} = k) \ . \tag{2.8}$$

The Expectation-Maximization algorithm is applied to find the optimal solution that correspond to translation probabilities $P(t_{ij} = k)$ and assignment probabilities $P(a_{ij} = k)$.

This method performs better than Co-occurrence Model [13] with a precision of 0.06 and recall of 0.04 on Corel2002 data set. However, the method also suffers from the same major assumption that each keyword is associated with a visterm, although a keyword represents potentially more than one region.

2.2.1.3 Cross Media Relevance Model (CMRM)

In this model [16], it is assumed that for a pair $J = \{Q, A\}$ of an image Q and its annotation A, there exists some underlying probability distribution $P(.|J)$ which is called relevance model of J. Similar to previous models, low level visual features from image regions are clustered to obtain visterms. Since we do not have any way of observing A for a query image Q, the probability of observing a word w is approximated by the conditional probability of w given that we observe Q. Assume, $Q = \{visterm_{Q,1}, visterm_{Q,2}, ..., visterm_{Q,N(Q)}\}$, $visterm_{Q,k}$ corresponds to the visual term obtain from clustering the image region $r_{Q,k}$ and $N(Q)$ is the number of regions in image Q. Hence, conditional word probability can be written as

$$P(w|J) \approx P(w|Q) \ . \tag{2.9}$$

On the other hand, the joint probability of w and Q can be estimated as follows:

$$P(w, Q) = \sum_{i=1}^{n} P(S_i)P(w, Q|S_i) \ , \tag{2.10}$$

where $S_i = (I_i, D_i)$.

Assuming observation of words and visterms are mutually independent, we can rewrite the above equation as:

$$P(w, Q) = \sum_{i=1}^{n} P(S_i) P(w|S_i) \prod_{k=1}^{K(i)} P(visterm_{ik}|S_i) \ , \tag{2.11}$$

where $P(S_i)$ is assumed to be a uniform distribution. $P(w|S_i)$ and $P(visterm_{ik}|S_i)$ are assumed to be multinomial distributions that are computed using the smoothed maximum likelihood as follows:

$$P(w|S_i) = (1 - \alpha_{S_i}) \frac{\#(A, S_i)}{N(i)} + \alpha_{S_i} \frac{\#(A, S)}{n} \ , \tag{2.12}$$

$$P(visterm_{ik}|S_i) = (1 - \beta_{S_i}) \frac{\#(visterm_{ik}, S_i)}{K(i)} + \beta_{S_i} \frac{\#(visterm_{ik}, S)}{n} \ , \tag{2.13}$$

where $\#(w, S_i)$ is the frequency of $word_j$ in image annotation, S_i and $\#(word_j, S)$ is the number of words in the training set, $\#(visterm_k, S_i)$ is the frequency of $visterm_k$ in image I_i and $\#(visterm_k, S)$ is the number of $visterm_k$ in the training set, α_{I_i} and β_{I_i} are smoothing parameters. In this model, words in the training set are propagated to a test image based on their similarity to the training images.

The precision and recall performance of this method is reported as 0.10 and 0.09, respectively for the Corel2002 data set. Although it performs better than Translation Model, because of the joint probability estimation, which assumes mutual independence of annotation words and low level visual features, this method can not reach the performance level of the methods that estimate conditional probabilities directly.

2.2.1.4 PLSA-Words

PLSA-Words algorithm is based on Probabilistic Latent Semantic Indexing method given in [20]. The algorithm links text words with image regions. The flowchart of the PLSA-Words feature extraction process is given in Figure 2.2. For each training image, two types of features are extracted. SIFT features are extracted from interest points detected by Difference of Gaussians feature detector. Hue-Saturation (HS) features are extracted from a grid of rectangles.

Both SIFT and HS features are clustered with K-means to obtain separate visual codebooks. Visual Codebook-1 and Visual Codebook-2 are obtained from HS and SIFT features, respectively. Both of these codebooks are used in the PLSA-Words algorithm. For a query image, visual features are extracted as explained above to find the corresponding visterms.

For each document D_i in the training set, a topic z is chosen according to a multinomial conditioned on the index i. The words are generated by drawing from a multinomial density conditioned on z. In PLSA, the observed variable i is an index into some training set. In PLSA, assuming T topics, D_i corresponding to ith document and $word_j$ corresponding to jth word, word document joint probability $P(word_j, D_i)$ is given by:

$$P(word_j, D_i) = P(D_i) \sum_{t=1}^{T} P_{wt}(word_j|z_t) P_{td}(z_t|D_i) \ . \tag{2.14}$$

Maximum likelihood parameter estimation is performed with the expectation maximization algorithm. The number of parameters for PLSA grows linearly with the number of documents in the training set.

Details of the PLSA-Words algorithm are given in Algorithm 1.

Algorithm 1 PLSA-Words algorithm.

1: Using PLSA algorithm compute $P_{wt}(word_j|z_t)$ and $P_{td}(z_t|D_i)$ probabilities.

2: Keeping $P_{td}(z_t|D_i)$ probabilities computed in the previous step fixed, compute$P_{wt}(visterm_j|z_t)$ probabilities using PLSA algorithm.

3: Using query image visual words and $P_{wt}(visterm_j|z_t)$ probabilities computed in the previous step, compute $P_{td}(z_t|query)$ probabilities using PLSA algorithm.

4: Compute conditional distribution of text words using the following: $P(word_j|query) = \sum_{t=1}^{T} P_{wt}(word_j|z_t) P_{td}(z_t|query)$

5: Output the most probable words for the given query image.

PLSA-Words performs better than CMRM with respect to mean average precision measure when SIFT and HS are used as low level features. PLSA-Words and CMRM mean average precision performances are 0.19 and 0.13, respectively on Corel2003 data set. This performance increase is due to the fact that instead of using the apparently strong mutual independence assumption for text words and visterms as is the case in CMRM method, PLSA-Words computes the conditional probabilities for a text words given visterms by using the product

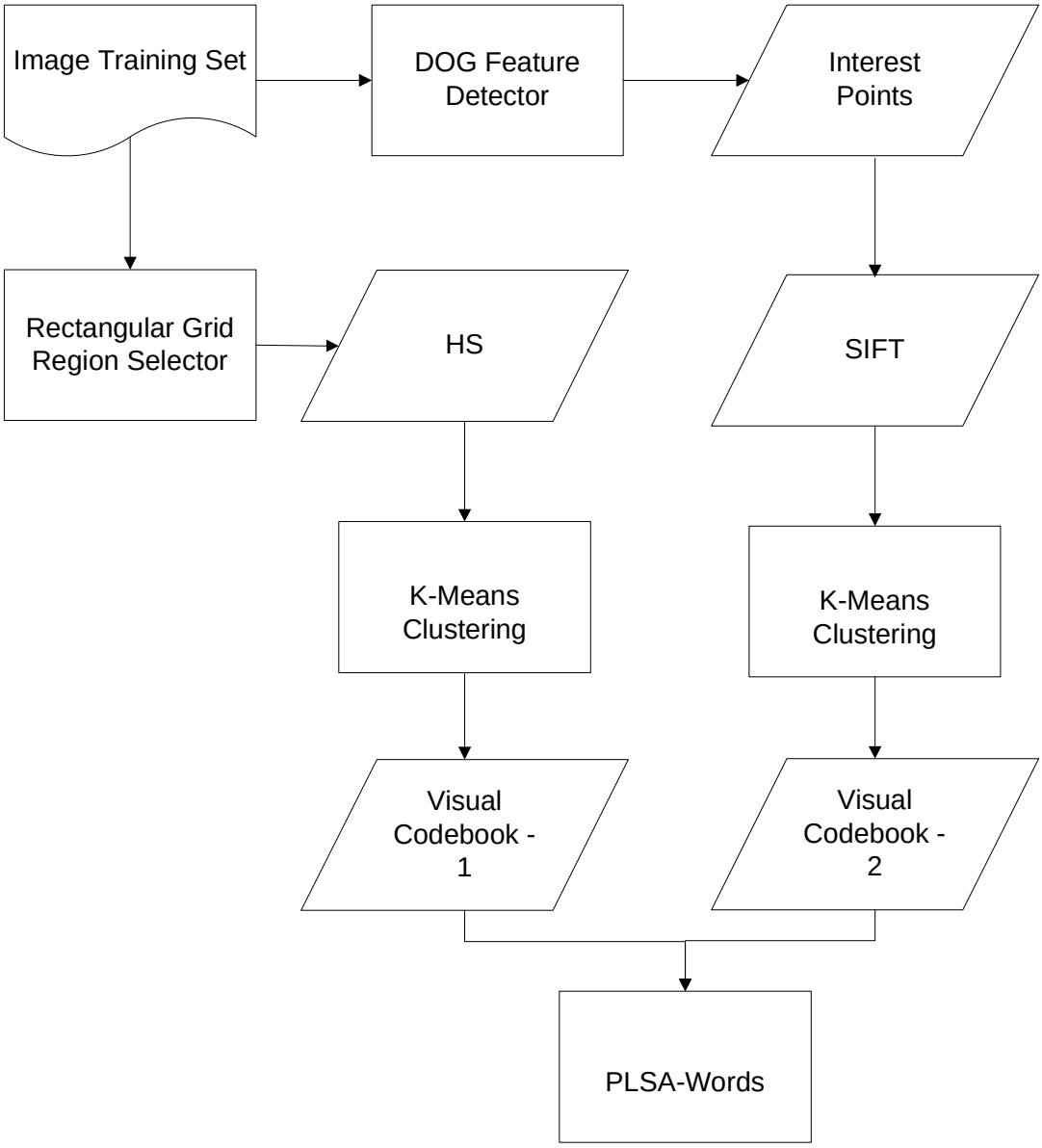

Figure 2.2: The Block Diagram of PLSA-Words Feature Extraction.

of estimated probabilities for text words given a hidden topic, and estimated hidden topic probabilities for a query image based on its visterms and marginalizing over the hidden topics. However, in this algorithm visterms are obtained through a standard K-means clustering algorithm. In this thesis, we improve the clustering process used for obtaining visterms using side information and get better results than PLSA-Words reaching mean average precision of 0.21. This is the best reported result, so far, in the current literature on Corel2003 dataset.

2.2.2 Image Annotation Using Continuous Features

Continuous features correspond to using low level visual features, without any quantization as is the case in the previous sub section. Although discrete representation simplifies the image representation and reduces the annotation complexity, it may lose some important information about the visual content of the image. In this section, some of the major studies for image annotation using continuous features are discussed, where low level visual features are extracted from the images and directly matched to the annotation words.

2.2.2.1 Hierarchical Model

In this model [30], images and corresponding text words are generated by nodes arranged in a tree structure. In this tree representation, the nodes above the leaf nodes correspond to topics and leaf nodes correspond to clusters obtained from the low level visual features and textual words associated with images. The arcs of the tree linking parents to children correspond to the hidden topic hierarchy. Arcs just above the leaf nodes correspond to association of clusters with the most specific topics. Each cluster takes place in one of the leaf nodes and associated with a path from root to the leaf. Hence, the nodes closer to the root are shared by many clusters, and nodes closer to leaves are shared by fewer clusters. This model creates a hierarchical context structure, nodes closer to the root corresponding to more general terms such as *animal* and the ones close to leaves corresponding to more specific items such as *cat*.

Image regions are generated assuming a Gaussian distribution for the feature space. On the other hand, words are generated using a multinomial distribution. Denoting low level image features used for region r_{ij} of image I_i by b_{ij}, and letting b_i denote the set of low level visual features for image I_i, the word and image region observation probabilities are computed as

follows:

$$P(D_i, I_i) = \sum_c p(c) \prod_{w \in D_i} [\sum_l p(w|l,c)p(l|D_i)]^{Z_1} \prod_{b \in I_i} [\sum_l p(b|l,c)p(l|I_i)]^{Z_2} \ , \tag{2.15}$$

where c is cluster index, l is tree level, D_i is sample document, Z_1 and Z_2 are normalization constants differing numbers of words and regions in each image. Z_1 and Z_2 constants are computed as follows:

$$Z_1 = \frac{N_w}{N_{w,D_i}} \ , \tag{2.16}$$

$$Z_2 = \frac{N_b}{N_{b,I_i}} \ , \tag{2.17}$$

where N_{w,D_i} denotes the number of words in document D_i, while N_w denotes the maximum number of words in any document, similarly same denotation applies to N_{b,I_i} and N_b.

To compute the multinomial and Gaussian distribution parameters, the Expectation Maximization algorithm of [39] is used.

There are three major approaches to implement hierarchical models [30], which are explained as follows:

Model I-O In this model, the joint probability of a tree level depends only on the sample document and computed as follows:

$$P(D_i, I_i) = \sum_c p(c) \prod_{w \in D_i} [\sum_l p(w|l,c)p(l|D_i)]^{Z_1} \prod_{b \in I_i} [\sum_l p(b|l,c)p(l|I_i)]^{Z_2} \ . \tag{2.18}$$

Because of the dependency of the tree level to the specific documents in the training set, this model is not a truly generative model.

Model I-1 In this model, the joint probability of a tree level depends on both sample document and the cluster. It is computed as follows:

$$P(D_i, I_i) = \sum_c p(c) \prod_{w \in D_i} [\sum_l p(w|l,c)p(l|c,D_i)]^{Z_1} \prod_{b \in I_i} [\sum_l p(b|l,c)p(l|c,I_i)]^{Z_2} \ . \qquad (2.19)$$

This model suffers from the same problem as the previous model. Both of these models show similar performance.

Model I-2 In this model, the joint probability of a tree level depends only on the cluster and it is computed as follows:

$$P(D_i, I_i) = \sum_c p(c) \prod_{w \in D_i} [\sum_l p(w|l,c)p(l|c)]^{Z_1} \prod_{b \in I_i} [\sum_l p(b|l,c)p(l|c)]^{Z_2} \ . \qquad (2.20)$$

In this model, estimation is performed only at the cluster level, training data is marginalized out. This method gives better performance compared to the previous two models.

In all of the above models, three performance measures are used. First measure is Kullback-Leibler (KL) divergence between predictive and target word distributions. Second measure is normalized score (NS) that penalizes incorrect keyword predictions. Third measure is the coverage (C) that is based on the number of correct annotations divided by the number of annotation words for each image in the ground truth. Experiments are performed on Corel2003 data set. Since Model I-O and Model I-1 performances are similar, results are reported only for I-0. Best results for I-0 are KL=0.099, NS=0.174 and C=0.688, while best reported performances for I-2 are KL=0.104, NS=0.179 and C=0.747 changing by the chosen topology of the tree structure or type of prior probability computations of tree levels. One can conclude that there is not a significant difference among the three models discussed above. However, note that all of them outperform the Translation Model that is reported to be KL=0.073, NS=0.111 and C=0.433 for the three measures mentioned above. These models have the same drawback as the CMRM method discussed in the previous section, since they use the poor assumption of mutual independence between textual words and visual features.

2.2.2.2 Annotation Models of Blei and Jordan

Blei and Jordan [15] propose three different hierarchical probabilistic models for matching the image and keyword data. Both region feature vectors and keywords are assumed to be

conditional on latent variables. The region feature vectors are assumed to have multivariate Gaussian distribution with diagonal covariance and the keywords have multinomial distribution over the vocabulary.

Model l: Gaussian multinomial mixture model (GMM) In this model, a single variable is assumed to be generating both words and image regions. The joint probability for latent class z, annotated words D and image regions can be computed as follows:

$$p(z, I_i, D_i) = p(z|\lambda) \prod_{j=1}^{N(i)} p(r_{ij}|z, \mu, \sigma) \prod_{k=1}^{K(i)} p(w_{ik}|z, \beta) \ , \tag{2.21}$$

where λ is the parameter corresponding to the probability distribution of the hidden variable *z*, which can take simply as uniform distribution. μ and σ are the parameters of the Gaussian distribution and β is the parameter of multinomial distribution that are estimated by the Expectation Maximization [39] algorithm.

Conditional distribution of words given an image can be computed using the Bayes rule and marginalizing out the hidden factor *z*:

$$p(w|Q) = \sum_{z} p(z|Q)p(w|z) \ . \tag{2.22}$$

In this model, it is assumed that textual words and image regions are to be generated by the same hidden factor.

Model 2: Gaussian Multinomial Latent Dirichlet Allocation Although in Gaussian multinomial mixture model, the textual words and images are generated by the same latent variable, in Gaussian Multinomial Latent Dirichlet Allocation (LDA), each document is considered to consist of several topics and word and image observations are generated from these different topics. In this method, the following generative process takes place:

1. A Dirichlet random variable θ, is sampled based on the parameter α [40].
2. Conditional on θ, a multinomial random variable *z* and conditional on *z* a Gaussian random variable *r*, is sampled for each image region.
3. Conditional on θ, a multinomial random variable *v* and conditional on *v*, a multinomial random variable *w* is sampled for each textual word.

Formally, the joint probability for latent class z, annotated words D and image regions can be computed as follows:

$$p(I_i, D_i, \theta, z, v) = p(\theta|\alpha) \prod_{j=1}^{N(i)} p(z_j|\theta) p(r_{ij}|z_j, \mu, \sigma) \prod_{k=1}^{K(i)} p(v_k|\theta) p(w_{ik}|z, \beta) \ . \tag{2.23}$$

Parameters of these conditional distributions are approximated using variational inference methods [15].

Model 3: Correspondence LDA In this model, first image region features are generated and keywords are generated next. Annotation keywords are generated, conditioned on the hidden factor related to the selected region.

The generative process that takes place in this method is as follows:

1. A Dirichlet random variable θ is sampled based on the parameter α [40].
2. Conditional on θ, a multinomial random variable z and conditional on z a Gaussian random variable r, with parameters μ and σ is sampled for each image region.
3. For each textual word, the following steps are performed:
   (a) A uniformly distributed random variable y is sampled based on parameter of the number of textual words in the image.
   (b) Conditional on z and y, a multinomial random variable w with parameter β is sampled.

Formally:

$$p(I_i, D_i, \theta, z, y) = p(\theta|\alpha) \prod_{j=1}^{N(i)} p(z_j|\theta) p(r_{ij}|z_j, \mu, \sigma) \prod_{k=1}^{K(i)} p(y_k|N(i)) p(w_{ik}|y_k, z, \beta) \ , \tag{2.24}$$

where y is assumed to have uniform distribution taking values ranging from 1 to $N(i)$.

The independence assumptions in this model is somewhere between Gaussian multinomial mixture model and Gaussian Multinomial Latent Dirichlet Allocation model. In the former, there is a strong dependence assumption between image regions and annotation keywords;

while in the latter, no correspondence is assumed between image regions and the annotation keywords.

Annotation Models of Blei and Jordan [15] are measured by caption (annotation keyword) perplexity. While the number of hidden factors increase from 1 to 200; caption perplexity for Gaussian multinomial mixture model (GMM) remains around 60 to 63, caption perplexity for Gaussian Multinomial Latent Dirichlet Allocation model steadily increases from 65 to 80 and for Correspondence LDA model steadily decreases from 72 to 50. Note that lower numbers mean better performance in perplexity measure. Among all these models, GMM performs the worst and the Correspondence LDA model performs the best. GMM's major weakness is the assumption that the same hidden topic generates both the image regions and textual words. Gaussian Multinomial Latent Dirichlet Allocation model assumes that textual words and image regions are generated by different hidden topics, hence lacking a direct correspondence. Last model lies somewhere between Model 1 and Model 2, but shows the greatest performance owing to the flexibility that multiple textual words can be generated for the same regions, and the textual words can be generated from a subset of the image regions.

2.2.2.3 Automatic Linguistic Indexing of Pictures by a Statistical Modeling Approach

In this model [18], each image is annotated by using a category which itself is described by a number of keywords. Categories are manually annotated as compared to hidden topics used in PLSA-Words algorithm, where topics are obtained automatically using PLSA algorithm. Categories in this model are used in a similar way to topics in PLSA-Words algorithm in the sense that both algorithms first try to identify the related topics or categories first, then choose annotation words based on statistical properties.

In the model, images are divided into rectangular grids size of which reduced to half, each time in a pyramid fashion and features extracted from these rectangles are modeled as two-dimensional Multi-resolution Hidden Markov Model (2D MHMM). Feature vectors are assumed to be drawn from a Gaussian distribution. The 2-dimensional nature of the Hidden Markov Model captures the relationship between grid rectangles. Given a test image, the similarity of the image to each 2D-MHMM model estimated for each category is computed. Test image is annotated by key words selected from the description of categories yielding highest likelihoods. Words are selected according to their statistical significance, which is based on

the occurrence of the word in the top most predicted categories.

This model assumes that a category is already assigned to each image, and uses a different dataset than the other methods. Therefore, it is not possible to directly compare this method with the other methods discussed in this thesis.

2.2.2.4 Continuous Relevance Model

Continuous-space relevance model [17] is an improvement to CMRM model that is based on the quantized image regions. In generating visual features, continuous probability density functions are used to avoid the abrupt changes related to quantization. In this model, it is assumed that for a pair $J = \{Q, A\}$ of an image Q and its annotation A, where $Q = \{r_{Q,1}, r_{Q,2}, ..., r_{Q,N(Q)}\}$, $N(Q)$ is the number of regions in image Q, low level visual features corresponding to regions are denoted by G where $G = \{g_{Q,1}, g_{Q,2}, ..., g_{Q,N(Q)}\}$, $A = \{w_{A,1}, w_{A,2}, ..., w_{A,K(A)}\}$, $K(A)$ is the number of words in annotation A, the joint probability of observing words and image regions is computed as follows:

$$p(Q, A) = \sum_{i=1}^{n} P_S(S_i) \prod_{j=1}^{K(A)} P_M(w_{A,j}|S_i) \prod_{k=1}^{N(k)} \int_{R^k} P_R(r_{(Q,k)}|g_{(Q,k)}) P_G(r_{(Q,k)}|S_i) dg_{(Q,k)} \ , \quad (2.25)$$

where $S_i = (I_i, D_i)$.

P_S is assumed to have uniform distribution. $P_R(r|g)$ probability distribution is used to map low level visual feature generator vectors g to actual image regions r. For every image region, one corresponding generator is used. The following distribution is assumed for P_R:

$$P_R(r|g) = \left\{ \begin{array}{ll} 1/N_g & if G(r) = g \\ 0 & otherwise \end{array} \right\} , \quad (2.26)$$

where N_g is assumed to be a constant independent of g.

Given a model S_i the following Gaussian distribution is used to generate the image features:

$$P_G(g|S_i) = \frac{1}{n} \sum_{i=1}^{N(i)} \frac{1}{\sqrt{2^k \pi^k |\Sigma|}} \exp\{(g - G(r_i))^T \Sigma^{-1} (g - G(r_i))\} \ , \quad (2.27)$$

where $G(r_i)$ is the feature vector of a region in image I_i and k is the length of the low level visual feature vector.

The word probability estimated based on multinomial distribution with Dirichlet smoothing can be computed as follows:

$$P_M(w|S_i) = \frac{\mu p_w + N_{w,S_i}}{\mu + \sum_{w'} N_{w',S_i}},$$

where μ is an empirically selected constant, p_w is the relative frequency of observing the word in the training set, N_{w,S_i} is the number of times word w occurs in the observation D_i.

As expected, this model performs better than its discrete counterpart, Cross Media Relevance Model with precision and recall values of 0.16 and 0.19 as opposed to 0.10 and 0.09, respectively on Corel2002 data set. Because of the joint probability estimation that assumes mutual independence of annotation words and low level visual features, this method can not reach to the performance level of method that estimates conditional probabilities directly.

2.2.2.5 Supervised Learning of Semantic Classes for Image Annotation and Retrieval Model

In this model [21] , image features are extracted from overlapping regions based on a sliding window over the image. In this model, it is assumed that for an image Q and its annotation A, where $Q = \{r_{Q,1}, r_{Q,2}, ..., r_{Q,N(Q)}\}$, $N(Q)$ is the number of regions in image Q, low level visual features corresponding to regions are denoted by G, where $G = \{g_{Q,1}, g_{Q,2}, ..., g_{Q,N(Q)}\}$.

First, for each image a class conditional density consisting of a mixture of 8 Gaussians is estimated using the following equation:

$$P_{G|W}(g|I_i, word_j) = \sum_{k=1}^{8} \pi_i^k G(I_i, \mu_i^k, \Sigma_i^k) \ , \tag{2.28}$$

where $\pi_i^k, \mu_i^k, \sum_i^k$ are maximum likelihood parameters for image I_i based on mixture component k. Next, by applying hierarchical EM algorithm [41] to the image level mixtures computed in the previous step, class conditional density consisting of a mixture of 64 Gaussians is computed for each word as follows:

$$P_{G|W}(g|w) = \sum_{k=1}^{64} \pi_w^k G(g, \mu_w^k, \Sigma_w^k) \ , \tag{2.29}$$

where $\pi_w^k, \mu_w^k, \Sigma_w^k$ are maximum likelihood parameters for word w based on mixture component k. For a given query image Q, for each word $word_i$, the following conditional log

probability is computed using Bayes rule as follows:

$$\log P_{W|G}(word_i|Q) = \log P_{G|W}(Q|word_i) + \log P_W(word_i) - \log P_G(Q) \ , \qquad (2.30)$$

where $P_W(word_i)$ is taken as the proportion of training set images containing $word_i$ and $P_G(Q)$ is taken as a constant.

This method has been compared with Co-occurence Model, Translation Model and CMRM mentioned in the previous sections. It has the highest reported precision and recall values of 0.23 and 0.29, respectively on Corel2002 data set. The reason for this performance is that there is no mutual independence assumption of annotation words and low level visual features. The class conditional density is computed directly without resorting to joint density estimation. Annotation problem is reduced to a multiclass classification problem, where each class corresponds to an annotation keyword. Class conditional densities are computed directly using hierarchical density model proposed in [41]. Regions of size 8x8 are extracted with a sliding window that moves by two pixels between consecutive frames. Having many local regions increases the information introduced into the system, and provides similar advantages obtained from interest point detectors, where local features are extracted. The method is computationally expensive and has been implemented on a cluster of 3,000 machines.

2.2.2.6 Hierarchical Image Annotation System Using Holistic Approach Model (HANOLISTIC)

In this model [23], image features are extracted from the whole image instead of making use of regions. It uses hierarchical annotation architecture, called HANOLISTIC (Hierarchical Image Annotation System Using Holistic Approach), which consists of two layers. In the first layer, each node computes the probability of a keyword based on fuzzy knn [42] algorithm, according to the distance of the query image to the images in the training set based on a distinct feature such as color structure or edge histogram. In the second layer, called meta layer, the output of these nodes is summed for each word to find the most likely words. Details of the algorithm are given in Algorithm 2.

Surprisingly, this model performs quite well with precision and recall values of 0.35 and 0.24, respectively on Corel2002 data set. This performance is partly due to the nature of the Corel dataset as stated in [37], where many similar images exist in the database with the same

Algorithm 2 Details of the HANOLISTIC algorithm.

1: Compute low level visual features based on each distinct feature.
2: For each distinct low level feature vector, compute annotation probabilities for each annotation word $word_j$ based on fuzzy knn algorithm[42].
3: Feed annotation probabilities computed for each word to the meta layer.
4: In the meta layer, simply sum the annotation probabilities for each word giving each distinct feature equal importance.
5: Output the most likely words as annotation result.

annotation words and the size of the dataset is small.

Although this method performs well on the Corel dataset, it has a generalization problem for image representations when the visual content of the whole image does not match the multiple annotation words. Thus, any change in image content will result in a different image representation, which makes it difficult to obtain invariance to rotations. Although simplicity is a major advantage, as the number of images grows in the dataset, it becomes more and more likely to have two semantically different images having the same global representation. Another disadvantage is the inadequacy of the global representation as the size of the text vocabulary increases. It becomes more and more difficult to represent a variety keywords based on single whole image content as the number of keywords increases.

2.3 Quantization of Visual Features in Image Annotation

One of the major steps in the image annotation is to quantize the visual features, so that one can match the visual features to textual words. A common technique used for this purpose is to cluster the visual features. Clustering has a long history and covers a wide area in pattern recognition. It is defined loosely as the process of organizing a collection of data items into groups, such that elements in each group are more ”similar” to each other than the elements in other groups, according to a similarity metric. Clustering is usually performed in an unsupervised manner without using any additional information other than the data elements themselves.

In this thesis, we propose to use semi-supervised clustering instead of using a standard clustering algorithm for the quantization of the visual features. Hence, this section is devote to

overview the major semi-supervised clustering algorithms.

If additional information is used to guide or adjust the clustering, this process is called semi-supervised clustering. Constraints are usually provided in the form of either ”must-link” constraints or ”can-not link” constraints. The additional information can be incorporated by defining a set of constraints and using these constraints during the clustering. ”Must-link” constraints consist of a set of data point pairs, where the points in the pair indicate that they should belong to the same cluster. Similarly ”cannot-link” constraints consist of a set of data point pairs where the points in the pair indicate that they should belong to different clusters.

Specifically, assume that the set of data points to be clustered is $X = \{x_i\}_{i=1}^{n}$, and the set of K disjoint partitions obtained after clustering is indicated by $\{C_k\}_{k=1}^{K}$, where n is the number of data points and K is the number of clusters. Must-link constraints are indicated by C_{ML} and its elements consist of (x_i, x_j) pairs such that if $x_i \in C_k$ then $x_j \in C_k$, $k = 1..K$ as well. Similarly cannot-link constraints are indicated by C_{CL} and its elements consist of (x_i, x_j) pairs such that if $x_i \in C_k$ then $x_j \notin C_k$ for $k = 1..K$.

There are two types of semi-supervised clustering approaches, namely, search based and distance metric based. In the following subsections, these methods shall be briefly explained.

2.3.1 Search based Semi-supervised Clustering: COP-KMeans Algorithm

In search based semi-supervised clustering approach, the standard clustering algorithm is modified so as to adhere to the constraints provided to the semi-supervisor. Demiriz et. al. [43] use a clustering objective function modified to include a penalty term for not specified constraints. In COP-KMeans algorithm [44], it is enforced that constraints are satisfied during cluster assignment process. In [45], constraint information is used for better cluster initialization. Law et. al. [46] use a graphical model, based on variational techniques.

COP-Kmeans involves two types of constraints: must-link constraints and cannot-link constraints. Must-link constraints indicate that the data elements must belong to the same cluster. Cannot-link constraints are used to provide the necessary information for the two data elements must not belong to the same cluster.

COP-Kmeans algorithm is based on the well known K-means algorithm [38]. The K-means

algorithm uses an iterative refinement heuristic that starts by partitioning the input points into K initial sets. Initial sets are formed either randomly or by making use of some heuristic data. Next, the mean point, or centroid, of each set is calculated. Then, a new partition is obtained by assigning each point to the closest centroid. Then, the centroids are recalculated based on the new partition, and algorithm iterates until convergence, which is achieved when the point assignment to clusters no longer changes the cluster centers. The objective function minimizes the overall distance between the data points and the cluster means. One of the popular objective functions is defined as the Euclidean distance between the samples and the centroids:

$$O = \sum_{i=1}^{K} \sum_{x_j \in C_i} (x_j - \mu_i)^2 \ , \tag{2.31}$$

where K is the number of clusters, C_i indicates partition i, and μ_i is the centroid that corresponds to the mean of all the points $x_j \in C_i$. Finding the global optima for the objective function is known to be NP-complete [47]. Although, there are many different varieties of K-means Clustering, the basic algorithm given below is the simplest and widely used in diverse fields of pattern recognition.

Algorithm 3 Basic K-means Clustering algorithm.

Require: A set of data points $X = \{x_j\}_{j=1}^{n}$.

Ensure: Disjoint k partitions $\{C_i\}_{i=1}^{k}$ satisfying the K-means objective function O.

1: Initialize cluster centroids $\{\mu_i\}_{i=1}^{k}$ at random
2: **repeat**
3: $t \leftarrow 0$
4: Assign each data point x_j to the cluster i^* where $i^* = \underset{i}{\text{argmax}} \ \|x_j - \mu_i^{(t)}\|^2$
5: Re-compute cluster means $\mu_i^{(t+1)} \leftarrow \frac{1}{|C_i^{(t+1)}|} \sum_{x \in C_i^{(t+1)}} x$
6: $t \leftarrow t + 1$
7: **until** convergence

In COP-Kmeans, data point assignment step is modified so that each data point is assigned to the closest cluster which does not violate any constraints. If no such cluster exists, algorithm fails.

Algorithm details of the COP-Kmeans are given in Algorithm 4.

Algorithm 4 The COP-Kmeans algorithm.

Require: A set of data points $X = \{x_j\}_{j=1}^{n}$, must-link constraints C_{ML} , cannot-link constraints C_{CL}.

Ensure: Disjoint k partitions $\{C_i\}_{i=1}^{k}$ satisfying the K-means objective function O.

1: Initialize cluster centroids $\{\mu_i\}_{i=1}^{k}$ at random
2: **repeat**
3: $t \leftarrow 0$
4: Assign each data point x_j to the cluster i^* where $i^* = \underset{i}{\text{argmax}} \|x_j - \mu_i^{(t)}\|^2$ **such that ConstraintViolation(**x_j, C_i, C_{ML}, C_{CL}**) is false**.
5: Re-compute cluster means $\mu_i^{(t+1)} \leftarrow \frac{1}{|C_i^{(t+1)}|} \sum_{x \in C_i^{(t+1)}} x$
6: $t \leftarrow t + 1$
7: **until** convergence

Algorithm 5 ConstraintViolation.

Require: data point x, cluster S, must-link constraints C_{ML}, cannot link constraints C_{CL} .

1: For each $(x; x_{ML}) \in C_{ML}$ If $x_{ML} \notin S$, return true
2: For each $(x; x_{CL}) \in C_{CL}$ If $x_{CL} \notin S$, return true
3: Otherwise, return false

2.3.2 Distance Metric based Semi-supervised Clustering

In distance metric based semi-supervised clustering approach, the distance metric used in the clustering algorithm is trained so as to satisfy the constraints given in semi-supervision. Distance metric techniques used in this approach include Jensen-Shannon divergence trained using gradient descent [48], Euclidean distance metric modified by a shortest-path algorithm [49], Mahalanobis distance metric trained by convex optimization [50], learning a margin-based clustering distance metric using boosting [51], learning a distance metric transformation that is globally linear but locally non-linear [52].

2.3.3 Summary

In this chapter, we provide the background information about the major visual image representation techniques used in image annotation studies, namely, color and texture. We discussed the state of the art image annotation algorithms under two categories: algorithms based on low level visual features that are quantized using a clustering algorithm and algorithms that use continuous low level features. Finally, we focus on the visual feature quantization techniques which is one of the core steps of image annotation. We discuss several techniques for clustering including search based semi-supervised clustering algorithms and distance metric based semi-supervised clustering algorithms.

CHAPTER 3

SSA: SEMI SUPERVISED ANNOTATION

In this chapter, we introduce a new technique for image annotation, which improves the representation of low level visual features to get visterms. The proposed technique, called Semi Supervised Annotation (SSA), is based on the assumption that there is already available ”side information” in the annotation system which is not utilized by the annotator. Therefore, this side information can be utilized to improve the performance by decreasing the randomness of the overall system. The side information can be added to the annotation system by semi-supervising the clustering process of the visual information extracted from the image regions, which is expected to sharpen the probability density function of each visterm.

The concept of semi-supervised clustering and making use of quantized image regions have been introduced in Chapter 2. Now, we propose to use the semi-supervised clustering for quantizing image regions. Our motivation is to guide the clustering of visual features using the extra available side information.

At this point the crucial question needs to be answered is how to define and formalize the ”side information”. As an example, one such information, may be text labels to infer the concepts and using these concepts to guide the clustering of visual features. While constructing visual words based on a specific feature, a potential guidance may come from making use of other related visual features.

In the following sections, SSA is explained in detail. In Section 3.1, the image annotation problem, in the framework of our proposed system, is formalized. Region selectors and low level features used in our system are described in Section 3.2. Then, in Section 3.3 the proposed semi-supervised clustering algorithm, which clusters the low level image features using ”side information” is explained. Parallel version of the semi-supervised clustering algorithm

is described in Section 3.6. Finally, computational complexity for SSA is discussed in Section 3.7.

Part of the work presented in this thesis, has already appeared in [53], [54], and [55].

Table 3.1: Nomenclature.

S	Training set
s	Size of the training set
$S_j = (I_j, D_j)$	Pair of image j and text document j
I_j	Image j
D_j	Text document j
W	Dictionary
w	Size of the dictionary
$Word_i$	ith word from the dictionary
w_i	Binary variable indicating whether word $Word_i$ appears in associated text document
RS	Set of region selector algorithms
a	Number of region selector algorithms
RS_i	Region selector algorithm i
T_k	The set of visual feature types for region selector RS_k
t_k	Number of visual feature types used for region selector RS_k
$FeatureType_{ki}$	ith feature type for region selector RS_k
I_{ji}	Set of visual features obtained from region selector RS_i based on feature type j
F_{jkl}	Visual features extracted from image j based on visual feature $FeatureType_{kl}$ using region selector RS_k
F_{jklm}	Visual feature obtained from the mth region or point of kth region selector for image I_j based on visual feature type T_{kl}
V_j	Sets of visterms obtained by quantizing the low level visual features found under all region selectors for image j
V_{jk}	Visterms obtained from low level features under region selector RS_k
V_{jkl}	Set of visterms extracted from I_j based on visual feature $FeatureType_{kl}$ using region selector RS_k
Q	Query image
$N(Q)$	Number of regions in query image Q
r_{Qm}	mth region in image Q
FQ	Visual features corresponding to regions of query image Q
$FQ_{Q,i}$	Visual features corresponding to ith region in query image Q
A	Annotation keywords of query image Q
$K(A)$	Number of annotation keywords for query image Q
$w_{A,i}$	ith annotation keywords for query image Q

3.1 Image Annotation Problem

In this section, we shall formalize the Image Annotation problem, for the development of the proposed system, Semi Supervised Annotation presented in the subsequent sections.

Mathematically speaking, the training set S, consists of s image and text document pairs, as follows;

$$S = \{(I_1, D_1), (I_2, D_2), ..., (I_s, D_s)\} , \tag{3.1}$$

where I_j and D_j corresponds to the image and associated text document of the jth pair of the training set. Each text document D_j consists of words obtained from a dictionary, W,

$$W = \{Word_1, Word_2, ..., Word_w\} , \tag{3.2}$$

where w is the size of the dictionary W, and

$$D_j = \{w_1, w_2, ..., w_w\} , \tag{3.3}$$

where w_i is a binary variable indicating whether word $Word_i$ appears in associated text document D_j of the jth pair of the training set or not.

Each image I_j consists of visual features obtained from potentially overlapping regions or points generated from a set of segmentation or regions of interest detector algorithms, such as normalized cut segmentation algorithm [14] or Difference of Gaussians (DoG) point detector [29]. Let us call these algorithms Region Selectors and define the set of such algorithms as:

$$RS = \{RS_1, RS_2, ..., RS_a\} , \tag{3.4}$$

where a is the number of region selectors.

A set of visual feature types T_k is used for each region selector RS_k. Let, the number of visual feature types used for region selector RS_k be t_k. Define the set of visual feature types used for region selector RS_k as:

$$T_k = \{FeatureType_{k1}, FeatureType_{k2}, ..., FeatureType_{kt_k}\} . \tag{3.5}$$

Image I_j consists of a many sets of visual features obtained from the region selectors.

$$I_j = \{I_{j1}, I_{j2}, ..., I_{ja}\} , \tag{3.6}$$

where I_{jk} corresponds to visual features obtained from region selector RS_k. Let,

$$I_{jk} = \{F_{jk1}, F_{jk2}, ..., F_{jkt_k}\} , \tag{3.7}$$

where F_{jkl} indicates the visual features, extracted from I_j, based on visual feature $FeatureType_{kl}$ using region selector RS_k. The number of visual features employed by a visual feature type T_{kl} for an image I_j is denoted by f_{jkl}, and the set of visual features obtained from an image I_j based on visual feature type T_{kl} using region selector RS_k is shown by:

$$F_{jkl} = \{Feature_{jkl1}, Feature_{jkl2}, ..., Feature_{jklf_{jkl}}\} . \tag{3.8}$$

$Feature_{jklm}$ corresponds to the low level feature obtained from the *m*th region or point of *k*th region selector for image I_j, based on visual feature type T_{kl}.

V_j consists of sets of visterms obtained by quantizing the low level visual features found under all region selectors,

$$V_j = \{V_{j1} \cup V_{j2} \cup ... \cup V_{ja}\} . \tag{3.9}$$

where V_{jk} corresponds to visterms obtained from low level features under region selector RS_k.

Let,

$$V_{jk} = \{V_{jk1} \cup V_{jk2} \cup ... \cup V_{jkt_k}\} , \tag{3.10}$$

where V_{jkl} indicates the set of visterms extracted from I_j based on visual feature $FeatureType_{kl}$ using region selector RS_k :

$$V_{jkl} = \{Visterm_{jkl1}, Visterm_{jkl2}, ..., Visterm_{jklf_{jkl}}\} . \tag{3.11}$$

Given a query image Q where $Q = \{r_{Q1}, r_{Q2}, ..., r_{QN(Q)}\}$, $N(Q)$ is the number of regions in image Q, and low level visual features corresponding to regions are denoted by FQ where $FQ = \{FQ_{Q,1}, FQ_{Q,2}, ..., FQ_{Q,N(Q)}\}$, image annotation is defined as finding a function

$$F(Q, FQ) = A$$

, where $A = \{w_{A,1}, w_{A,2}, ..., w_{A,K(A)}\}$, $K(A)$ is the number of words in annotation A and $w_{A,i}$ is obtained from dictionary W.

Nomenclature table corresponding to the notations used in this section is given in Table 3.1.

After the above formal representation of the image annotation problem, in the following sections, we formally introduce the necessary concepts such as image document, text document,

text dictionary, visual dictionary, region selectors and low level visual features and their relationships.

3.2 Region selectors and Visual Features

In chapter 2, we have discussed the available region selectors and feature spaces for image annotation in the literature. The design of the feature spaces in pattern recognition problems is still an art rather than an engineering issue and depends on the application domain. The selection of feature spaces has a great impact on the performance of the image annotation problem. In this thesis, we did not focus on developing new feature spaces , but we investigate the same three region selectors and feature spaces that have been used in the state of the art image annotation algorithm of [20], to be able to compare the proposed algorithm SSA to that of [20]. The first one is normalized cut segmentation introduced by [14]. The second one is uniform grid, which divides the image into a set of uniform regions [20]. The last one is Difference of Gaussians (DoG) point detector [29]. We employ different sets of visual features for each region selector, which is explained below.

3.2.1 Visual Features for Normalized Cut Segmentation

Blob features obtained from the regions extracted by the Normalized Cut Segmentation method, originally used in [30] consists of a combination of size, position, color, texture and shape visual features that are represented in a 40 dimensional feature vector. The low level visual features used in Blob Feature are given in Table 3.2.

There are many studies [16], [30], [15] [17], [20], which use and investigate the pros and cons of the blob features. Concatenation of all the incompatible color, texture and shape features yields a high dimensional and sparse vector space. In our opinion, this feature space bears many problems, such as curse of dimensionality and statistical instability. However, in order to make our results compatible with that of [20], we used blob features.

Table 3.2: Low level visual features used in Blob Feature.

Low level feature	**Description**	**Dimension**
Size	Portion of the image covered by the region	1
Position	Coordinates of the region center	1
Ave RGB	Average of RGB	3
Ave LAB	Average of LAB	3
Ave rg	Average of rg, where r=R/(R+G+B), g=r=G/(R+G+B)	3
RGB stddev	Standard deviation of RGB	3
LAB stddev	Standard deviation of LAB	3
rg stddev	Standard deviation of rg, where r=R/(R+G+B), g=r=G/(R+G+B)	3
Mean Oriented Energy	12 Oriented filters in 30 degree increments	12
Mean Difference of Gaussians	4 Difference of Gaussians Filters	4
Boundary/area	ratio of the area to the perimeter squared	1
Moment-of-inertia	the moment of inertia about the center of mass	1
Convexity	ratio of the region area to that of its convex hull	1
	Total	**40**

3.2.2 Visual Features for Grid Segmentation

We use Hue-Saturation (HS) feature after dividing the image into rectangles using a uniform grid as in [20]. To obtain illumination invariance, color brightness value is discarded from the Hue-Saturation-Value (HSV) color space. A two-dimensional histogram is obtained by quantizing the Hue and Saturation values separately.

3.2.3 Visual Features for Interest Points

We use three types of visual features for interest points under Difference of Gaussians region selector. The first one is the orientation value assigned to each interest point. Orientation information is lost in standard SIFT descriptor, since the interest point is aligned along the dominant orientation direction. Although this approach maintains the rotation invariance, some valuable information is lost for objects that are usually displayed in a known orientation direction or when a similar local structure is displayed in different orientations on the same scene.

The second visual feature we use is color information around the interest point as in [28]. Since SIFT descriptor does not have any color content, it is reasonable to associate SIFT descriptor with color. This approach captures the texture information created by certain colors. LUV color space is chosen because of its perceptual property arising from the linearization of the perception of the color distances, and it is known to work well in image retrieval applications [28], [56], [57]. LUV values are computed on a window normalized to cover the area given by interest point descriptor. The mean and standard deviation values are computed along each color space dimension and concatenated under the same vector. Each entry of this vector is normalized to unit variance to avoid domination of luminance.

The third visual feature we use under Difference of Gaussians is the standard SIFT descriptor [29]. SIFT features are extracted using the local histograms of edge orientation from each interest point. SIFT features are robust to occlusion and clutter and have the ability to generalize to a large number of objects, since they are local.

The visual features, obtained above or other available feature extraction algorithms, enable us to characterize the low level visual content of the image to a certain extent. These rep-

resentations bear several problems: First of all, the high dimensional feature spaces require combinatorially explosive number of samples to yield a statistically stable data set, which is practically impossible. Reduction of dimension is employed in some of the systems, but this time there is a tradeoff between the information loss and statistical stability. Even if we create a very high dimensional vector space, the low level features are far from representing the high level concepts carried under the annotation words. A third problem comes from the locality of the visual features. This local information extracted from a region and/or around an interest point is not one-to-one neither onto with the textual words. In order to improve the common image annotation systems one need to attack the problems mentioned above.

There is a tremendous amount of studies to create a feature space, well suited to a specific application domain [2]. In this thesis, we approach the above mentioned problems from an information theoretic point of view. Given a set of low level features FQ and a dictionary W, the annotation function $F(Q, FQ) = A$ for a query image Q, requires a labeling process for the regions of the vector space created by FQ. At this point, most of the image annotation algorithms cluster low level visual features. The clustering process does not only label the low level features with high level concepts but also, enables a more compact image representation and a lower computational complexity [13], [14], [16], [20]. The crucial point is how to cluster the low level image features to represent high level document words. One may improve the clustering process by employing a type of supervision, called semi-supervised clustering. In the following sections, we introduce the concept of ”side information”, discuss the difference between commonly used clustering algorithm of K-means and the proposed method of semi-supervised clustering, using side information. We, finally, describe, the code book construction methods using the proposed semi-supervised clustering method.

3.3 Side Information for Semi Supervision

In a general sense, side information can be defined as any kind of information, which is already available, but not used in the clustering process of low level visual features. Side information is already in annotation system, but it is somehow neglected or unused in the clustering of visual features. It can be based either on visual features or on annotation keywords. We classify side information into two groups based on whether it is obtained from the whole image or from the image regions. If side information is obtained from the whole image

or annotation keywords, we call it as global, if it is obtained from image regions or interest points we call it as local side information.

We use side information in such a way that, while clustering visual features, those with the same side information are constrained to fall in the same clusters. By grouping visual features with the same side information together, we hope to obtain clusters that are more homogeneous with respect to the provided side information. Therefore, we expect to have clusters with sharper probability density functions. Consequently, distributions of visual features become less random resulting in better annotation performance.

We quantize side information by clustering the side information features to obtain groups corresponding to each cluster label. For each side information, we define two functions. First function assigns each visual feature to a group or a set of groups depending on the specific side information, the visual feature associated with. This function is side information specific. Second function assigns a visual cluster to a group or a set of groups.

Mathematically speaking, let us assume $SI = \{SI_i\}_{i=1}^{si}$ denote the set of side information, we employ and we have *si* many different types of them. For each side information SI_i, $i = 1..si$ we assume cluster labels are grouped into g_{SI_i} many categories. Although it can be done in a variety of different ways, we simply assign visual clusters to groups so that each group is assigned approximately equal number of clusters. More specifically, for a region r_{jm}, and its associated side information SI_{ijm}, we have a function performing region assignment $RegionAssignment_{SI_i}(r_{jm}) = G_{ijm} \subset \{1, ..., g_{SI_i}\}$, and a function performing cluster assignment for a cluster C_k, $ClusterAssignment_{SI_i}(C_k) = g$, $g = 1..g_{SI_i}$. Note that if SI_i is a global side information, SI_{ijm} is same for all the regions r_{jm} within an image. Otherwise SI_{ijm} is obtained from the region corresponding to r_{jm}.

3.3.1 Representation of Side Information

Although it can be formulated in many different ways, in this thesis, we define three different types of side information. The first one comes from the text document consisting of annotated keywords associated with images. Since this side information is global, the same side information is associated with visual features extracted from all the regions of a given image. We quantize this side information by obtaining hidden topic probabilities from the PLSA algo-

rithm proposed in [26], so that each hidden topic corresponds to a group in our terminology. We assign visual features to only ”highly likely” topics (groups). Highly likely topics are determined by K-means clustering applied to the topic probabilities obtained for an image through PLSA algorithm where K is chosen as 2, corresponding to ”likely” and ”not likely” topics, in a sense acting as a threshold.

The second side information we define, is the orientation information around each interest point. This side information is used for supervising the clustering of SIFT features. Orientation information is readily available in Difference Of Gaussians region selector. Orientation of an interest point is computed as follows. For an interest point at pixel $P(x,y)$ at region r_{jm}, orientation side information is computed as follows [29]:

$$SI_{ijm} = \theta(x,y) = tan^{-1}((P(x,y+1) - P(x,y-1))/(P(x+1,y) - P(x-1,y))) \ . \qquad (3.12)$$

Next, an orientation histogram is computed from these gradient orientations of sample points that are within a region around the interest point. The orientation histogram consists of 36 bins covering the 360 degree range of orientations. Then, each sample is weighted by its gradient magnitude and by a Gaussian-weighted circular window with a scale which is 1.5 times of the interest point. Dominant directions of these local gradients are found by choosing the peaks in the orientation histogram. The side information, corresponds to the dominant direction θ computed for each interest point. We quantize the orientation θ into NO number of bins as follows:

$$orientation = 1 + round((\theta + \pi)/(2 * \pi) * (NO - 1)) \ . \qquad (3.13)$$

The assignment of a SIFT feature to an orientation group is computed directly using the above formula.

The third side information we use, is the color information around each interest point. This side information is used for supervising the clustering of SIFT features as well. Color information around each interest point is obtained by computing LUV color features around interest points as discussed in [28]. First, LUV values are computed on an 11x11 grid normalized to cover the local area given by the interest point detector resulting in a feature vector of dimension 121. Next, the mean and standard deviation for each LUV color dimension is computed and concatenated resulting in a 6-dimensional vector. Finally, each dimension of this vector is normalized to unit variance. The quantization of this color information into NC number of bins is made through K-means algorithm choosing K as NC. A SIFT feature is

assigned to a color group by simply choosing the nearest group based on Euclidean distance of the color information around its interest point.

One may ask why we choose the above mentioned features to define the side information. Unfortunately, at this point, we have no formal answer to this question, nor we have a systematic way of selecting and defining the side information. However, intuitively speaking, all the above mentioned features provide extra information to guide and bring constraints to the clustering process. This extra side information somehow narrows the semantic gap between the visual and textual features. It should be noted that there is no unique and complete definition of side information for a given image representation.

3.4 Semi-supervised Clustering versus Plain Clustering for Visual Feature Quantization

As it is mentioned before, in most of the image annotation methods the visual features are clustered to obtain visterms using a standard K-means algorithm. In this method, data points are distributed to K clusters in such a way that each data point belongs to the cluster with the nearest mean. Hence, all the information we use in K-means clustering is the low level visual features. If we can provide more information to the clustering process, indicating whether two data points co-exist in the same cluster or not, we can get better clustering results as reported by recent research on semi-supervised clustering [43], [44], [45], [46], [48], [50], [51], [52].

The important question is how to define and feed this information to the clustering process. It is not feasible to get this information from the users. However, one should note that there exists some implicit information in annotated images besides the visual features, which might consistently co-occur with the visual feature to be clustered, such as annotation keywords, position of low level visual features and information regarding other low level visual features. This additional information called ”side information” can be used in providing the constraints to the semi-supervised clustering process automatically.

We can embed the available side information to the clustering process of visual features as follows. First, we represent and quantize the available side information, by clustering side information features collected from the annotated images to obtain groups, where each side information cluster label corresponds to a group. Quantization of side information can be

done in many ways. As this can be done with standard K-means algorithm, other hard or soft clustering [58], [59] methods can be used as well.

Next, each visual feature data point F_{jm}, associated with the co-existing side information SI_{ijm}, are assigned to a group or set of groups G_{ijm}. Finally, we constrain the visual feature clustering process with the available side information so that visual points that fall in the same cluster should all have the same group label assignments.

Recently, there have been attempts to improve clustering methods employing some constraints [44]. If this additional information is used to guide or adjust the clustering, this process is called semi-supervised clustering. There are two types of semi-supervised clustering approaches, namely, search based and distance metric based. In search based semi-supervised clustering approach, the clustering algorithm is modified so as to adhere to the constraints provided to the algorithm. In distance metric based semi-supervised clustering approach, the distance metric used in the clustering algorithm is trained so as to satisfy the constraints given in semi-supervision.

The closest approach of semi-supervised clustering to ours is COP-Kmeans [44], where constraints are provided in the form of must-link and cannot-link constraints specifying that two visual features must belong to the same cluster and two visual features must not belong to the same cluster, respectively. Our approach in providing constraints is different than [44] in the sense that in [44], a must-link constraint between two data points indicate that they belong to the same cluster, in our case, assigning group label(s) to each data point provides a constraint that the data point belongs to one of the clusters labeled with its assigned group(s). We do not use any cannot-link constraints.

We gain two major benefits by using the available ”side information” in the annotated images besides the visual features that are clustered. First, it is expected that clusters become more homogeneous with respect to the provided side information. Therefore, clusters have sharper probability density functions resulting in less overall entropy of the system and the distribution of visual features being clustered becomes less random. By decreasing the entropy of the overall system, we hope to increase the annotation performance. Second, we reduce the search space during clustering since we compare a visual feature with not all of the cluster centers but with only centers of those clusters that are assigned to the visual feature based on its associated side information. Therefore, we get better performance as far as the computational

complexity of the clustering is concerned.

In the next sections, we describe how to obtain code books using this side information through semi-supervised clustering.

3.5 Code Book Construction by Semi Supervised Clustering

Our code-book construction method for visterms is a modified version of K-Means to include semi-supervision. We constrain the clustering by employing the side information. For this purpose, we determine the groups with the same side information and enforce the clustering algorithm to assign the visual features to only one of the clusters within the same group or groups determined according to the available side information. Therefore, this method constrains the visual feature clustering process with the available side information so that visual points that fall in the same cluster should all have the same group label assignments.

Initially the total number of groups is chosen as the number of classes in the side information. Next, we simply assign visual clusters to groups, so that each group is assigned to approximately equal number of clusters, assuming visual features that co-exist with each side information group have equal chance of being assigned to any of the visual clusters. Note that, other variations such as assigning clusters to groups based on their number of occurrence in the training set could be used as well.

Next, each visual feature F_{jm}, associated with the co-existing side information SI_{ijm}, are assigned to a group or set of groups G_{ijm}. The visual feature F_{jm} is assigned to the nearest or k-nearest of the g_{SI_i} groups with respect to a distance metric, such as the Euclidean distance of the side information feature SI_{ijm}, to group cluster centers. The rest of the algorithm applies a modified version of the standard K-means algorithm. Initially, visual features are included randomly in of the clusters that are assigned to any of G_{ijm}. Then, mean of each cluster is computed. Next, each visual feature is included in the closest cluster that is assigned to any of G_{ijm}. Iteration continues until the convergence. The details of the method are given in Algorithm 6.

Once a visual codebook is constructed, visual features F_{jm} of query images are assigned to the codebook depending on the type of the co-existing side information SI_{ijm}. If the side

Algorithm 6 Code Book Construction using Semi-supervised Clustering Algorithm.

Require: A set of data points $X = \{Feature_{jm}\}$, $j = 1..s$, $m = 1..f_j$, extracted from regions r_{jm} where f_j is the number of regions in image I_j, each point corresponds to low level visual feature obtained from region m of image I_j, text document D_j associated with image I_j, given side information SI_i.

Ensure: Disjoint K partitions $\{C_k\}_{k=1}^{K}$ satisfying the K-means objective function O.

1: Choose total number of groups g_{SI_i} depending on side information SI_i

2: Label each cluster $C_k, k = 1..K$ with one of the g_{SI_i} groups so that each group has approximately equal number of clusters, where K is the total number of clusters using cluster assignment function $ClusterAssignment_{SI_i}$.

3: Construct a set of group label(s) G_{ijm} based on $RegionAssignment_{SI_i}(r_{jm})$ that each visual feature $Feature_{jm}$ can be assigned.

4: Assign each $Feature_{jm}$ randomly to one of the clusters labeled with one of the groups within G_{ijm}.

5: **repeat**

6: Re-compute cluster means

$$\mu_k \leftarrow \frac{1}{|C_k|} \sum_{x \in C_k} x \tag{3.14}$$

7: Assign each $Feature_{jm}$ to the nearest cluster labeled with one of the groups corresponding to G_{ijm} as follows: Using Euclidean Distance function d, compute $d(Feature_{jm}, \mu_k)$ for $k = 1..K$. Assign $Feature_{jm}$ to k^* where

$$d(Feature_{jm}, \mu_{k^*}) <= d(Feature_{jm}, \mu_k), k = 1..K.$$

8: **until** no feature to cluster assignment changes

9: **if** Side information SI_i is based on annotation keywords **then**

10: Apply Linear Discriminant Analysis to the clustering results so as to obtain a transformation matrix U as explained in Subsection 3.5.0.1

11: Update cluster centers μ_k and test image visual Blob features based on U.

12: **end if**

information is based on visual feature F_{jm}, then it is assigned to the nearest cluster within the groups G_{ijm} based on the side information feature SI_{ij}. If the side information is based on the annotation keywords, visual feature F_{jm} is assigned to the nearest cluster within the codebook.

Since during codebook construction, a visual feature is assigned to only one of the clusters labeled with its assigned group, it is not assigned to the closest cluster among all the clusters, but only to one of the clusters under the same group as its side information. In effect, clusters with different assigned group labels might have means that are close in Euclidean Space as opposed to the cluster means that are computed through standard K-means clustering algorithm.

An example of the block diagram representation for the feature assignment is shown in Figure 3.1. In this example, we have 8 groups for 8 distinct classes of the side information SI_i , corresponding to 8 directions. We have 32 visual clusters. Each group is assigned to 4 clusters. Visual feature F_{j1} is assigned to group 2, since its co-existing side information SI_{ij1} is closest to group 2. Visual feature F_{j2} is assigned to group 1, since its co-existing side information SI_{ij2} is closest to group 1. During codebook construction we compare visual feature F_{j1} with only cluster centers 5 through 8, and visual feature F_{j2} with only cluster centers 1 through 4. Therefore, in this example it is possible that any of the cluster centers 1 through 4, might be close in Euclidean space to any of the cluster centers 5 through 8. This possibility does not create any problem for query image features associated with visual side information, since they are compared with clusters only assigned to them. However, since visual features of query images associated with textual side information are compared with all the clusters in the codebook, we need a mechanism to separate clusters that are assigned to different groups.

In case the side information is based on textual annotation words, to keep clusters as apart from each other as possible, and visual features within each cluster as close in Euclidean distance as possible, we apply Linear Discriminant Analysis to the clustering results in order to obtain a transformation matrix that we further apply to the visual features of the test image. Details of the Linear Discriminant Analysis method are given in the next subsection.

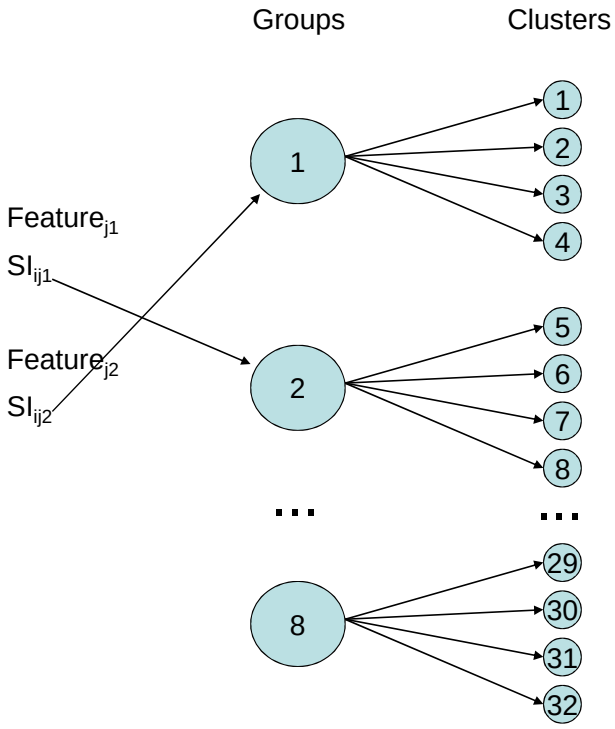

Figure 3.1: The block diagram for a sample cluster assignment to groups.

3.5.0.1 Linear Discriminant Analysis for Projection of Visual features

Given a set of features $Feature^{ci}$, where c is the cluster label obtained by semi-supervised clustering, $c = 1..K$, and i is the sample id in the cth cluster, $i = 1..n_c$ and n_c is the number of data points within cth cluster. Our goal for using Linear Discriminant Analysis (LDA) is to find a projection of the visual features that separate the clusters as much as possible while keeping the visual features within clusters as close as possible.

Let, μ_c denote the mean of the individual cluster c, where μ_c can be computed using the following equation:

$$\mu_c = (1/n_i) \sum_{i=1}^{n_c} Feature^{ci} \ . \tag{3.15}$$

The overall mean becomes:

$$\mu = (1/n) \sum_{c=1}^{K} \sum_{i=1}^{n_c} Feature^{ci} \ , \tag{3.16}$$

where $n = \sum_{c=1}^{K} n_c$.

The within-class scatter matrix M_w and the between-class scatter matrix M_b can be computed

as follows:

$$M_w = (1/n) \sum_{c=1}^{K} n_c \sum_{i=1}^{n_c} (Feature^{ci} - \mu_c)(Feature^{ci} - \mu_c)^T \ , \tag{3.17}$$

$$M_b = (1/n) \sum_{c=1}^{K} n_c (\mu_c - \mu)(\mu_c - \mu)^T \ . \tag{3.18}$$

Our goal is to find a transformation matrix U, such that

$$U^* = \underset{U}{\text{argmax}} \ \frac{|U^T M_b U|}{|U^T M_w U|} \ . \tag{3.19}$$

The projection matrix U^* can be computed from the eigenvectors of $M_w^{-1} M_b$. Projected new cluster means for the visual code-book, and projected feature vectors for test images are computed using:

$$\mu_c = U^T (\mu_c - \mu) \ , \tag{3.20}$$

$$Feature_{jm} = U^T (Feature_{jm} - \mu) \ . \tag{3.21}$$

This transformation makes the clusters as apart from each other as possible while keeping the features within a cluster as close as possible.

3.5.1 SSA-Topic: Semi-supervised Clustering Using Text Topic Information as Side Information

As it is mentioned in Section 3.3, one of the methods to define the side information is to use the text topic probabilities extracted from the PLSA algorithm. These probabilities may guide the clustering process to quantize the visual information to improve the relationship between the text topic information and the visual information. It is expected that this approach decreases the semantic gap between the visual and the textual words.

In this method, we determine the groups to which the visual features will be assigned based on the text topic information and enforce the clustering algorithm, such that visual features are assigned only to one of the clusters labeled with the assigned group or groups.

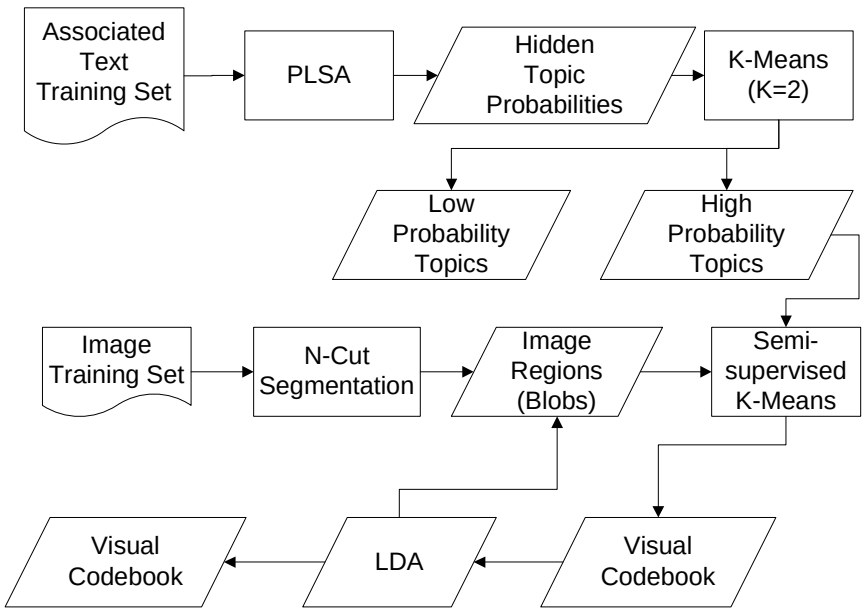

Figure 3.2: Flow chart for SSA-Topic.

The flowchart of the SSA-Topic algorithm is given in Figure 3.2. Note that SI_{ijm} is same for all F_{jm} since the side information is based on annotation keywords. Since each annotation potentially consists of several topics, we use a set of groups instead of just one group for feature assignment. To compute the set of groups to be assigned to visual features, initially image annotations are fed into PLSA algorithm that has been discussed in Chapter 2. Using this algorithm, the hidden topic probabilities are computed for each image annotation. Using a standard K-means algorithm and choosing K as 2, two sets of hidden topic probabilities are obtained that correspond to ”high” and ”low” probability topics. These high probability topics correspond to the set of groups for visual feature assignment. Note that, for feature assignment, we could select the most likely k topics based on an arbitrary number k, or could cluster topic probabilities by choosing K as greater than 2 instead of 2 before choosing the cluster that corresponds to ”high” probability topics. The reason for choosing number of topic probability clusters as 2, is simply to use clustering as a threshold to choose ”high” probability topics. Image regions are obtained from N-Cut (normalized cut) segmentation algorithm [14]. Low level features called ”Blob features” corresponding to these regions are computed as discussed in sub-section 3.2.1, to obtain Blob features. These features are clustered using SSA-Topic Algorithm details of which are given in Algorithm 7 that takes as input ”high” probability topics obtained from image annotations.

The Semi-supervised Clustering algorithm using text topic information is a specific instance of the general Semi-supervised Clustering algorithm introduced in Section 3.5. The details of the algorithm are given in Algorithm 7. Since, the side information is based on the textual keywords, Linear Discriminant Analysis is applied to the clustering results.

The above mentioned semi-supervised clustering can be considered as both search based and distance metric based, since it not only guides the clustering, but also improves the distance metric by transforming the feature space through application of Linear Discriminant Analysis algorithm.

3.5.2 Semi-supervised Clustering Using Complementary Visual Features as Side Information

As it is explained in the previous subsection, topic probabilities serve as side information to yield ”better” clusters in terms of text topics. Another way of using side information may be to accentuate some of the visual information in the image regions. The choice of the complementary visual feature depends on the database and application domain. For example, in Corel data set, the objects are heavily described by color and/or orientation information. It is expected that using additional color and/or orientation information to semi-supervise the clustering process improve the homogeneity of the clusters with respect to orientation or color.

This method is a type of search based Semi-supervised Clustering method, since it only guides the clustering, but does not improve the distance metric as opposed to the SSA-Topic method introduced in the previous section, where the feature space is changed by the application of Linear Discriminant Analysis algorithm. Grouping clusters based on side information adds another dimension to visual code-book so that clusters encode not only the visual information but also the side information attached to these visual features.

The flowchart of the Semi-supervised Clustering using Orientation, as side information (SSA-Orientation) method and Semi-supervised Clustering using Color Information as side information (SSA-Color) method are given in Figure 3.3 and in Figure 3.4, respectively. For both of these methods, interest points are found out automatically by difference of Gaussians (DoG) point detector [29] as discussed in Section 2.1.1.2.

For SSA-Orientation method, two types of information are employed from each interest point.

Algorithm 7 SSA-Topic Algorithm.

Require: A set of data points $X = \{Feature_{jm}\}$, $j = 1..s$, $m = 1..f_j$, where f_j is the number of visterms in image I_j, each point corresponds to low level Blob feature obtained from region m of image I_j after Normalized Cut Segmentation, text document D_j associated with image I_j.

Ensure: Disjoint K partitions $\{C_k\}_{k=1}^{K}$ satisfying the K-means objective function O.

1: Set number of groups g as the possible number of topics T so that groups are represented by topic numbers. We use group and topic interchangeably within this algorithm.

2: Label each cluster $C_k, k = 1..K$ with one of the T groups so that each group has approximately equal number of clusters, where K is the total number of clusters.

3: Using PLSA method of Subsection 2.2.1.4 , for each D_j compute topic probability P_{jk} where $j = 1..s$, $k = 1..T$, T is the number of topics.

4: Using a standard K-means algorithm on the topic probabilities computed in the previous step, and choosing K=2 to act as a threshold to find high and low topic probabilities, find out C_{j1} and C_{j2} sets. If the mean of set C_{j1} is higher than the mean of C_{j2}, take the likely topics $G_{ijm} = C_{j1}$, otherwise take $G_{ijm} = C_{j2}$ where $j = 1..s$, for image I_j associated with text document D_j.

5: Assign each $Feature_{jm}$ randomly to one of the clusters labeled with one of the groups within G_{ijm}.

6: **repeat**

7: Re-compute cluster means

$$\mu_k \leftarrow \frac{1}{|C_k|} \sum_{x \in C_k} x \tag{3.22}$$

8: Assign each $Feature_{jm}$ to the nearest cluster labeled with one of the groups corresponding to G_{ijm} as below: Using Euclidean Distance function d, compute $d(Feature_{jm}, \mu_k)$ for $k = 1..K$. Assign $Feature_{jm}$ to k^* where

$$d(Feature_{jm}, \mu_{k^*}) <= d(Feature_{jm}, \mu_k), k = 1..K.$$

9: **until** no feature to cluster assignment changes

10: Apply Linear Discriminant Analysis to the clustering results so as to obtain a transformation matrix U as explained in Subsection 3.5.0.1

11: Update cluster centers μ_k and test image visual Blob features based on U.

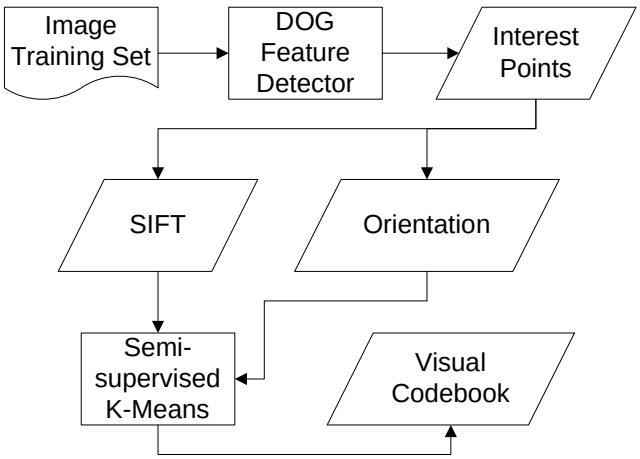

Figure 3.3: Flow chart for SSA-Orientation.

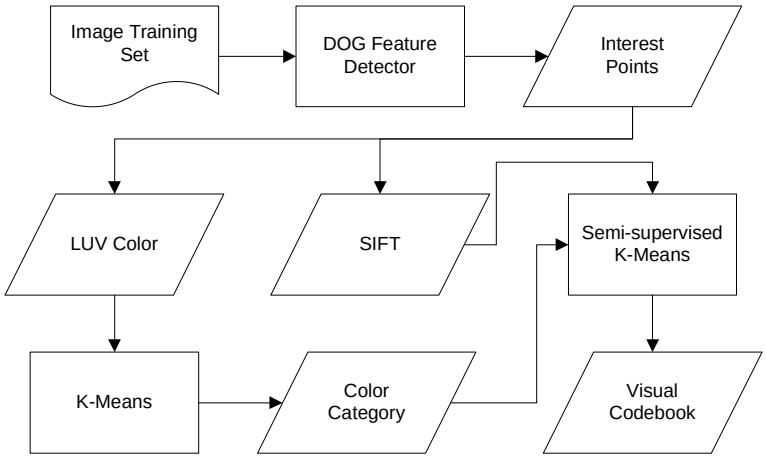

Figure 3.4: Flow chart for SSA-Color.

First is the dominant orientation of the interest point, second is the SIFT descriptor as described in Chapter 2. Note that SIFT descriptor does not carry orientation information since, interest points are normalized along the most dominant direction to obtain the descriptor.

For SSA-Color method, two types of information are used for each interest point. First information is the color category around each interest point, obtained through K-means clustering of LUV color features around interest points as discussed in [28]. Second information is the SIFT descriptor as explained in Section 2.1.1.2.

Note that, SSA-Orientation and SSA-Color algorithms are specific instances of the general Semi-supervised Clustering algorithm proposed in this study. The details of the SSA-

orientation method and SSA-Color method are given in Algorithm 8 and in Algorithm 9, respectively.

For SSA-Orientation, total number of groups corresponds to the total number of possible orientation values. RegionAssignment function for each SIFT feature constructs the set of group labels using only one element that correspond to the orientation value of the interest point corresponding to the SIFT descriptor.

Algorithm 8 SSA-Orientation Algorithm.

Require: A set of data points $X = \{Feature_{jm}\}$, $j = 1..s$, $m = 1..f_j$, where f_j is the number of SIFT features in image I_j, each point corresponds to low level mth SIFT feature obtained from image I_j after Difference of Gaussians feature detection, for each SIFT feature, $Orientation_{jm}$ values corresponding to the orientation of the interest point.

Ensure: Disjoint k partitions $\{C_k\}_{k=1}^{K}$ satisfying the K-means objective function O.

1: Set number of groups g as the possible number of orientations so that groups are represented by orientation numbers. We use group and orientation interchangeably within this algorithm.

2: Label each cluster $C_k, k = 1..K$ with one of the g groups so that each group has approximately equal number of clusters, where K is the total number of clusters.

3: Assign each $Feature_{jm}$ randomly to one of the clusters labeled with $G_{ijm} = \{Orientation_{jm}\}$.

4: **repeat**

5: Re-compute cluster means

$$\mu_k \leftarrow \frac{1}{|C_k|} \sum_{x \in C_k} x \tag{3.23}$$

6: Assign each $Feature_{jm}$ to the nearest cluster labeled with one of the groups labeled with $G_{ijm} = \{Orientation_{jm}\}$ as below: Using Euclidean Distance function d, compute $d(Feature_{jm}, \mu_k)$ for $k = 1..K$. Assign $Feature_{jm}$ to k^* where

$$d(Feature_{jm}, \mu_{k^*}) <= d(Feature_{jm}, \mu_k), k = 1..K.$$

7: **until** no feature to cluster assignment changes

For SSA-Color, total number of groups correspond to the color category quantization level obtained from K-Means clustering of LUV color feature around interest points as explained

in Section 3.3. RegionAssignment function for each SIFT feature constructs the set of group labels using only one element that correspond to the color category of the interest point corresponding to the SIFT descriptor.

Algorithm 9 SSA-Color Algorithm.

Require: A set of data points $X = \{Feature_{jm}\}$, $j = 1..s$, $m = 1..f_j$, where f_j is the number of SIFT features in image I_j, each point corresponds to low level mth SIFT feature obtained from image I_j after Difference of Gaussians feature detection, for each SIFT feature, LUV_{jm} values corresponding to the color of the interest point.

Ensure: Disjoint k partitions $\{C_k\}_{k=1}^{K}$ satisfying the K-means objective function O.

1: Cluster the set of data points $Y = \{LUV_{jm}\}$, $j = 1..s$, $m = 1..f_j$, where f_j is the number of SIFT features in image I_j to g groups using standard K-Means algorithm. Assign each SIFT point to a $ColorCategory_{jm}$, $j = 1..s$, $m = 1..f_j$ depending on its corresponding cluster. We use group and color category interchangeably within this algorithm.

2: Label each cluster $C_k, k = 1..K$ with one of the g groups so that each group has approximately equal number of clusters, where K is the total number of clusters.

3: Assign each $Feature_{jm}$ randomly to one of the clusters labeled with $G_{ijm} = \{ColorCategory_{jm}\}$.

4: **repeat**

5: Re-compute cluster means

$$\mu_k \leftarrow \frac{1}{|C_k|} \sum_{x \in C_k} x \tag{3.24}$$

6: Assign each $Feature_{jm}$ to the nearest cluster labeled with one of the groups labeled with $G_{ijm} = \{ColorCategory_{jm}\}$ as below: Using Euclidean Distance function d, compute $d(Feature_{jm}, \mu_k)$ for $k = 1..K$. Assign $Feature_{jm}$ to k^* where

$$d(Feature_{jm}, \mu_{k^*}) <= d(Feature_{jm}, \mu_k), k = 1..K.$$

7: **until** no feature to cluster assignment changes

3.6 Parallelization of the Clustering Algorithm

A close look at the K-means algorithm indicates that the computational complexity is in the order of $O(sKdL)$, where s is the number of data points, K is the number of clusters, d is the

dimension of the feature vector and L is the number of iterations. This requires impractically long time, when implemented on a high end single processor machine.

To obtain speedup in computation time and divide the memory requirement to multiple processors, we use parallelism in standard K-means and semi-supervised K-Means algorithms. Our approach is based on [60] that is outlined in Algorithm 10.

The most common approach for implementing parallelism is message passing which is based on communicating through sending of messages to recipients. We use Message Passing Interface (MPI), which is a standardized and widely used library for message passing [61, 62]. MPI commands employed in this study are outlined in Table 3.3.

Table 3.3: MPI Commands Used in Parallel Clustering.

MPI Command	**Decription**
MPI_Comm_size()	return the number of processes
MPI_Comm_rank()	return the process identifier
MPI_Bcast(A, root)	broadcast the value of variable ”A” from the process with ”root” identifier to all of the processes
MPI_Allreduce(A, B, MPI_SUM)	sum the local values of variable ”A” of all the processes and distribute the result back to the processes in variable ”B”

3.7 Computational Complexity of SSA Algorithm

We first analyze the computational complexity of the sequential version of the K-means algorithm. The computational complexity depends on the number of visual features to be clustered s, the dimension of the visual feature d, and the number of iterations, L, while converging. Each addition, multiplication or comparison operation is considered as one floating point operation (flop). At each iteration of a loop, Euclidean distance from each point to each cluster center and cluster means are computed. Euclidean distance calculations take $3sKd + sK + sd$ flops [60]. Cluster center calculations take Kd flops. Assuming that the time for each flop to be t_{flops}, the overall computational complexity of the sequential K-means algorithm is:

$$C_{sequential\_kmeans} = [3sKd + sK + sd + Kd]\, Lt_{flop} \ . \tag{3.25}$$

Algorithm 10 Outline of the Parallel Semi-supervised K-means Algorithm.

Require: A set of data points $X = \{x_j\}_{j=1}^{n}$.

Ensure: Disjoint k partitions $\{C_k\}_{k=1}^{K}$ satisfying the K-means objective function O.

1: $p = MPI\_Comm\_size()$ // number of processes
2: $r = MPI\_Comm\_rank()$ // process identifier
3: $MSE = 0$ // mean squared error
4: $OldMSE = \infty$ // old mean squared error
5: **if** $(r = 0)$ **then**
6: Initialize cluster centroids $\{\mu_i\}_{k=1}^{K}$ at random
7: **end if**
8: $MPI\_Bcast(\{\mu_k\}_{k=1}^{K}, 0)$
9: Initialize semi-supervision constraints
10: **while** $MSE < OldMSE$ **do**
11: $OldMSE \leftarrow MSE$
12: $MSE' \leftarrow 0$ // mean squared error within ith cluster
13: **for** $i = 1$ to k **do**
14: $n_i' \leftarrow 0$ // the number of data points within ith cluster for process r
15: $\mu_i' \leftarrow 0$ // mean of ith cluster for process r
16: **end for**
17: **for** $j = r * (n/p) + 1$ to $(r + 1) * (n/p)$ **do**
18: Assign data point x_j to the cluster i^* where $i^* = \underset{i}{\operatorname{argmax}} \|x_j - \mu_i\|^2$. Only consider clusters that satisfy semi-supervision constraints
19: $n_{i^*}' \leftarrow n_{i^*}' + 1; \mu_{i^*}' \leftarrow \mu_{i^*}' + x_j$
20: $MSE' \leftarrow MSE' + \|x_j - \mu_{i^*}\|^2$
21: **for** $i = 1$ to k **do**
22: $MPI\_AllReduce(n_i', n_i, MPI\_SUM)$
23: $MPI\_AllReduce(\mu_i', \mu_i, MPI\_SUM)$
24: $n_i \leftarrow max(n_i, 1); \mu_i \leftarrow \mu_i / n_i$
25: **end for**
26: $MPI\_AllReduce(MSE', MSE, MPI\_SUM)$
27: **end for**
28: **end while**

Parallel version of K-means algorithm reduces distance calculation time by distributing the s number of visual features among P processors. For the sake of simplicity, assume that s is divided by P without any remainder. If all the K-cluster centers are available to each processor, we can divide s features among P processors and compute distance to center calculations in a parallel fashion. Hence, the number of flops for Euclidean distance calculations are reduced by P. Thus, total time for distance calculations is $(3sKd + sK)/P + sdt_{flops}$.

Parallel computation requires that at each iteration, number of points within a cluster and the sum within each cluster are distributed to other processes. Assume that each transfer operation takes t_{reduce} time, which is reported for most architectures to be $O(logP)$ [63]. Hence, the elapsed total time is:

$$C_{parallel\_kmeans} = \left[\frac{3sKd + sK + sd}{P}\right] Lt_{flop} + KdLt_{reduce} \ . \tag{3.26}$$

In semi-supervised sequential clustering, since each visual feature is compared with cluster centers belonging to one of g groups and each group has equal number of clusters, the distance calculations take $(3sKd + sK + sd)/g)Lt_{flop}$ time. Therefore, the overall computational complexity becomes:

$$C_{semi\_sequential\_kmeans} = \left[\frac{3sKd + sK + sd}{g} + Kd\right] Lt_{flop} \ . \tag{3.27}$$

In parallel version of semi-supervised clustering; the time spent in distance calculations is reduced by P, due to the same reasoning behind the parallel version of the standard K-means clustering. Then, the overall complexity becomes,

$$C_{semi\_parallel\_kmeans} = \left[\frac{3sKd + sK + sd}{gP}\right] Lt_{flop} + KdLt_{reduce} \ . \tag{3.28}$$

As indicated in [60], communication cost among processors becomes insignificant compared to the distance calculation if

$$\frac{Pt_{reduce}}{3t_{flop}} << s \ . \tag{3.29}$$

Since the left hand side of the equation is a machine constant, as s increases, distance calculation cost gradually dominates the communication cost.

Both sequential and parallel versions of semi-supervised clustering have less time complexities due to the reduction of factor g, in distance calculations.

3.8 Summary

In this chapter, we introduce a new image annotation system, called Semi Supervised Annotation (SSA).

The proposed SSA system, utilizes the unused available information to guide and restrict the clustering of low level visual features. For this purpose, the concept of ”side information” is introduced. Then, this general concept is instantiated by using the local and global properties of the images in the database. The side information is used to semi-supervise the clustering process. Clustering of low level visual features is performed in such a way that features with the same side information are constrained to fall in the same cluster groups. Semi-supervised clustering enables us to have clusters with sharper probability density functions which in turn reduce the overall entropy of the system. By reducing the randomness, we get better annotation performances as will be demonstrated in the next chapter. Moreover, during the clustering process, we compare the visual features with not all of the cluster centers, but with only those assigned to them based on their side information. Hence, we get better performance with respect to the computational complexity. To speed up both standard K-means and SSA algorithms, we introduce parallel versions and discuss the efficiency gained in the computational complexity.

CHAPTER 4

EXPERIMENTAL ANALYSIS OF SEMI SUPERVISED IMAGE ANNOTATION AND PERFORMANCE METRICS

In this chapter, we shall present an experimental analysis of the proposed Semi-Supervised Annotation System, SSA and compare it to the PLSA-Words, which is the state of the art image annotation algorithm proposed in [20].

First, we describe the data set used in the experiments. Next, we discuss the performance metrics in image annotation problem. These performance metrics are used to obtain the optimal parameters of the proposed semi-supervised image annotation technique using cross validation. Then, we show that in terms of mean average precision performance SSA performs better than PLSA-Words. Next, we analyze the performance per word. Then, we show that entropy of the SSA system is decreased when compared with PLSA-Words. We conclude the chapter by discussing the weaknesses and superiorities of the proposed technique.

4.1 Data Set

We use the same data set as in [20], which is a subset of the Corel data. It contains mostly outdoor scene photographs taken by professional photographers.

Sample images from the data set are given in Figure 4.1. Note that the number of annotation words for each image changes between 1 and 5. Note also that the words ”sky” and ”water” appear more frequently than for example ”clouds”. Data set consists of ten subsets each of which is divided into training and testing sets. Training set and test set constitute 75% and 25% of the subset, respectively. The number of images and the number of text words used

Figure 4.1: Sample images and their annotations from the Corel data set.

Table 4.1: The average and standard deviation of the number of images in training and test subsets, and the number of words used in each subset.

	Training	**Test**	**Number of Words**
Mean	5244	1750	161
Standard Deviation	39	26	9

Table 4.2: Twenty words (ranked in decreasing order) that occur most frequently in each subset for subsets 1-5.

Subset # 1	**Subset # 2**	**Subset # 3**	**Subset # 4**	**Subset # 5**
water	water	water	water	water
sky	sky	tree	trees	sky
tree	tree	sky	sky	tree
people	people	people	people	people
buildings	grass	grass	grass	flowers
grass	building	rocks	rock	grass
clouds	rock	flowers	snow	buildings
rock	mountain	mountain	mountains	rock
birds	flowers	snow	building	mountains
mountain	close-up	fish	flower	snow
stone	clouds	buildings	bird	clouds
snow	snow	ocean	ocean	leaves
street	plane	clouds	stones	fish
plane	fish	birds	boat	plants
flowers	street	closeup	clouds	boats
pattern	jet	gardens	plants	close-up
jet	field	coral	coral	closeup
texture	cat	leaves	field	cat
fish	horses	plants	leaves	stone
coast	pattern	boats	fish	street

in annotations are different in each subset. The total number of annotation words is 437. For these 10 subsets, the average number of images, the standard deviation of the number of images, and the average number of words used in ten training and test subsets is provided in Table 4.1, to give an idea about the statistical properties of the data set.

Each image in the data set is annotated with 1-5 words. Word frequencies in the first subset are given in Figures 4.2. Word frequencies in all the 10 subsets can be found in Appendix A. The distribution of words in all subsets looks similar and highly skewed. Twenty words that occur most and least frequently in each subset are given in Tables 4.2 (Sets 1-5), 4.3 (Sets 6-10) and 4.4 (Sets 1-5), 4.4 (Sets 6-10), respectively.

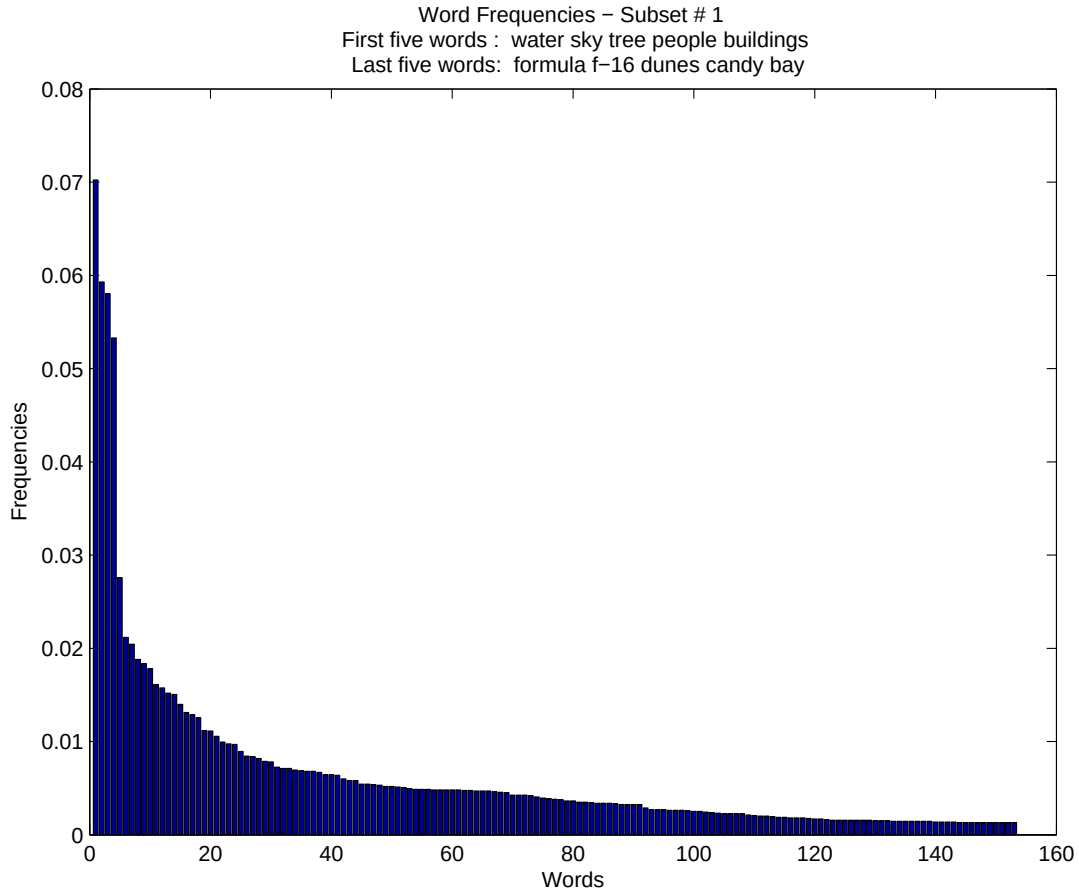

Figure 4.2: Word frequencies in subset 1. Words are sorted based on their frequencies in decreasing order from left to right. Most frequent and least frequent 5 words are listed at the top of the figure.

Table 4.3: Twenty words (ranked in decreasing order) that occur most frequently in each subset for subsets 6-10.

Subset # 6	Subset # 7	Subset # 8	Subset # 9	Subset # 10
water	water	water	water	water
sky	tree	sky	sky	sky
people	people	tree	people	tree
tree	sky	people	tree	people
buildings	grass	grass	grass	flowers
flowers	buildings	snow	snow	mountain
mountain	birds	buildings	building	snow
grass	rock	mountain	mountains	rocks
snow	close-up	flowers	flowers	buildings
clouds	snow	rocks	rocks	grass
rock	mountains	clouds	clouds	clouds
plants	cat	birds	fish	plants
birds	clouds	street	bird	leaves
leaves	street	close-up	boat	plane
fish	fish	field	closeup	fish
boats	stone	boats	ground	ocean
ocean	flowers	fish	sand	jet
horses	beach	patterns	plants	coast
plane	boats	plants	bear	boats
field	vegetables	texture	street	stones

Table 4.4: Twenty words (ranked in decreasing order) that occur least frequently in each subset for subsets 1-5.

Subset # 1	Subset # 2	Subset # 3	Subset # 4	Subset # 5
ships	pillar	relief	roofs	saguaro
saguaro	goats	kitten	island	palm
roofs	f-18	furniture	harbor	hillside
perch	door	floor	farm	herd
courtyard	display	sign	dog	carvings
castle	bobcat	prototype	dock	bushes
seals	ship	peaks	cheetah	roofs
prototype	hotel	park	formation	dall
outside	grapes	kauai	entrance	butterfly
detail	fan	goats	dunes	bay
tables	designs	bay	crop	wood
shrine	costume	anemone	sponge	slope
paintings	vegetation	sponges	costumes	goats
light	smoke	slope	arches	frozen
kauai	restaurant	sail	ship	formation
formula	giraffe	island	kitten	flags
f-16	courtyard	formation	iguana	columns
dunes	caterpillar	flag	herd	architecture
candy	bottles	f-18	floor	plain
bay	bengal	bush	bottles	detail

Table 4.5: Twenty words (ranked in decreasing order) that occur least frequently in each subset for subsets 6-10.

Subset # 6	Subset # 7	Subset # 8	Subset # 9	Subset # 10
skyline	tail	reefs	saguaro	town
petals	pyramid	hotel	rabbit	saguaro
butterfly	palm	horns	plain	waterfall
village	jaguar	grapes	harbor	tracks
formula	grapes	flight	courtyard	shrine
fan	cliff	face	carvings	peaks
doorway	art	bushes	rapids	outside
caribou	paintings	branches	hillside	antlers
zebra	moss	beetle	formula	restaurant
roofs	mosque	vineyard	bengal	candy
palm	lake	slope	tail	kauai
herd	bay	runway	sun	entrance
entrance	village	night	shrine	design
white-tailed	stairs	lake	shadow	castle
turn	shrine	kitten	race	hotel
waterfall	ships	hut	museum	furniture
castle	rabbit	herd	man	f-18
bull	kauai	grizzly	herd	f-16
bear	dog	costume	f-18	columns
baby	beetle	bottles	castle	bay

4.2 Performance Measurement

To be able to compare our method with the systems in the literature, we use the same metrics that have been used previously, namely, precision, recall and Mean Average Precision (MAP) values. For every annotation word, precision and recall is computed. Precision is the number of correctly annotated images divided by the total number of images annotated by that word. Recall is the number of images correctly annotated by a given word divided by the total number of images that have that word in the training set . Precision and recall values are averaged over all words. More precisely, letting Q_i be a query image, T_i is its true annotation, A_i is the estimated annotation, precision and recall values for a given word w is computed as follows:

$$precision(w) = \frac{size(\{Q_i | w \in T_i \ and \ w \in A_i\})}{size(\{Q_i | w \in A_i\})} , \quad i = 1..NQ , \tag{4.1}$$

$$recall(w) = \frac{size(\{Q_i | w \in T_i \text{ and } w \in A_i\})}{size(\{Q_i | w \in T_i\})} , \quad i = 1..NQ , \tag{4.2}$$

where NQ is the number of images in the test set and $\{Q_i | Condition_i\}$ corresponds to the subset of Q such that $Condition_i$ is satisfied for each of its elements.

Mean Average Precision (MAP) is calculated as in [20] by computing the mean of average precisions over all words. To compute average precision (AP) of a word; first, all images are ordered based on their probabilities in the model; next, for each rank for which corresponding image is relevant to the word, a precision showing the percentage of images that are correctly guessed up until to that rank is computed and then, these precisions are averaged over all such ranks. More precisely, if relevant images for a word w is denoted by $rel(w) = \{Q_i | w \in T_i\}$, total number of words in dictionary is L, image corresponding to *rank* is denoted by $I(w, rank)$, percentage of images that are correctly guessed up until to a given *rank* is denoted by $RankPrecision(w, rank)$:

$$AP(w) = \frac{\sum\limits_{I(w,rank) \in rel(w)} RankPrecision(w, rank)}{|rel(w)|} \tag{4.3}$$

and

$$MAP = \frac{\sum\limits_{j}^{L} AP(j)}{L} . \tag{4.4}$$

In the next sub-section, a new criterion, called Comparison of Average Precision Curve (CAP Curve) is introduced. Based on CAP, three metrics are defined to compare per-word average precision performances of annotation algorithms.

4.3 Comparison of Average Precisions

In this section, we define a new function,so-called CAP curve, for comparing the performances of image annotation algorithms. Two annotation algorithms may differ in such a way that, some words are estimated better by one of the algorithms and vice versa. Simply comparing the popular annotation metrics, MAP, precision or recall values for two algorithms do

not give any indication about the percentage of words that are better estimated by any of the two algorithms. We suggest computing the total average performance of words that are estimated better/worse than any other algorithm. Moreover, comparing MAP, precision or recall values, does not give any idea about the distribution of relative per-word performances of two different annotation algorithms. To be able to see this distribution visually, we sort per-word average precision difference values and plot these difference values sorted from highest to lowest. CAP Curve of annotation algorithm A_1 with respect to annotation algorithm A_2 is defined by subtracting the per-words average precision values of A_2 from those of A_1.

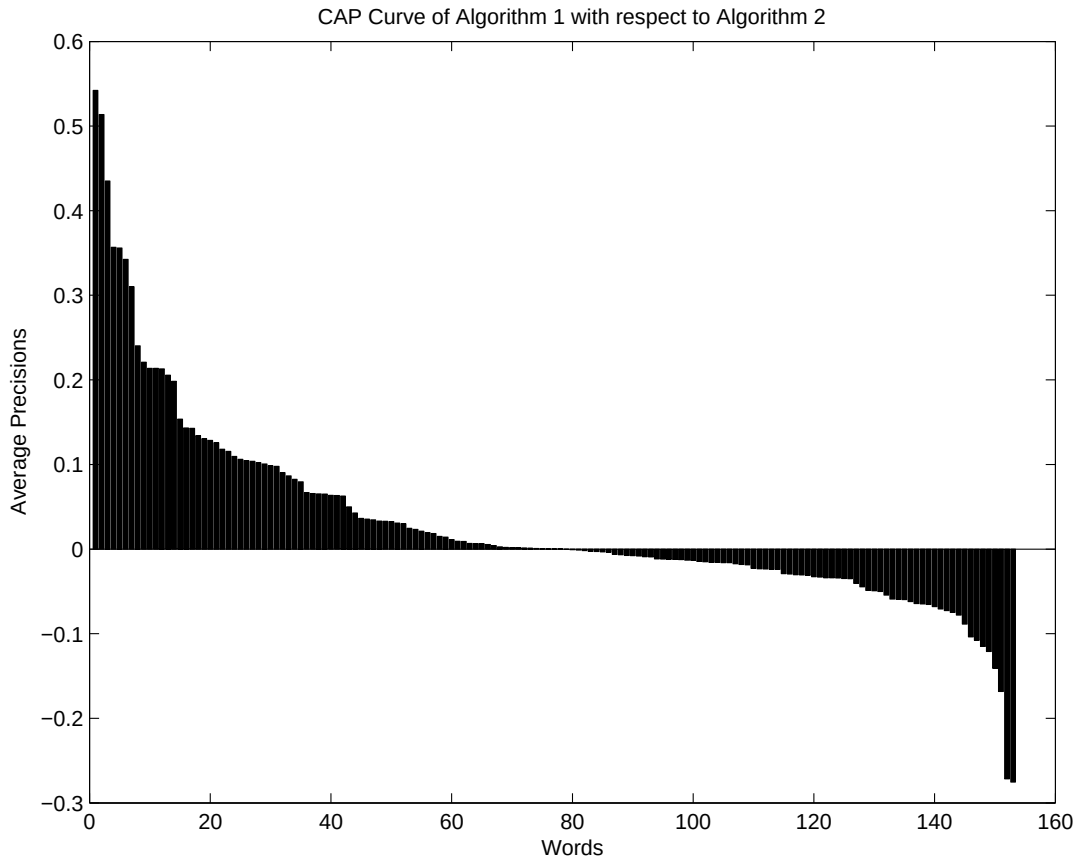

Figure 4.3: A sample CAP Curve that shows performance of Algorithm 1 with respect to Algorithm 2. CAP-percent-better shows the percentage of words where Algorithm 1 performs better. CAP-total-better and CAP-total-worse, correspond to areas above and below axis, respectively. Higher CAP-total-better and lower CAP-total-worse indicate the superiority of Algorithm 1 compared to Algorithm 2. CAP-percent-better:78/153, CAP-total-better:7.73, CAP-total-worse:3.29.

We define three new metrics based on CAP Curve, namely, CAP-percent-better, CAP-total-better and CAP-total-worse.

Firstly, CAP-percent-better is defined as the number of words that are better estimated by A_1 compared to A_2 divided by the total number of words. Secondly, CAP-total-better is the total of average precision differences for words that are better estimated by A_1. Thirdly, CAP-total-worse is the total of average precision differences for words that are better estimated by A_2.

A sample CAP Curve is presented in Figure 4.3. Note that, the sum of values above and below the x-axis correspond to CAP-total-better and CAP-total-worse, respectively. In CAP-percent-better value, 78/153, 78 corresponds to the number of words before the curve goes below x-axis whereas 153 is the total number of words.

In evaluation of image annotation algorithms, a variety of other performance metrics has been used. Blei and Jordan, [15] uses annotation perplexity, Barnard et. al. [30] use three different scores, namely, Kullback-Leibler divergence between predictive and target word distributions, normalized score that penalizes incorrect keyword predictions and the coverage. There is not any consensus as to which metric ”best” measures the image annotation performance, which requires further research in this area.

4.4 Estimation of Hyper-parameters of SSA by Cross-validation

One of the important parameters used in PLSA-Words and SSA is the grid size, where the HS features are extracted. It is clear that the optimum grid size depends on the size of images in the database. The optimal value is determined by cross validation among window sizes ranging from 10 to 100 in increments of 10. Figures 4.4 and 4.5 show mean average performance for cluster sizes 500 and 1000, respectively. As it can be seen in both figures performance increases steadily as the window size decreases.

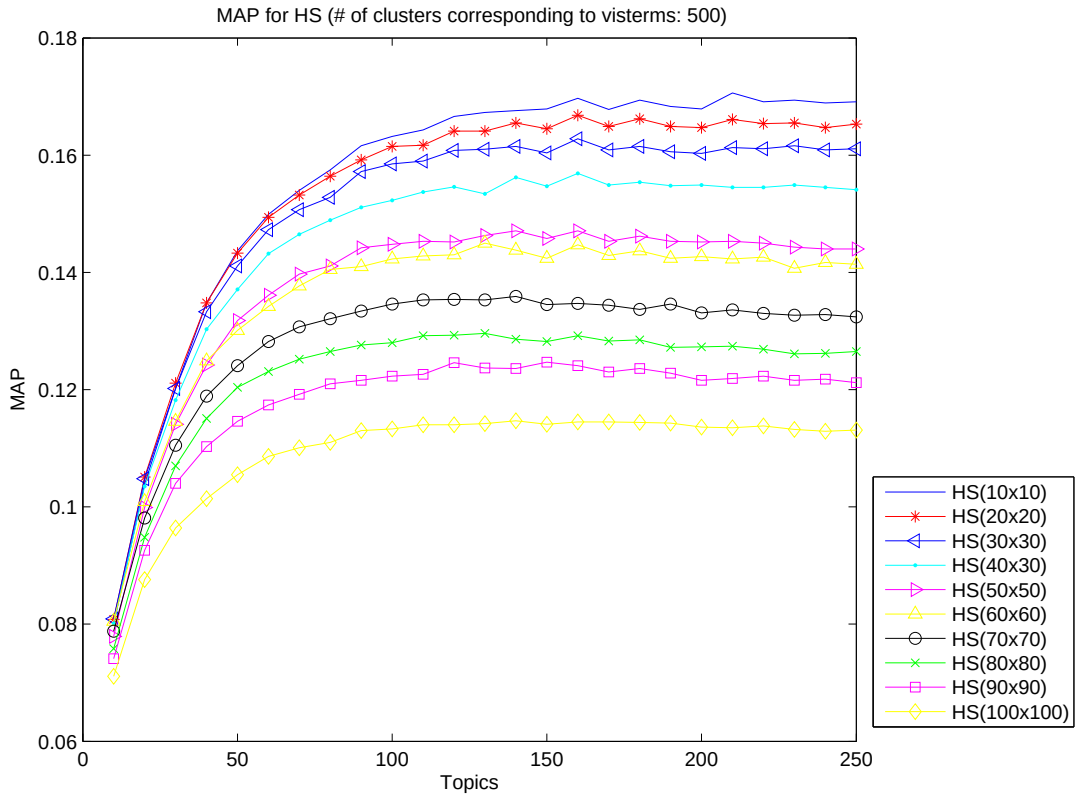

Figure 4.4: Cross Validation MAP results for HS for grid sizes ranging from 10x10 to 100x100 for 500 visterms. Grid window size is shown in parentheses. As the window size gets smaller, mean average precision values get higher consistently for all the number of hidden topics ranging from 10 to 250 in increments of 10.

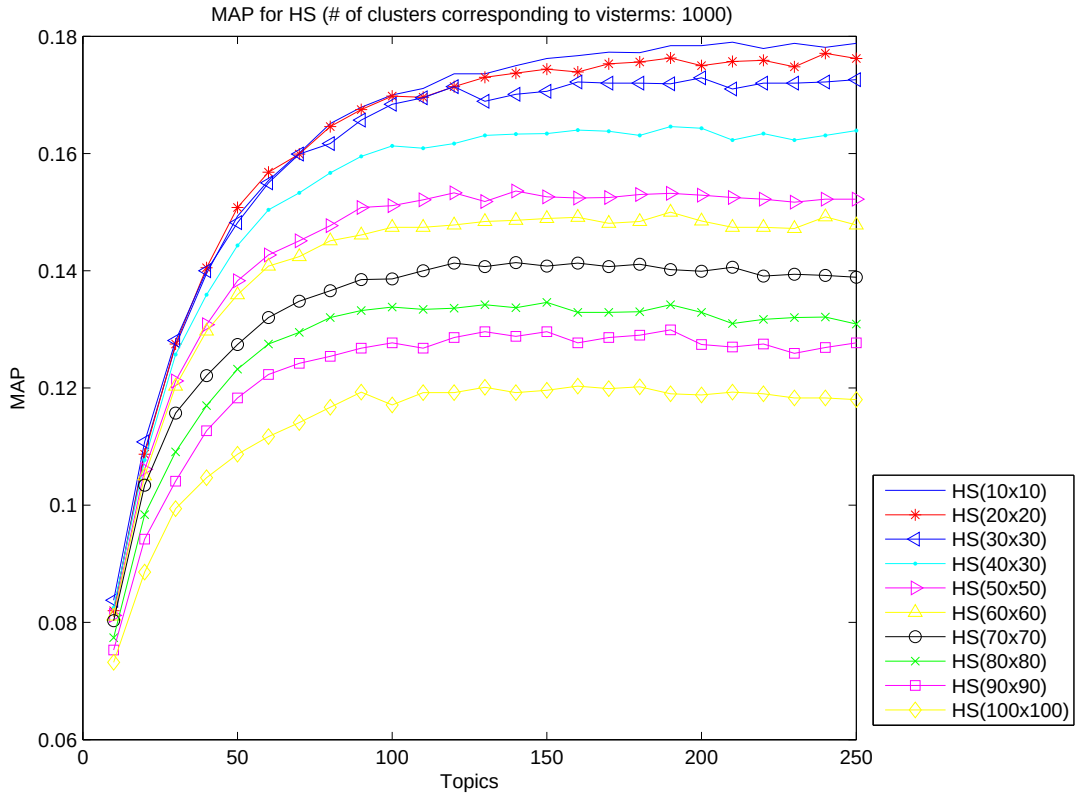

Figure 4.5: Cross Validation MAP results for HS for grid sizes ranging from 10x10 to 100x100 for 1000 visterms. Grid window size in parentheses. As the window size gets smaller, mean average precision values get higher consistently for all the number of hidden topics ranging from 10 to 250 in increments of 10.

MAP performances on the cross validation set for SSA-Topic and PLSA-Words Blob visterms are given in Figures 4.6, 4.7. As it can be seen from the figures, for both 500 and 1000 visterms, SSA gives better performances compared to PLSA-Words, for number of hidden topics higher than 40 and 60, respectively. For smaller number of hidden topics, SSA- Topic performs poorer, most probably because relatively coarse topics cannot provide enough information constraints to the clustering process, since topics that are too general are likely to correspond to every type of visual Blob feature. For 500 visterms, the MAP performances for PLSA-Words and SSA-Topic reach maximum of 0.14 and 0.16 at 120 and 130 hidden topics, respectively. For 1000 visterms, the MAP performances for PLSA-Words and SSA-Topic reach maximum of 0.14 and 0.17 at 140 and 150 hidden topics, respectively.

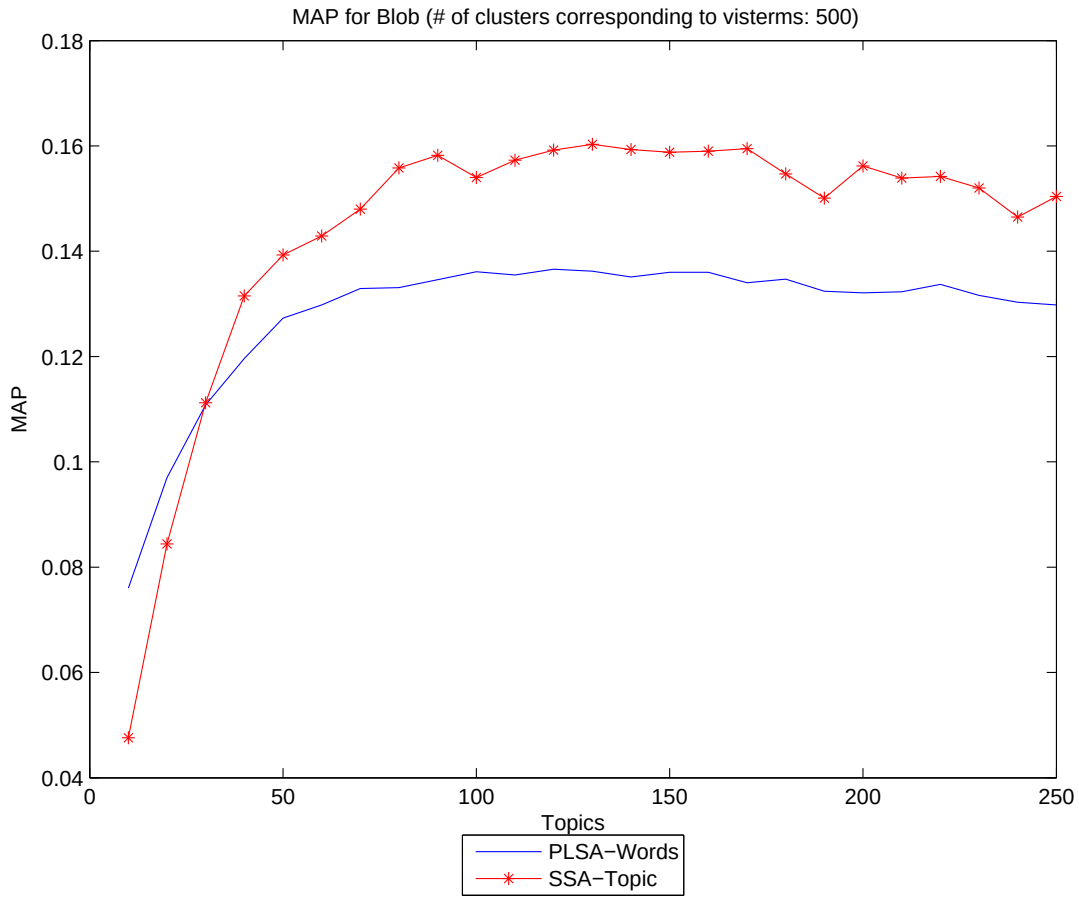

Figure 4.6: Cross Validation MAP results for PLSA-Words vs. SSA-Topic using 500 visterms. Mean average precision values for SSA-Topic is consistently better than PLSA-Words for number of hidden topic values higher than 30.

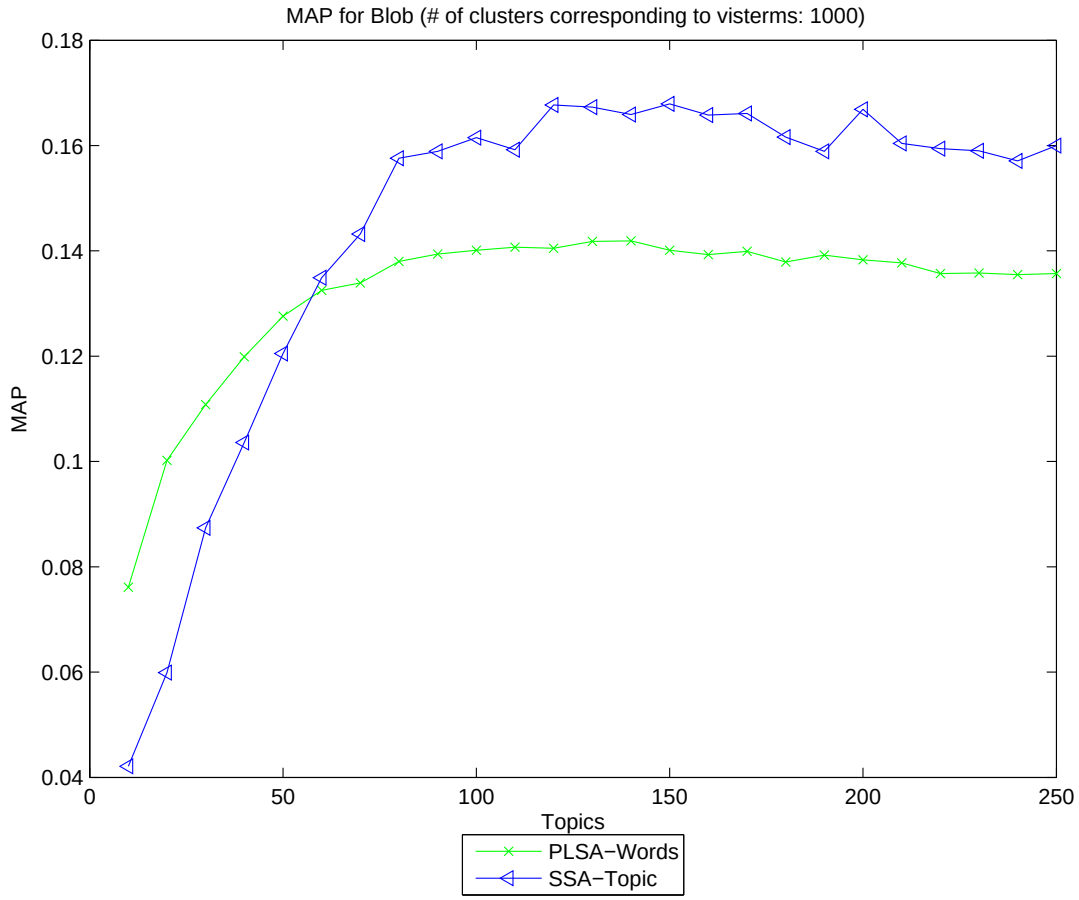

Figure 4.7: Cross Validation MAP results for PLSA-Words vs. SSA-Topic using 1000 visterms. Mean average precision values for SSA-Topic is consistently better than PLSA-Words for number of hidden topic values higher than 60.

MAP performances on the cross validation set for SSA-Orientation SIFT visterms based on group sizes 4 and 8 for 500 clusters are given in Figure 4.8. SSA-Orientation with group size 8, performs consistently better than the one with group size 4 for all hidden topic sizes. Comparison of MAP performances based on 500 visterms for SSA-Orientation and PLSA-Words is given in Figure 4.9. SSA-Orientation performs consistently better than PLSA-Words reaching its maximum of 0.14 at 230 hidden topics. PLSA-Words has the best result of 0.12 at 220 hidden topics.

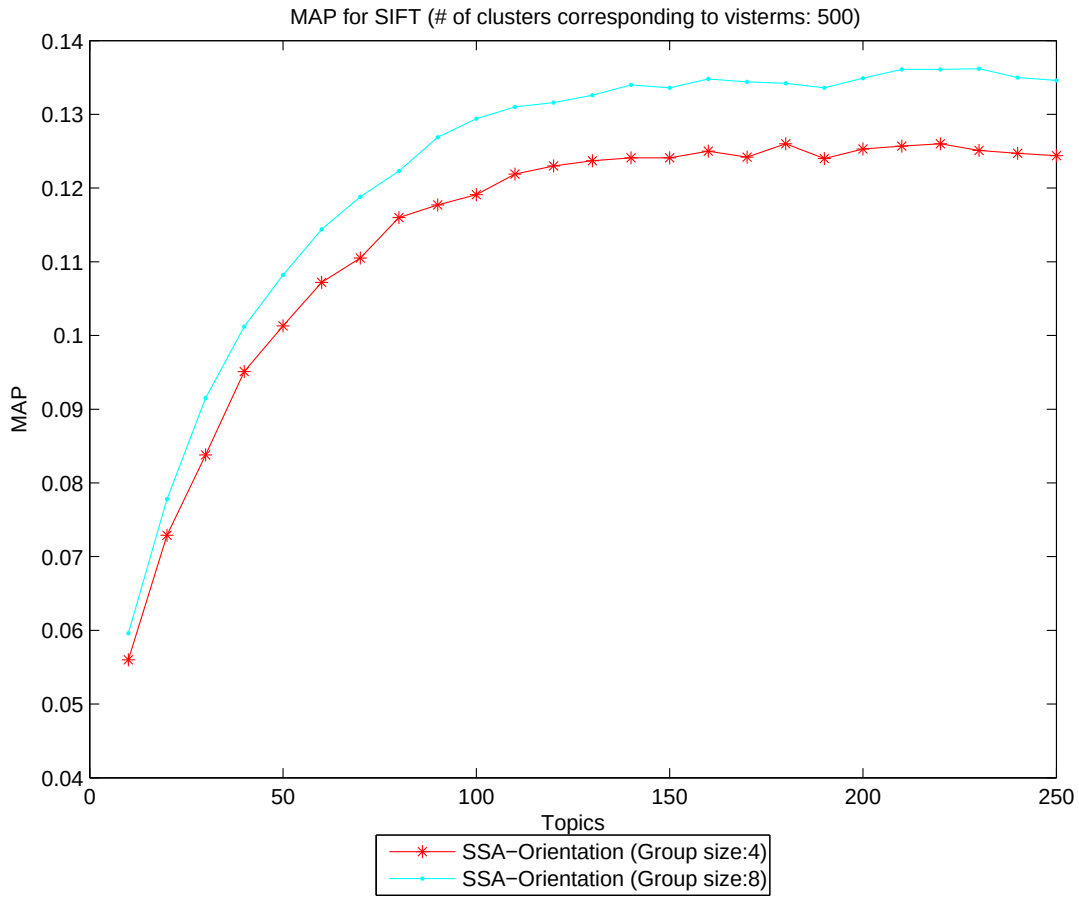

Figure 4.8: Cross Validation MAP results for SSA-Orientation using 500 visterms. Mean average precision values for SSA-Orientation with group size 8 is consistently better than SSA-Orientation with group size 4 for all the number of hidden topic values.

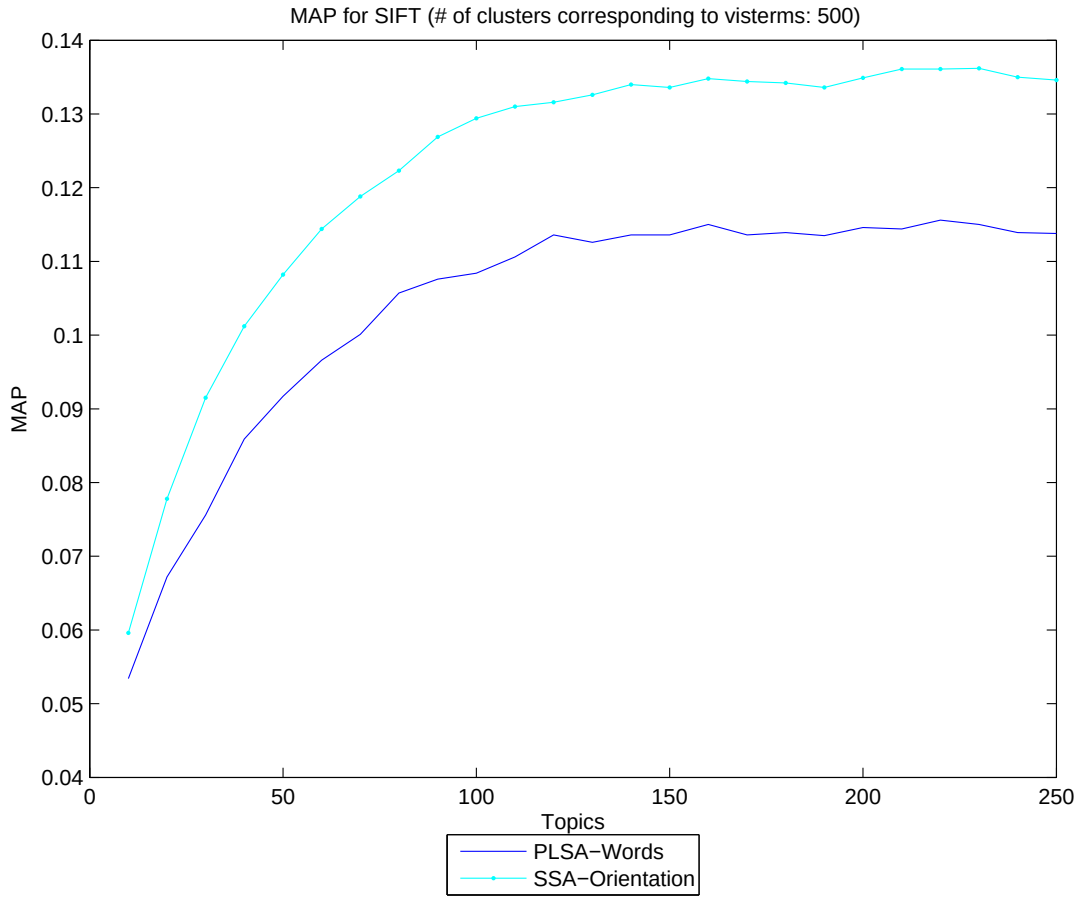

Figure 4.9: Cross Validation MAP results for PLSA-Words vs. SSA-Orientation using 500 visterms. Mean average precision values for SSA-Orientation is consistently better than PLSA-Words for all the number of hidden topics.

MAP performances on the cross validation set for SSA-Orientation SIFT visterms based on group sizes 4 and 8 for 1000 clusters are given in Figure 4.10 . As is the case for 500 clusters, SSA-Orientation with group size 8, performs consistently better than the one with group size 4 for all hidden topic sizes. Comparison of MAP performances based on 1000 visterms for SSA-Orientation and PLSA-Words is given in Figure 4.11. SSA-Orientation performs consistently better than PLSA-Words. SSA-Orientation and PLSA-Words reach their maximum MAP values of 0.14 and 0.12 at 240 and 210 hidden topics.

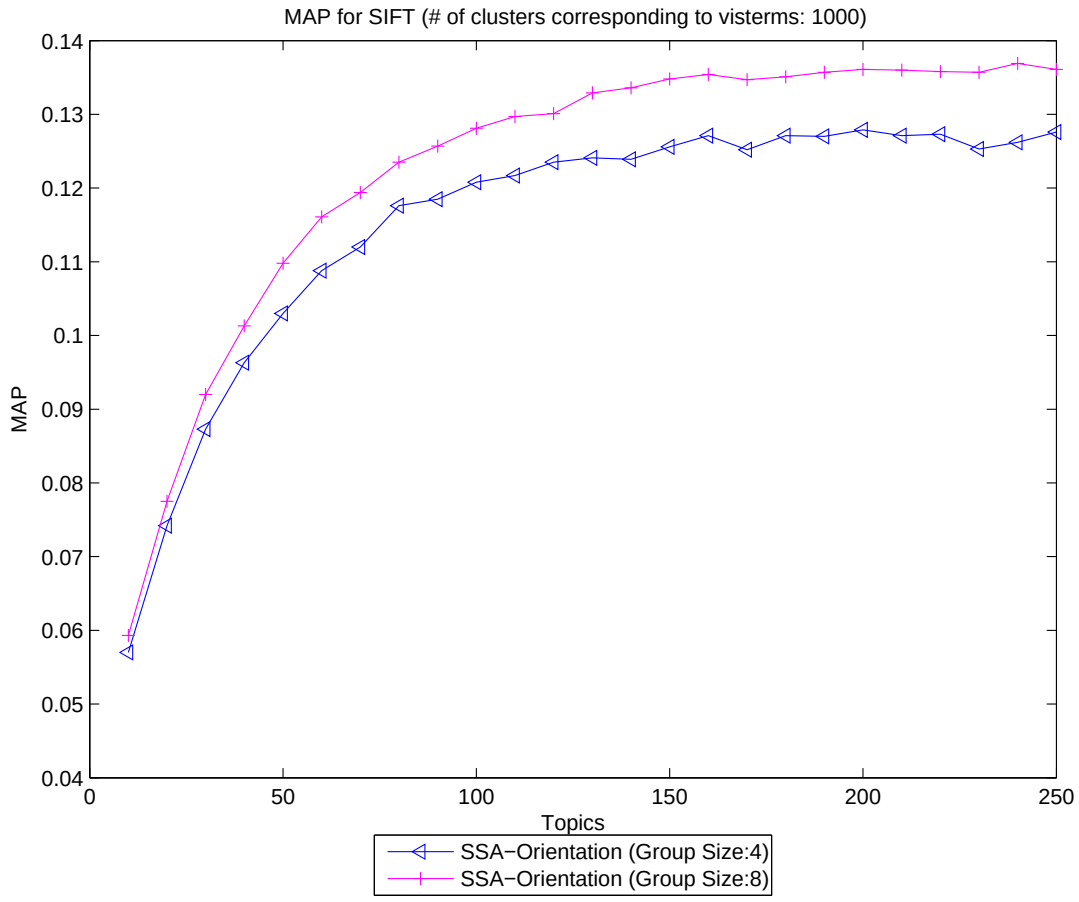

Figure 4.10: Cross Validation MAP results for SSA-Orientation using 1000 visterms. Mean average precision values for SSA-Orientation with group size 8 is consistently better than SSA-Orientation with group size 4 for all the number of hidden topics.

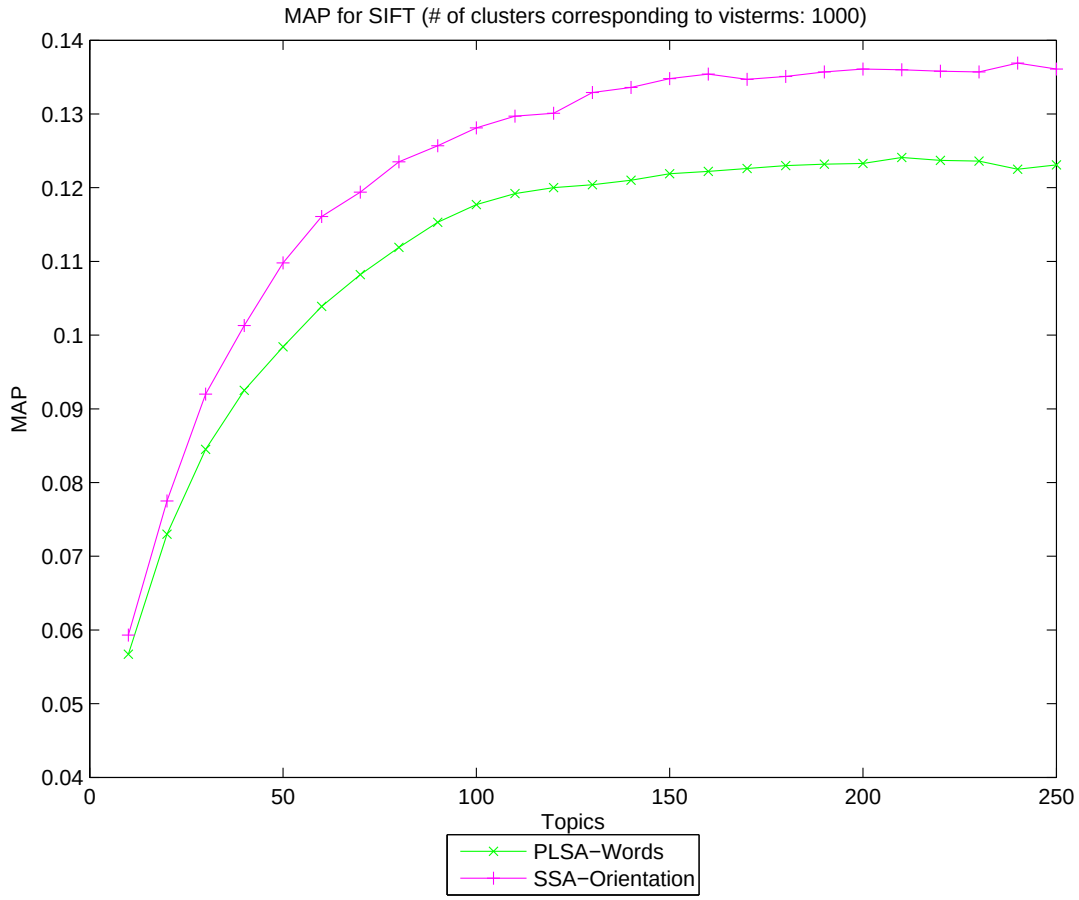

Figure 4.11: Cross Validation MAP results for PLSA-Words vs. SSA-Orientation using 1000 visterms. Mean average precision values for SSA-Orientation is consistently better than PLSA-Words for all the number of hidden topics.

MAP performances on the cross validation set for SSA-Color SIFT visterms based on group sizes of 8, 16, 32 and 64 for 500 clusters are given in Figure 4.12 . MAP values for SSA-Color with group sizes 32 and 64 are very close, and consistently better than those with group sizes 8 and 16 for all hidden topic sizes. Comparison of MAP performances based on 500 visterms for SSA-Color and PLSA-Words is given in Figure 4.13. For each hidden topic number, group size that gives the maximum MAP value is used. SSA-Color performs consistently better than PLSA-Words reaching its maximum of 0.17 at 240 hidden topics. PLSA-Words reaches its maximum MAP value of 0.12 at 220 hidden topics.

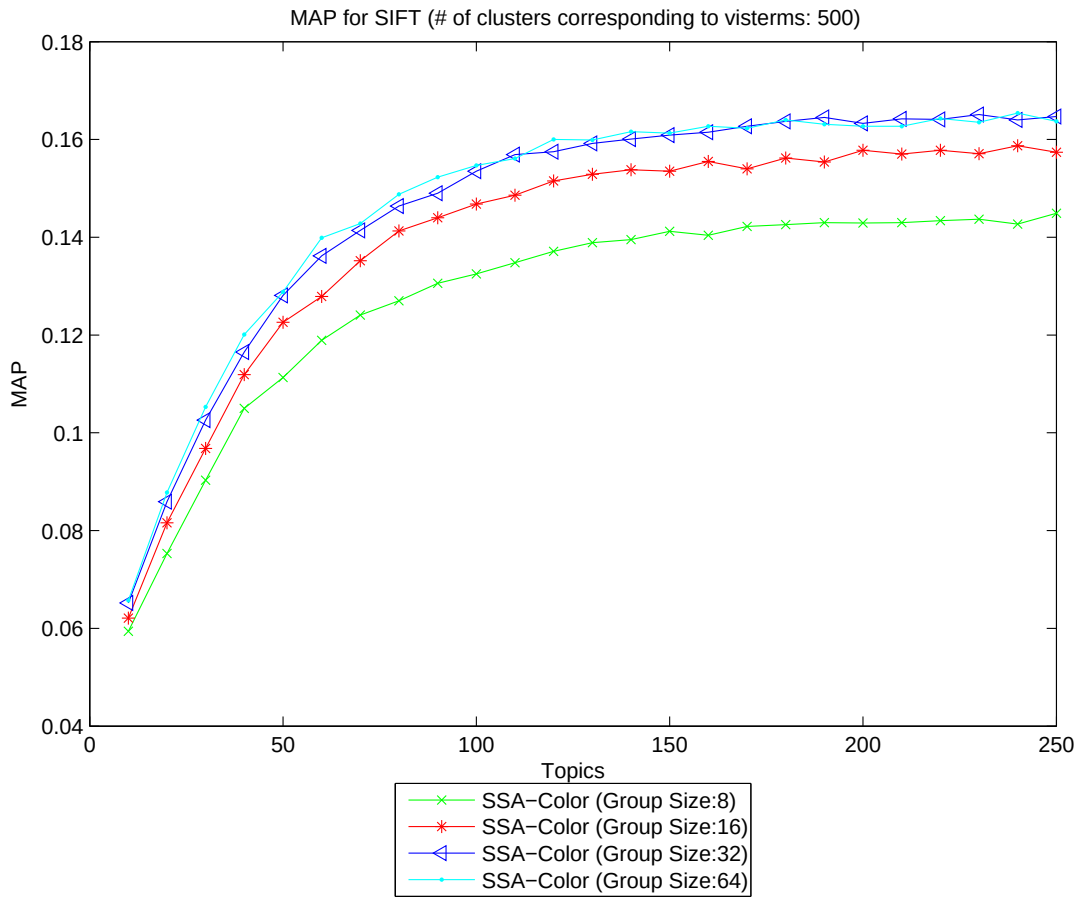

Figure 4.12: Cross Validation MAP results for SSA-Color using 500 visterms. Mean average precision values for SSA-Color gets higher as group size increases in general. Mean average precision values for group sizes 16 and 32 are close to each other. Depending on the number of topics, one or the other shows higher performance.

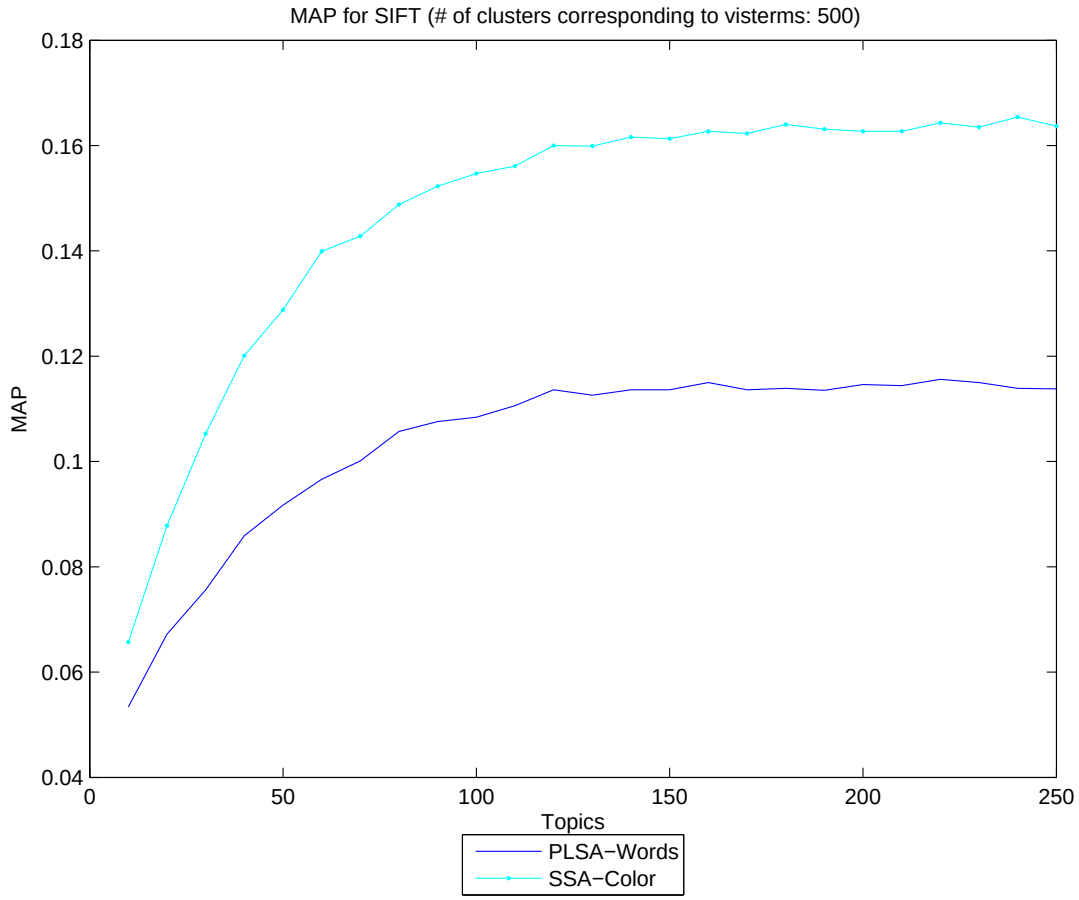

Figure 4.13: Cross Validation MAP results for PLSA-Words vs. SSA-Color using 500 visterms. Mean average precision values for SSA-Color is consistently better than PLSA-Words for all the number of hidden topics.

MAP performances on the cross validation set for SSA-Color SIFT visterms based on group sizes of 8, 16, 32 and 64 for 1000 clusters are given in Figure 4.14 . MAP values for SSA-Color with group sizes 64 are slightly better than the one with group size 32, and both are consistently better than those with group sizes 8 and 16 for all hidden topic sizes. Comparison of MAP performances based on 1000 visterms for SSA-Color and PLSA-Words is given in Figure 4.15. SSA-Color performs consistently better than PLSA-Words reaching its maximum of 0.17 at 220 hidden topics. PLSA-Words reaches its maximum MAP value of 0.12 at 210 hidden topics.

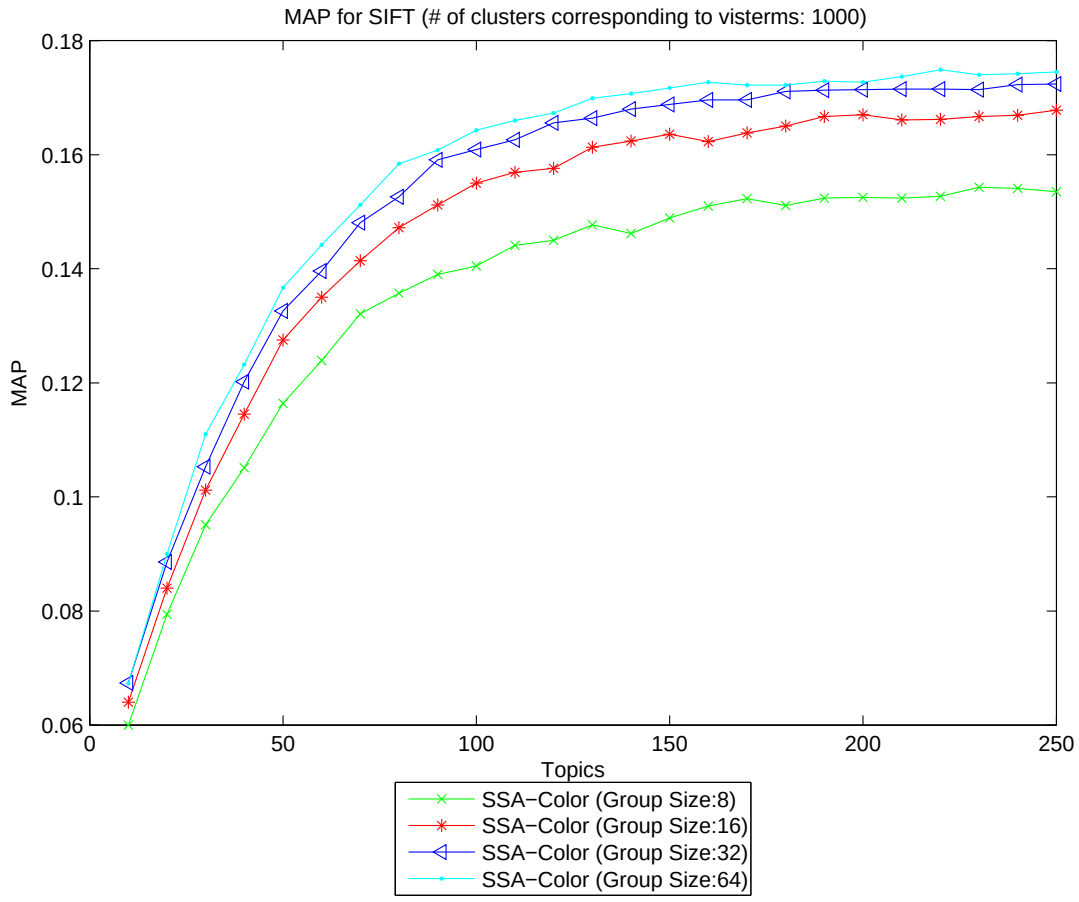

Figure 4.14: Cross Validation MAP results for SSA-Color using 1000 visterms. Mean average precision values for SSA-Color gets higher as group size increases for all the number of hidden topics.

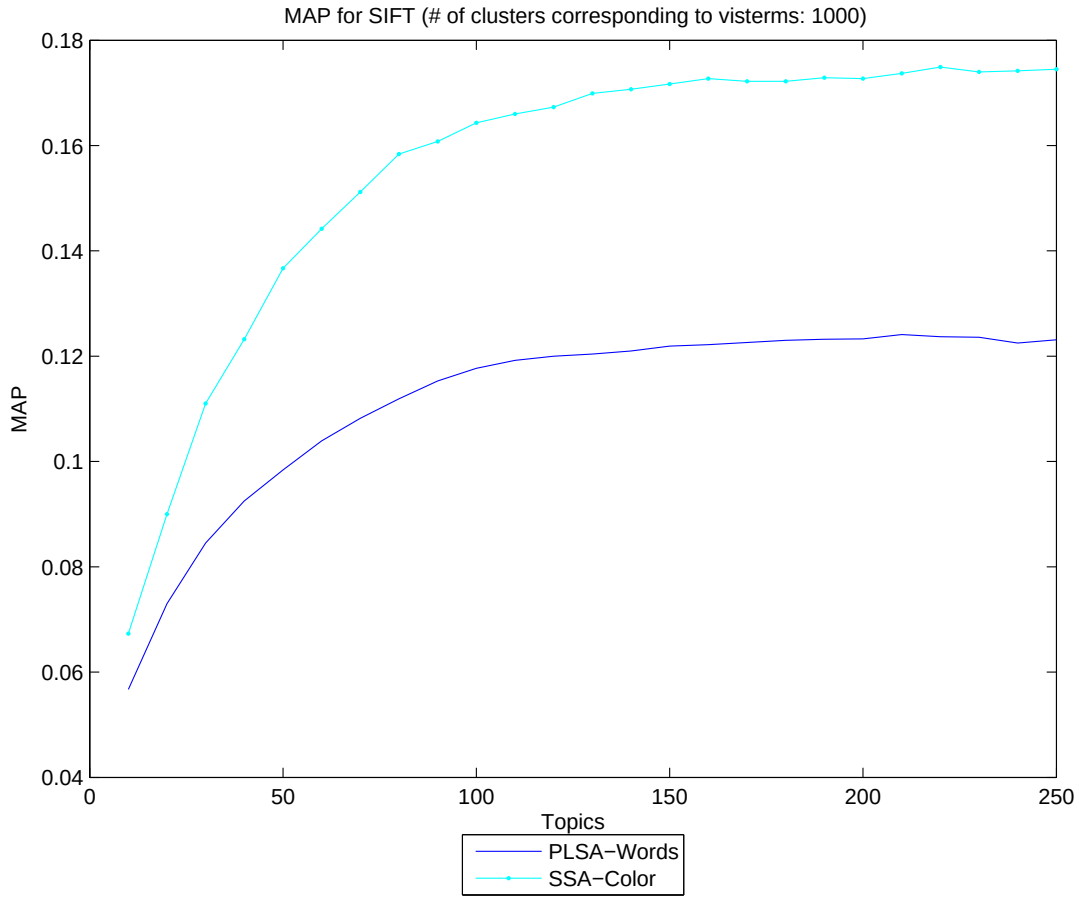

Figure 4.15: Cross Validation MAP results for PLSA-Words vs. SSA-Color using 1000 visterms. Mean average precision values for SSA-Color is consistently better than PLSA-Words for all the number of hidden topics.

We fixed total number of visterms to 2000 to be able to do a fair comparison with [20]. Using 500 or 1000 clusters for SSA-Color, SSA-Orientation, SSA-Topic and HS features, we made experiments on cross validation dataset for all combinations of cluster sizes such that the total number of clusters is 2000. Table 4.6 shows the results obtained for each such combination sorted from the lowest to highest MAP value top to bottom. We found out that the cluster sizes combination that gives the highest MAP score is the following: 500 SSA-Orientation features with a group size of 8, 500 SSA-Color features with a group size of 64 and 1000 HS features. SSA-Topic feature although better than plain Blob feature did not make into the best cluster combination. The highest MAP score has been obtained for 240 hidden topics, among the number of hidden topics ranging from 10 to 250 in increments of 10.

Table 4.6: Cross-validation Performance Results.

Rank	Type	Group Size	Cluster #	# Topics	MAP
1	SSA-Orientation SSA-Topic	4 140	1000 1000	140	0.1855
2	HS (10x10) SSA-Topic	 250	1000 1000	250	0.1862
3	SSA-Color SSA-Topic	64 210	1000 1000	210	0.188
4	SSA-Orientation HS (10x10) SSA-Topic	8 250	500 500 1000	250	0.2068
5	SSA-Color HS (10x10) SSA-Topic	64 240	500 500 1000	240	0.2074
6	SSA-Orientation HS (10x10) SSA-Topic	8 250	1000 500 500	250	0.2095
7	SSA-Color HS (10x10) SSA-Topic	64 200	1000 500 500	200	0.2101
8	SSA-Color HS (10x10) SSA-Topic	64 220	500 1000 500	220	0.212
9	SSA-Orientation HS (10x10)	8	1000 1000	240	0.2128
10	SSA-Orientation HS (10x10) SSA-Topic	8 250	500 1000 500	250	0.2132

Table 4.7: Cross-validation Performance Results (Continued).

Rank	Type	Group Size	Cluster #	# Topics	MAP
11	SSA-Color HS (10x10)	64	1000 1000	250	0.214
12	SSA-Color SSA-Orientation SSA-Topic	64 8 240	500 500 1000	240	0.2197
13	SSA-Color SSA-Orientation	64 8	1000 1000	250	0.2231
14	SSA-Color SSA-Orientation SSA-Topic	64 8 240	500 1000 500	240	0.2239
15	SSA-Color SSA-Orientation SSA-Topic	64 8 250	1000 500 500	250	0.2244
16	SSA-Color SSA-Orientation HS (10x10) SSA-Topic	64 8 220	500 500 500 500	220	0.2263
17	SSA-Color SSA-Orientation HS (10x10)	64 8	1000 500 500	240	0.2279
18	SSA-Color SSA-Orientation HS (10x10)	64 8	500 1000 500	240	0.2287
19	SSA-Color SSA-Orientation HS (10x10)	64 8	500 500 1000	240	0.2302

Table 4.8: Overall Performance Results.

	PLSA-WORDS HS(10x10)	**SSA HS(10x10)**	**PLSA-WORDS HS(30x30)**	**SSA HS(30x30)**
Mean per-word precision	0.15	0.17	0.17	0.18
Mean per-word recall	0.28	0.31	0.30	0.32
Mean average precision	0.18	0.21	0.20	0.21

Table 4.8 shows mean per-word precision, recall and MAP values for PLSA-Words and SSA. When we compare SSA with PLSA-Words; we see an increase in precision, recall, and mean average precision values both when the window size is taken as 10x10 obtained by cross-validation and the window size is taken as 30x30 as in [20] to be able to directly compare results of SSA with PLSA-Words.

4.5 Per-word Performance of SSA compared with PLSA-Words

In this section we will compare per-word performance of SSA with that of PLSA-Words.

In Figure 4.16, we show the CAP Curve of SSA with respect to PLSA-Words based on the first subset of the data set. The result is quite interesting: The values above the x-axis show the words, where SSA better performance, while the values below the x-axis shows the words where PLSA-Words has better performance. Note that the black area above the x-axis corresponding to CAP-total-better is larger than that of the area below the x-axis corresponding to CAP-total-worse, showing that the overall performance of SSA is higher compared to PLSA-Words. Moreover, 66 percent of the words are better estimated by SSA compared to PLSA-Words. This plot shows the importance of the design of side information. It is intuitive that the selection of the side information depends on the relationship between visual content of regions and the actual annotation words. It is highly difficult to find generic side information that is valid for all the words in the vocabulary. However, one may expect to extract this information to cover the wide range of words.

Relative MAP improvement for the best 20 words is shown in Figure 4.17 and the corresponding test images with highest average precision improvement for the first 8 words is given in Figure 4.18. Words that show the highest improvement corresponds to objects that have a known color and consistent orientation.

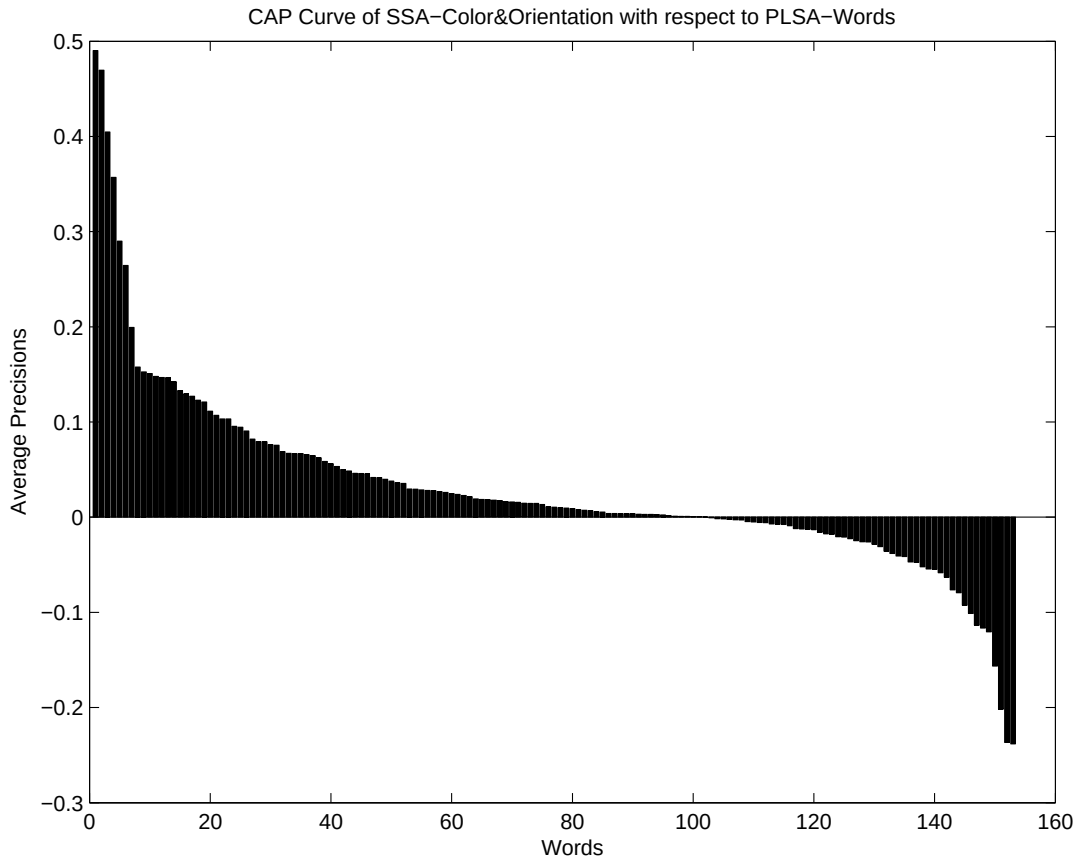

Figure 4.16: CAP Curve of SSA with respect to PLSA-Words. CAP-percent-better shows the percentage of words where SSA performs better. CAP-total-better and CAP-total-worse, correspond to areas above and below axis, respectively. Higher CAP-total-better and lower CAP-total-worse indicate the superiority of SSA compared to PLSA-Words. CAP-percent-better:102/153, CAP-total-better:6.96, CAP-total-worse:2.43.

The word ”Zebra” shows the highest performance gain because of its distinctive black and white color, and discriminative stripes texture having in different orientations. In regular SIFT feature, since interest point are normalized along the most dominant orientation, the fact that there is the same texture with different orientations on the same image is ignored. With the proposed SSA method, the orientation side information together with SIFT feature captures the stripe texture which occurs in different orientations for zebra.

The images annotated with ”Runway” have usually gray background with blue sky and gray colored planes. Since the planes are usually pictured while they are on the ground, the orientation side information corresponding to the body and the tail stay relatively same. Top and bottom of the body consists of horizontal edges , while the tail has mostly diagonal edges. Therefore, using color as side information captures the sky and gray background, while the orientation as side information captures the planes.

Pillars have usually brownish color with mostly vertically oriented textures. Pumpkins have a distinctive orange color, standing on the ground some of them having face pictures on them with consistent orientations. Hence, using color and orientation as side information in SSA, enables the system to identify both of these objects correctly.

Although ”black” does not correspond to a specific object, training images annotated with ”black” have bears and helicopters. Although the color or orientation information, when used separately, are not enough to discriminate these objects, the combination of them enables the system to increase the performance of identifying them.

Images annotated with ”tracks” usually have gray background with green grass on the side so that using color as side information enables the system to correctly recognize track objects. Cars displayed on the ground have relatively stable orientation values, top and bottoms of which having horizontal orientation values. Hence, the orientation side information correctly captures the car objects.

Perch has mostly a greenish color, and pictured while they are standing straight with consistent orientation values. Saguaro has green color and a stable distinctive orientation. Therefore, using color and orientation as side information enables the system to correctly recognize both of these objects.

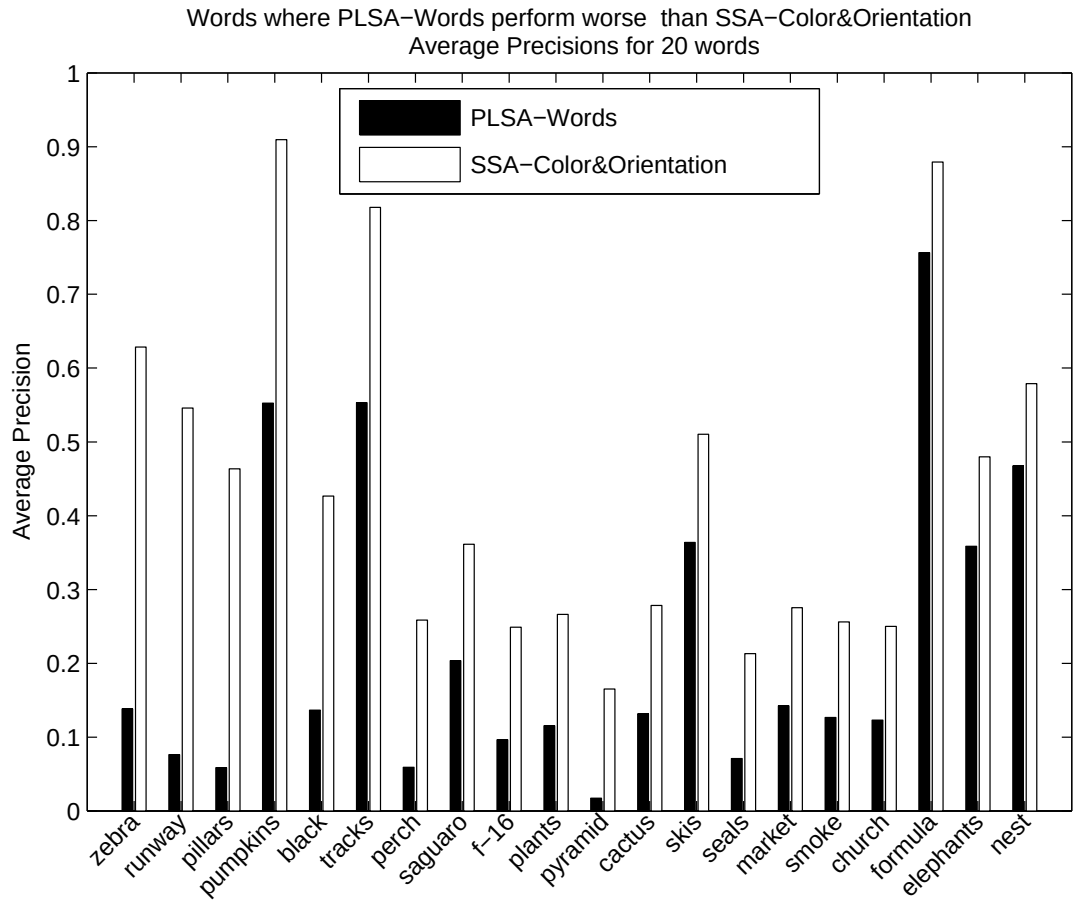

Figure 4.17: Relative average precision improvement for the best 20 words. Average precision difference is highest to lowest sorted from left to right.

Relative MAP reduction for the worst 20 words with respect to SSA is shown in Figure 4.20 and the corresponding test images with highest probability decrease for the first 8 words is given in Figure 4.21. As expected, the words that have the lowest performance correspond to objects that do not have any specific color or a consistent orientation. The word ”face” corresponds to human face, pumpkins prepared for Halloween painted as a face, or side of mountain in image annotations as it can be seen in Figure 4.19. Consequently, the word ”face” represents a variety of colors and textures having different orientations resulting in bad performance. The word ”texture” does not correspond to any specific object. Therefore, the annotations do not have any consistency in terms of neither color nor orientation of textures. The word ”branches” usually corresponds to gray, brownish or green color. Although the corresponding number of colors are not many, since there is not any consistency in the orientations, the performance is worse in the SSA compared to PLSA-Words. The same reasoning as why performance decrease occurs for the word ”texture” applies to the word ”pattern” as well. Images annotated by ”pattern” do not have any consistency in terms of neither color nor orientation of textures. The images annotated with the word ”lion” although has usually brownish color, there does not seem to be any texture in lion images showing a consistent orientation. ”Coral” images have many colors and they do not carry any specific orientation consistency. Images annotated with ”Birds” have many colors and they are pictured in a variety of orientations either standing or flying in different directions. Although images with ”Forest” annotation have either white or green color in common, there is not any specific orientation.

The above analysis indicates that the definition of the side information depends on several characteristics of both visual and textual words and their complex relationship.

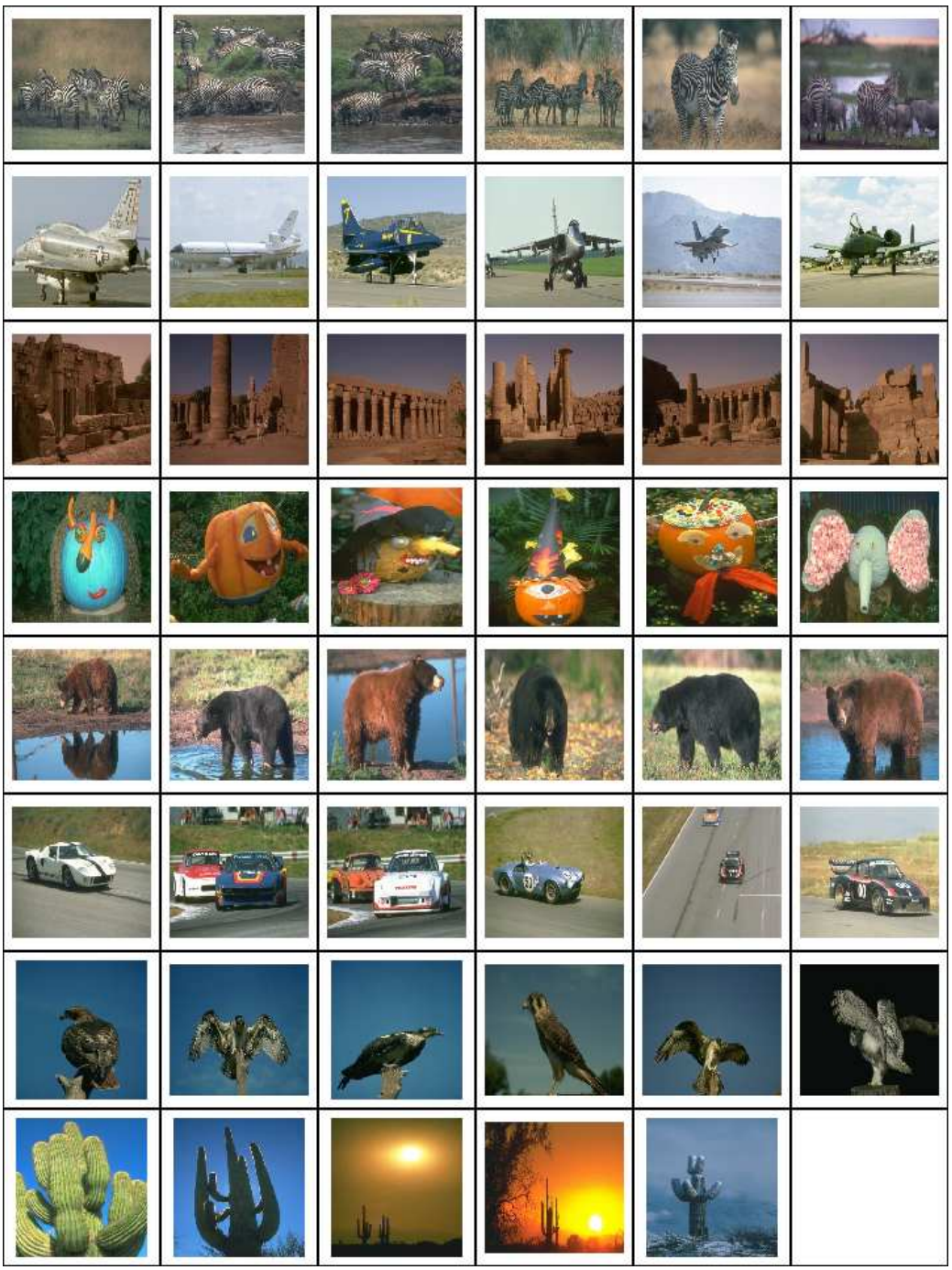

Figure 4.18: Test images with highest average precision improvement for the best 8 words. Model probability improvement of test images decrease left to right, top to bottom. Each row corresponds to a word. Words top to bottom: zebra, runway, pillars, pumpkins, black, tracks, perch, saguaro.

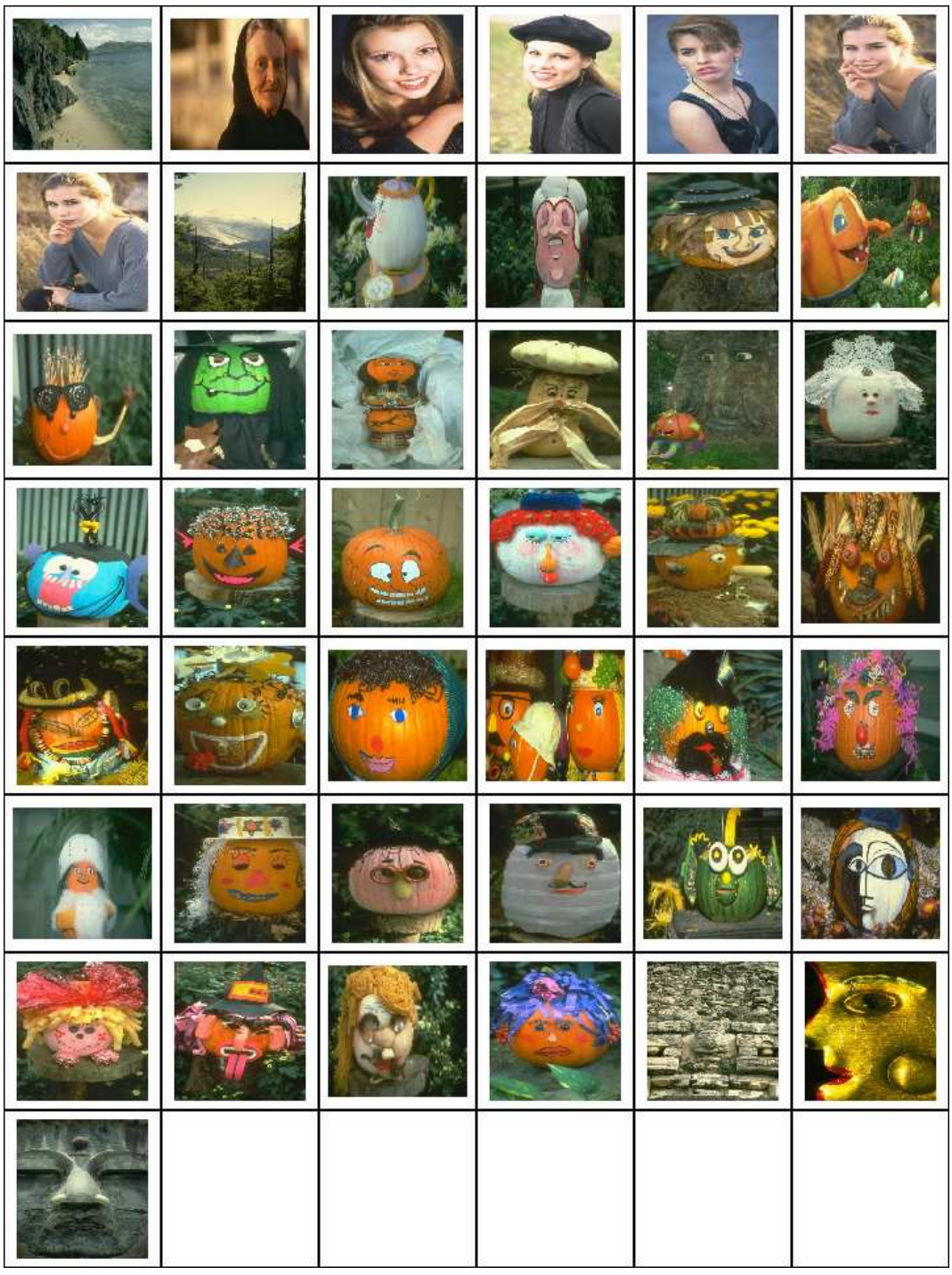

Figure 4.19: Training images for the word ”face”. ”Face” corresponds to different objects, namely, human face, pumpkins and side of a mountain.

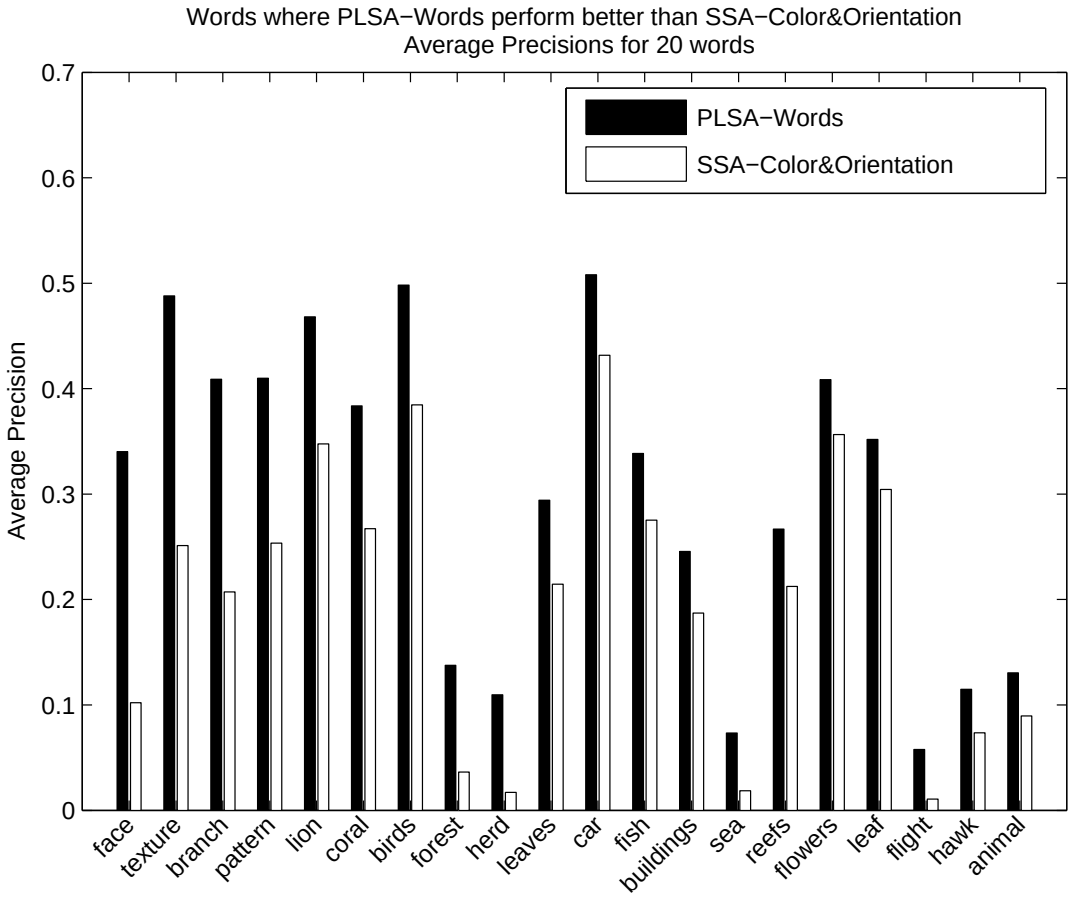

Figure 4.20: Relative average precision reduction for the worst 20 words. Average precision difference is highest to lowest sorted from left to right.

To see the effects of SSA-Orientation and SSA-Color features, the following subsections discuss experiments using SSA-Color only or SSA-Orientation only features using 500 clusters compared it with PLSA-Words method using only 500 regular SIFT clusters.

4.5.1 Per-word Performance of SSA-Orientation compared with PLSA-Words

In Figure 4.22, we present CAP Curve of SSA-Orientation with respect to PLSA-Words. The values above the x-axis show the words where SSA-Orientation performs better compared to PLSA-Words, while the values below the x-axis shows the words where PLSA-Words has better performance. Note that the black area above the x-axis is larger than that of the area below the x-axis showing that the overall performance of SSA-Orientation is higher compared to PLSA-Words. Although the percentage of words that are better estimated by SSA-Orientation is not high with a CAP-percent-better value of 0.55, CAP-total-better is approximately 72 percent higher than CAP-total-worse.

Relative MAP improvement for the best 20 words is shown in Figure 4.23 and the corresponding test images with highest average precision improvement for the first 8 words is given in Figure 4.25. Words that show the highest improvement corresponds to those objects that have a consistent orientation. Therefore, supervision of the clusters with the orientation as side information improves the relationship between the words and the visual features. For the images annotated with ”Runway”, since planes are usually are pictured while they are on the ground, the orientation values corresponding to the body and the tail stay relatively same, top and bottom of the body being horizontal with diagonal features on the tail. The images annotated with ”Sculpture” correspond to stable objects that have usually vertical orientations. Although images annotated with ”Birds” are pictured in a variety of orientations, either standing or flying in different directions, these variations seem to be captured by SSA. A close look into the training set shows that a relatively high percentage of images is annotated by the word ”Bird” as shown in Figure 4.24 suggesting that number of training images has a positive influence in SSA. The ”turn” word corresponds to scene where the side of the road has features orientations of which have regularly increasing orientations. This property seems to be captured by the SSA-Orientation feature resulting in better performance. Body, legs and trunks of elephants are usually pictured in the same orientations. Images annotated with ”Saguaro” has consistently the same orientation. Trunk of elephants show the similar

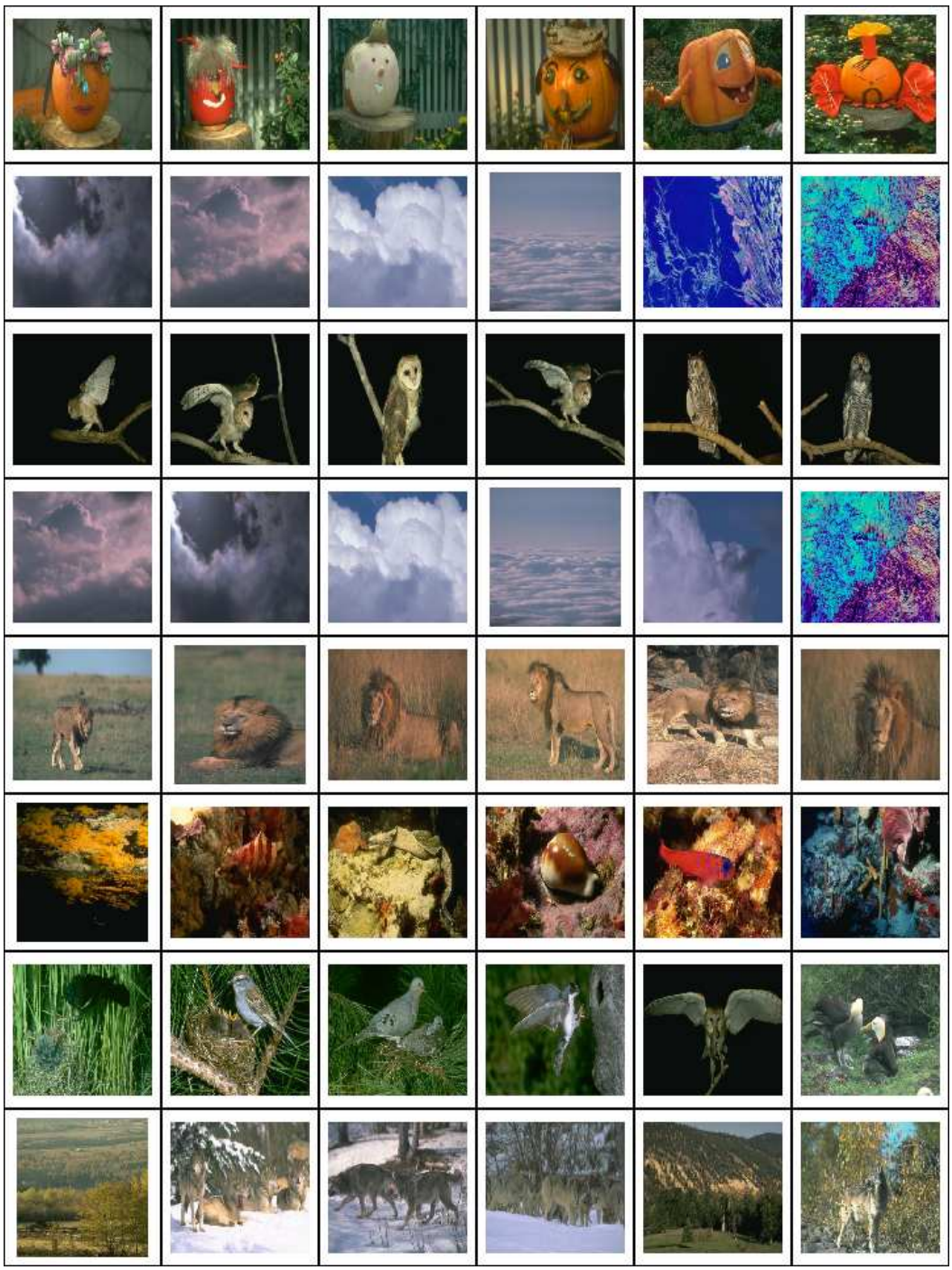

Figure 4.21: Test images with lowest average precision reduction for the worst 8 words. Model probability reduction of test images decrease left to right, top to bottom. Each row corresponds to a word. Words top to bottom: face, texture, branch, pattern, lion, coral, birds, forest.

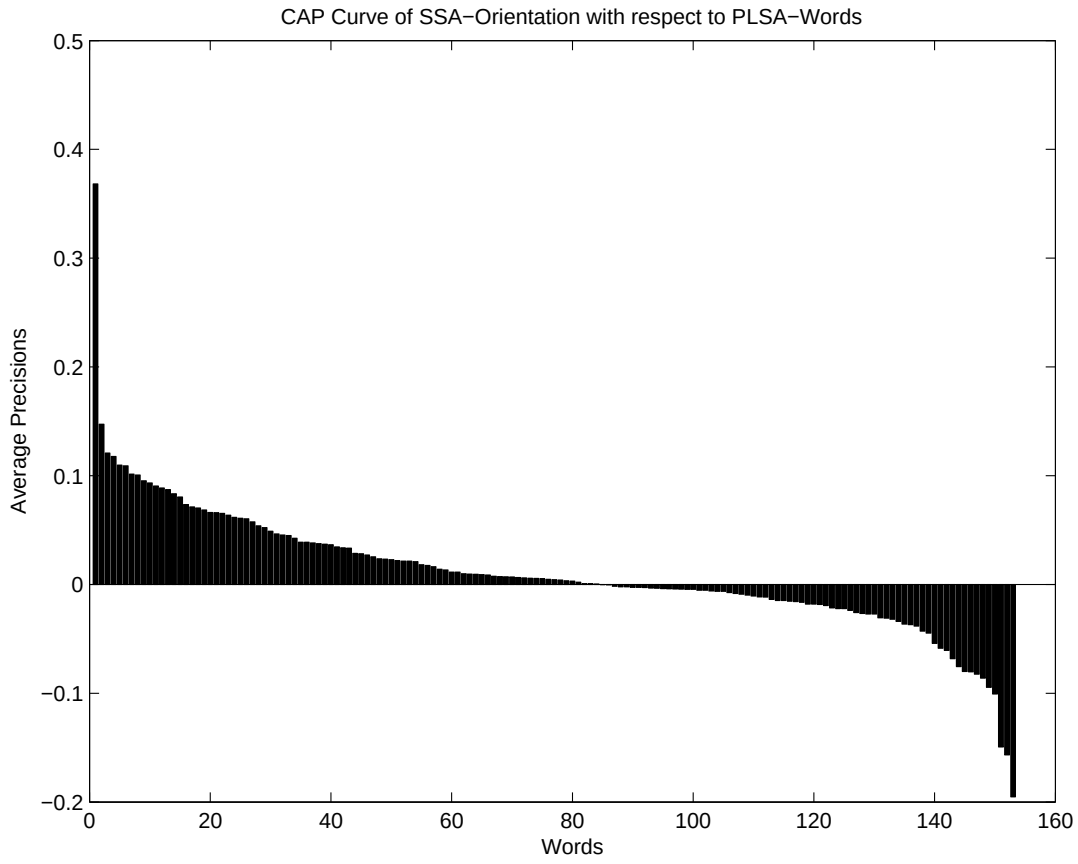

Figure 4.22: CAP Curve of SSA-Orientation with respect to PLSA-Words for 500 visterms. CAP-percent-better shows the percentage of words where SSA-Orientation performs better. CAP-total-better and CAP-total-worse, correspond to areas above and below axis, respectively. Higher CAP-total-better and lower CAP-total-worse indicate the superiority of SSA-Orientation compared to PLSA-Words. CAP-percent-better:84/153, CAP-total-better:3.74, CAP-total-worse:2.18.

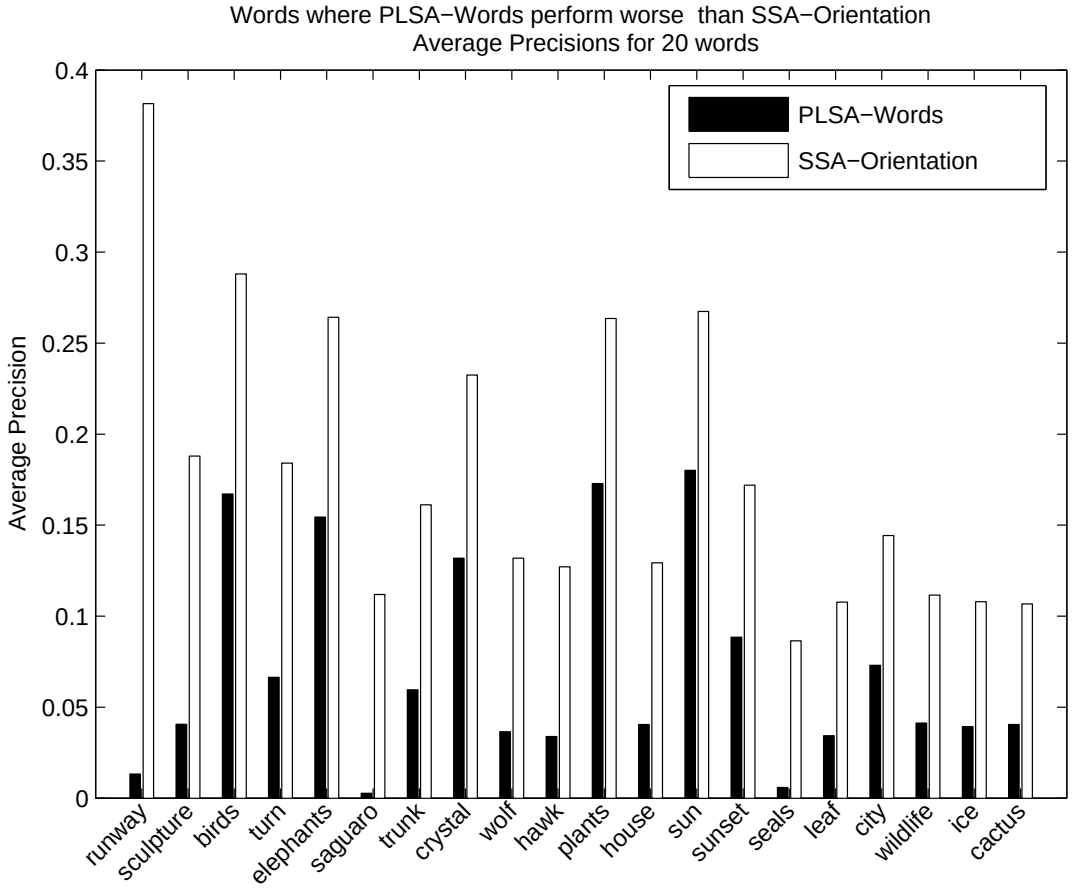

Figure 4.23: Relative average precision improvement for the best 20 words for PLSA-Words vs. SSA-Orientation (500 clusters). Average precision difference is highest to lowest sorted from left to right.

improvement as the word ”elephants” itself. Crystal objects, although do not have a specific orientation, usually display the same detail consistently in different orientations in the same image. This fact is captured by different visterms in SSA resulting in better performance.

Relative MAP reduction for the worst 20 words with respect to SSA-Orientation is shown in Figure 4.26 and the corresponding test images with highest probability decrease for the first 8 words is given in Figure 4.27. Since the word ”black” does not correspond to a specific object, there is not any consistency in orientations which result in worse performance in SSA, as expected. Since windows are shown in a variety of different angles and shapes there is not any consistency in orientation. The word ”night” might potentially refer to many objects with potentially different orientations. The words ”fungus”, ”snake”, ”light, and ”smoke” do not have any consistent orientations either resulting in worse performance, as expected.

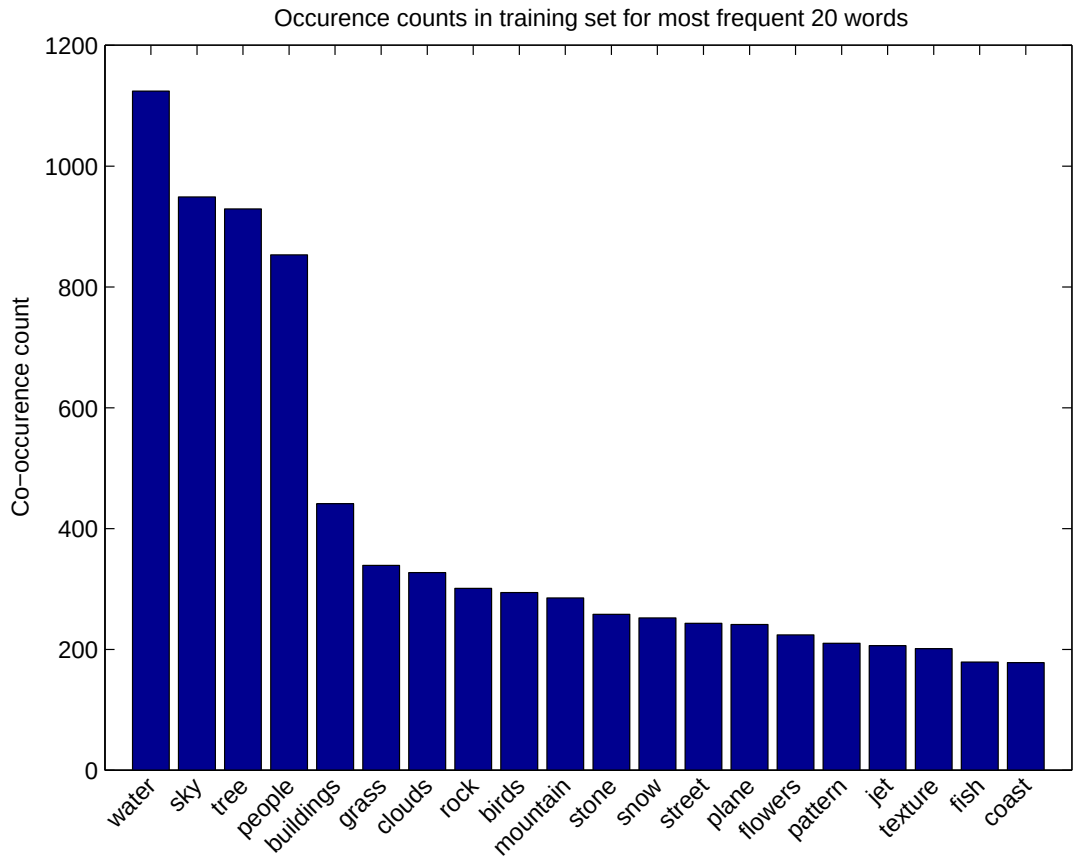

Figure 4.24: Occurrence counts in training set for most frequent 20 words. A relatively high percentage of images are annotated by the word ”Bird”. With around 300 annotated images, the word ”bird” ranks as the sixth most frequently annotated word.

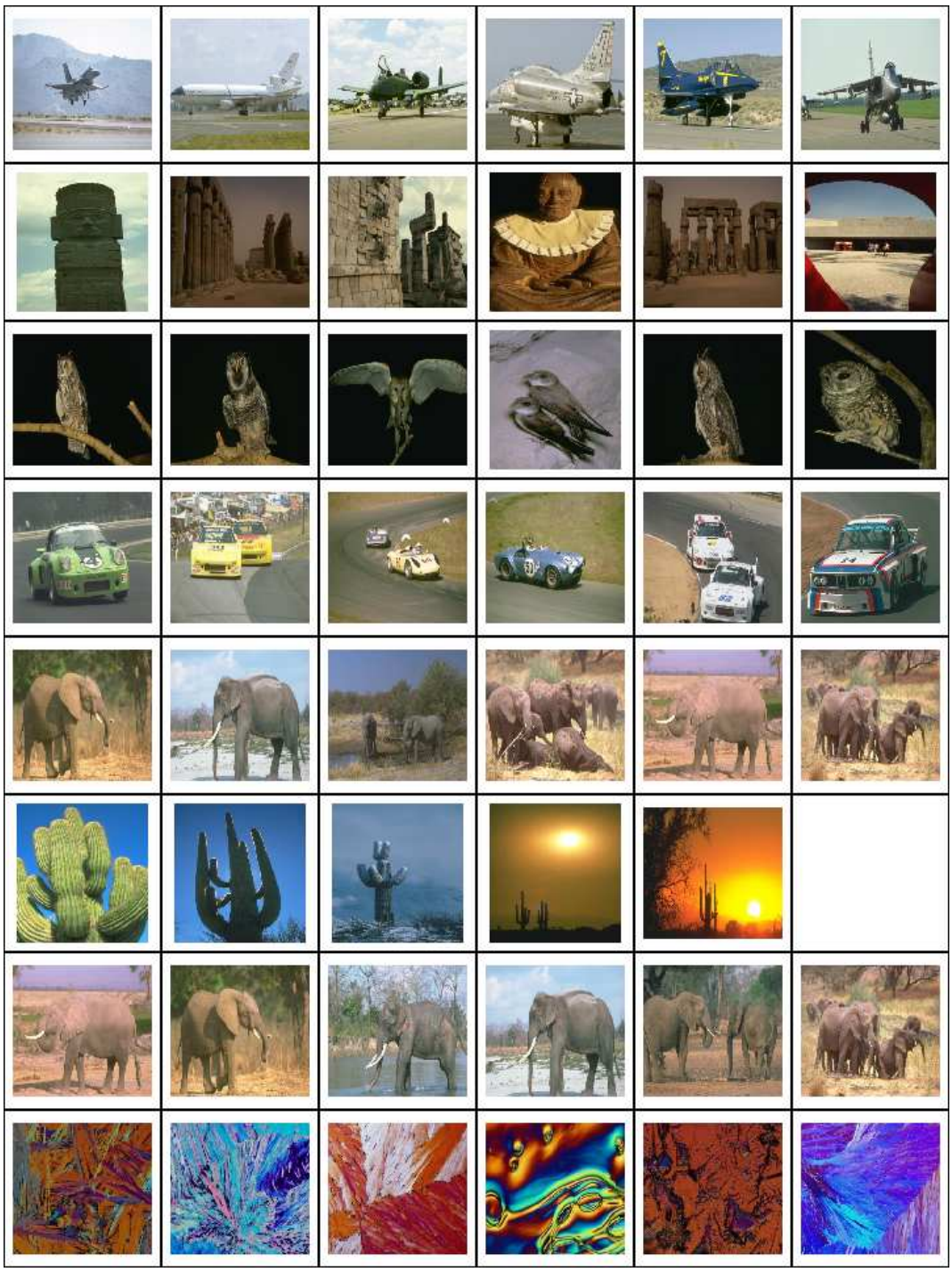

Figure 4.25: Test images with highest average precision improvement for the best 8 words for PLSA-Words vs. SSA-Orientation (500 clusters). Model probability improvement of test images decrease left to right, top to bottom. Each row corresponds to a word. Words top to bottom: runway, sculpture, birds, turn, elephants, saguaro, trunk, crystal.

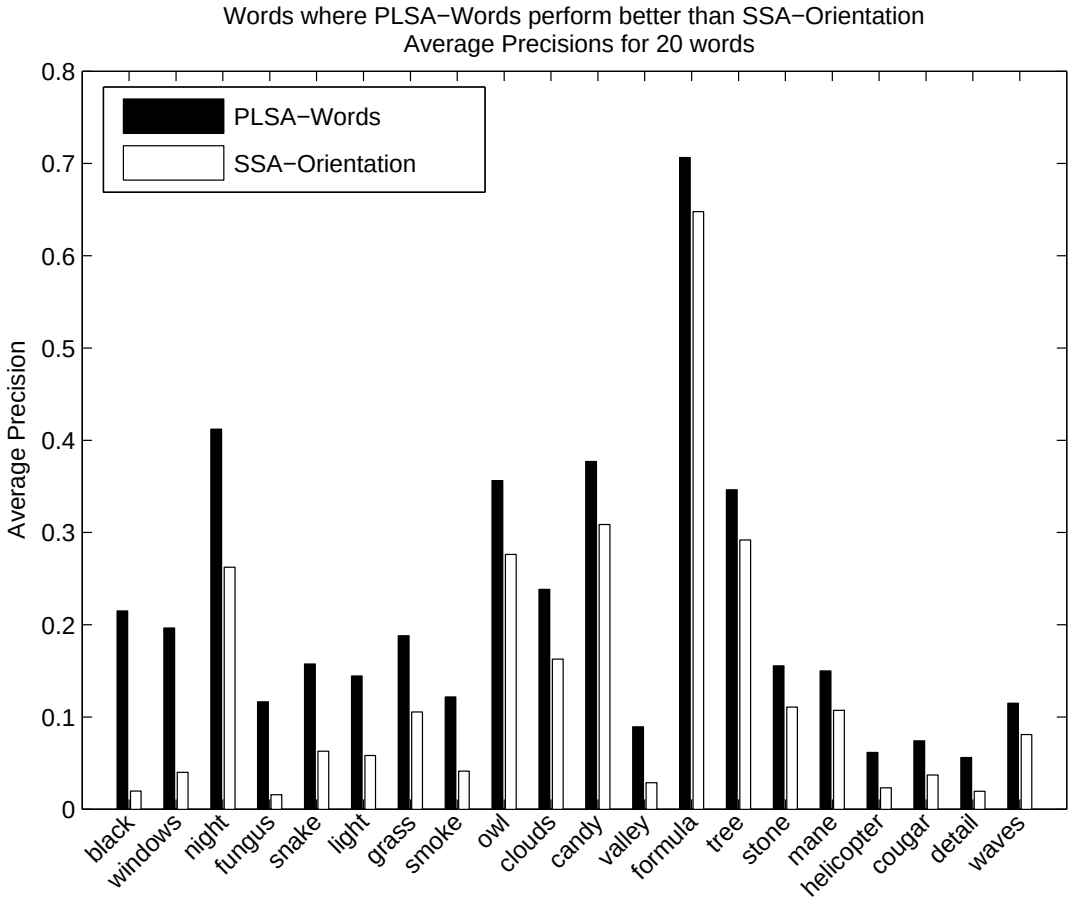

Figure 4.26: Relative average precision reduction for the worst 20 words for PLSA-Words vs. SSA-Orientation (500 clusters). Average precision difference is highest to lowest sorted from left to right.

The above analysis reveals that for the words where PLSA-Words performs better than SSA-Orientation; one needs to update the orientation side information. Let us, now, investigate the behaviors of color side information in the following sub-section.

4.5.2 Per-word Performance of SSA-Color compared with PLSA-Words

In Figure 4.28, we present CAP Curve of SSA-Color with respect to PLSA-Words for 500 visterms. The values above the x-axis show the words, where SSA-Color has a better performance, while the values below the x-axis shows the words where PLSA-Words has a better performance. Note that the black area above the x-axis is larger than that of the area below the x-axis showing that the overall performance of SSA-Color is higher compared to

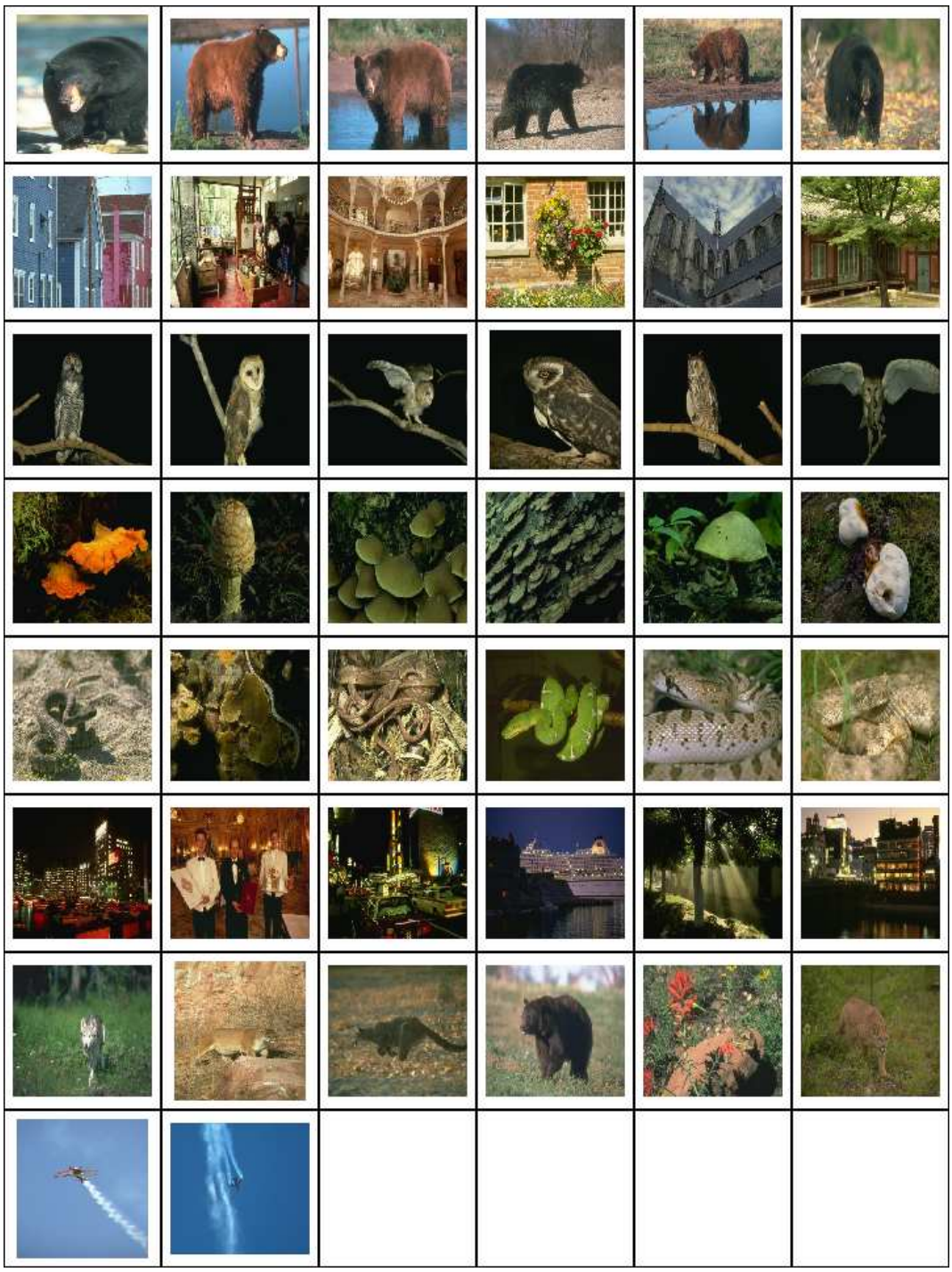

Figure 4.27: Test images with lowest average precision reduction for the worst 8 words for PLSA-Words vs. SSA-Orientation (500 clusters). Model probability reduction of test images decrease left to right, top to bottom. Each row corresponds to a word. Words top to bottom: black, windows, night, fungus, snake, light, grass, smoke.

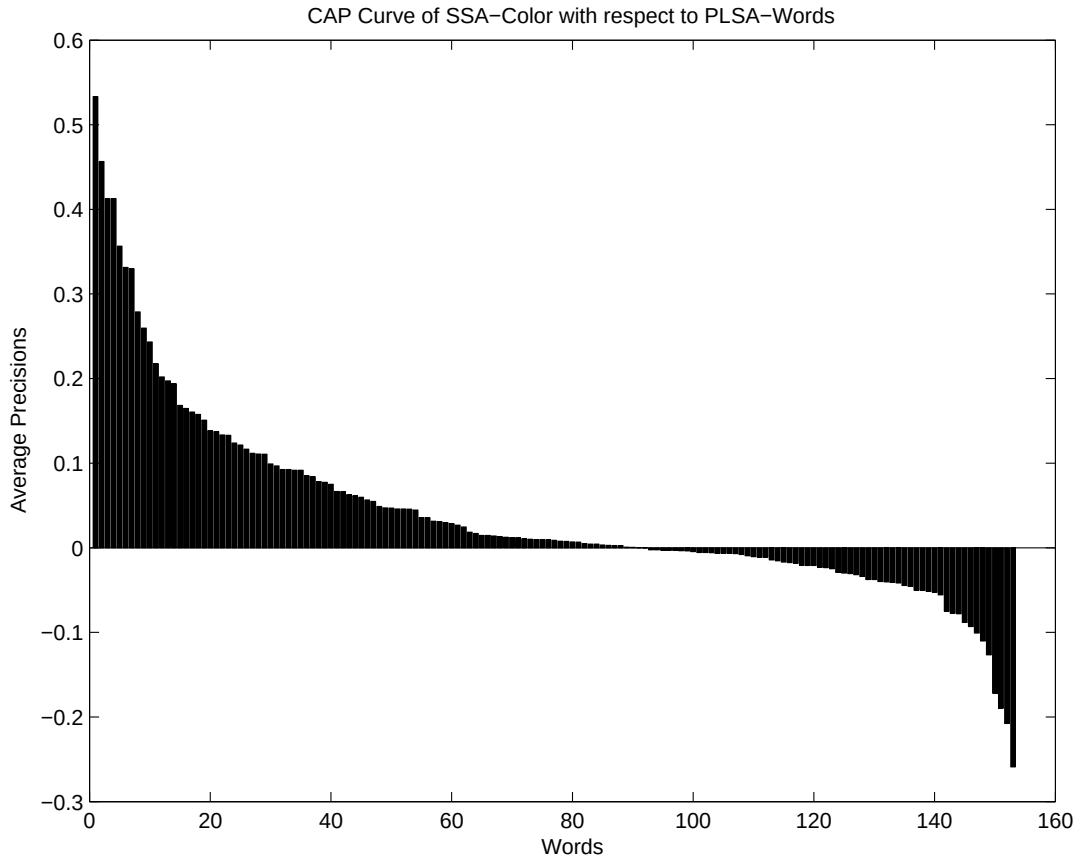

Figure 4.28: CAP Curve of SSA-Color with respect to PLSA-Words for 500 visterms. CAP-percent-better shows the percentage of words where SSA-Color performs better. CAP-total-better and CAP-total-worse, correspond to areas above and below axis, respectively. Higher CAP-total-better and lower CAP-total-worse indicate the superiority of SSA-Color compared to PLSA-Words.

PLSA-Words. With a CAP-percent-better value of 0.59, slightly more than half of the words are better estimated by SSA-Color. CAP-total-better to CAP-total-worse ratio is 3.26, showing that the average precision performance of better-estimated words for SSA-Color is much higher compared to PLSA-Words.

Relative MAP improvement for the best 20 words is shown in Figure 4.29 and the corresponding test images with highest average precision improvement for the first 8 words is given in Figure 4.30. Words that show the highest performance improvement correspond to the objects that have a consistent color. Pumpkins have consistently orange color showing the greatest improvement among all words. Crystal objects usually have color in variations of purple. Fungus, mushrooms and nest usually have the same green tones. Although face and vegetables correspond to a variety of colors in training set; face, vegetables and pumpkins co-occur

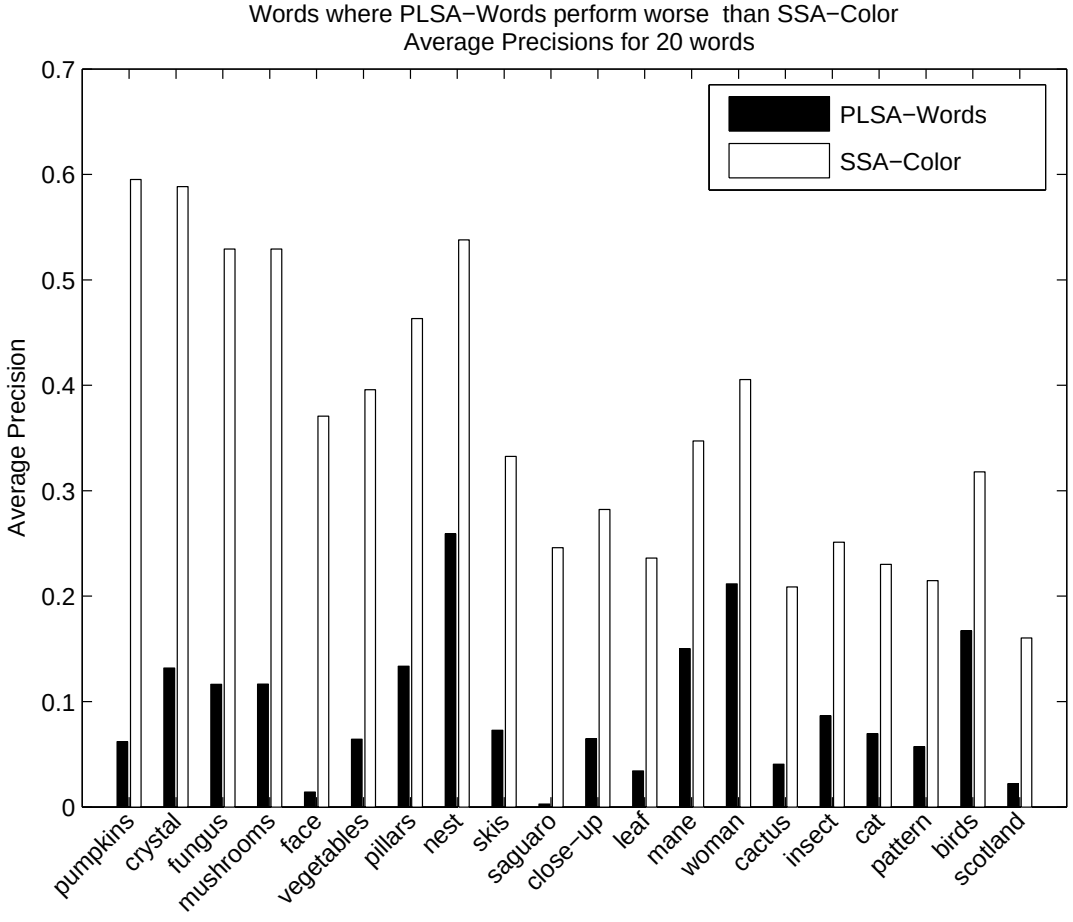

Figure 4.29: Relative average precision improvement for the best 20 words for PLSA-Words vs. SSA-Color (500 clusters). Average precision difference is highest to lowest sorted from left to right.

in many images as seen in Figures 4.31 and 4.32. Hence, the performance increase seen for pumpkins is seen for the words ”face” and ”vegetables” as well. Pillars have usually brown color. As shown in Figure 4.33 test images with the same color gets the highest improvement, using color as side information.

Relative MAP reduction for the worst 20 words with respect to SSA-Color is shown in Figure 4.34 and the corresponding test images with highest probability decrease for the first 8 words is given in Figure 4.35. Annotation ”Herd” corresponds to images with a variety of different colors depending on the type of animal the herd consists of. Since annotation ”Black” corresponds to different objects, namely, bears and helicopters having different textures, and even some brown Bears are annotated as ”Black”, performance decrease occurs for the word ”Bear”. Images annotated with ”Black” in the training set is shown in Figure 4.36. Images

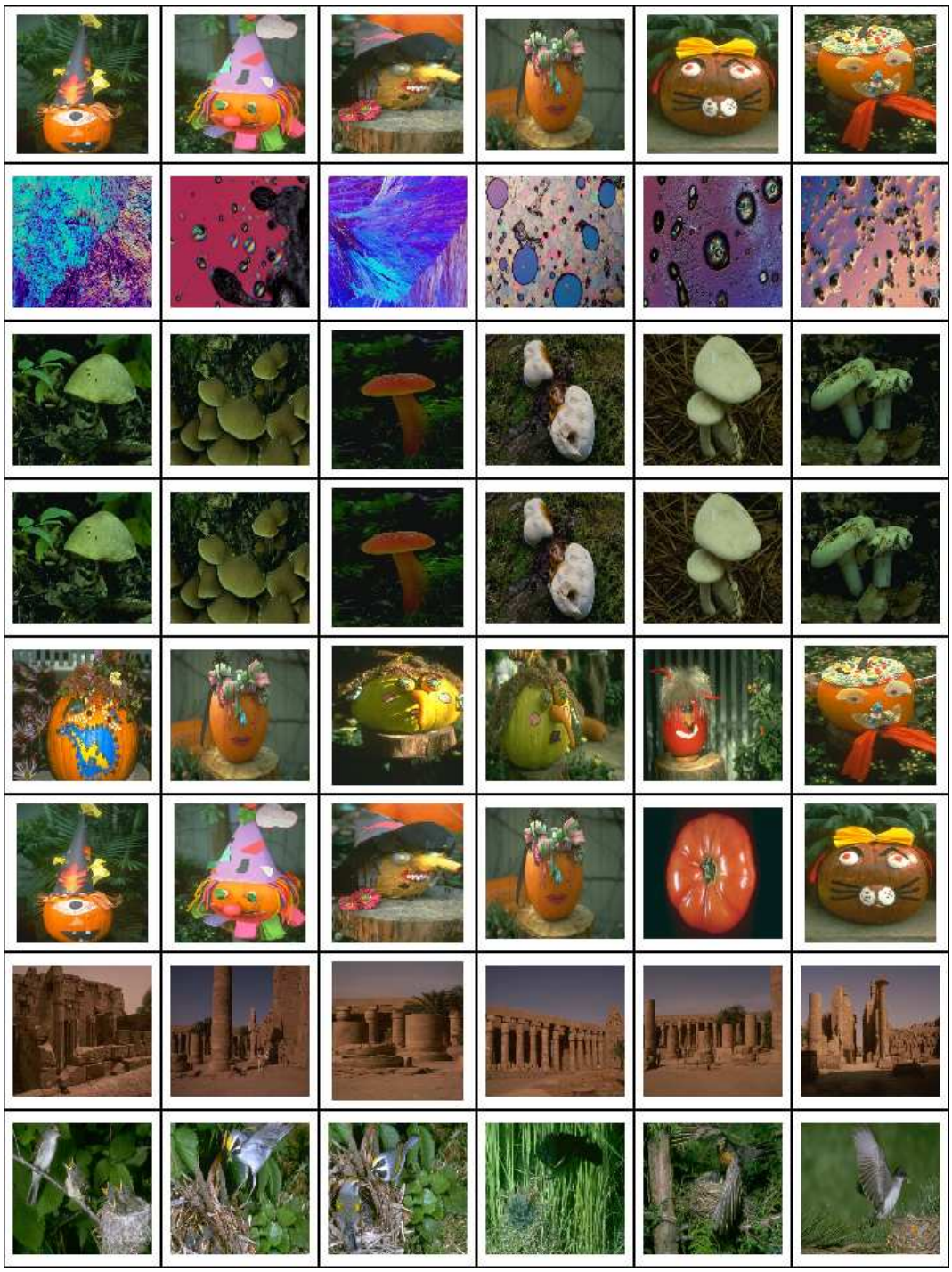

Figure 4.30: Test images with highest average precision improvement for the best 8 words for PLSA-Words vs. SSA-Color (500 clusters). Model probability improvement of test images decrease left to right, top to bottom. Each row corresponds to a word. Words top to bottom: pumpkins, crystal, fungus, mushrooms, face, vegetables, pillars, nest.

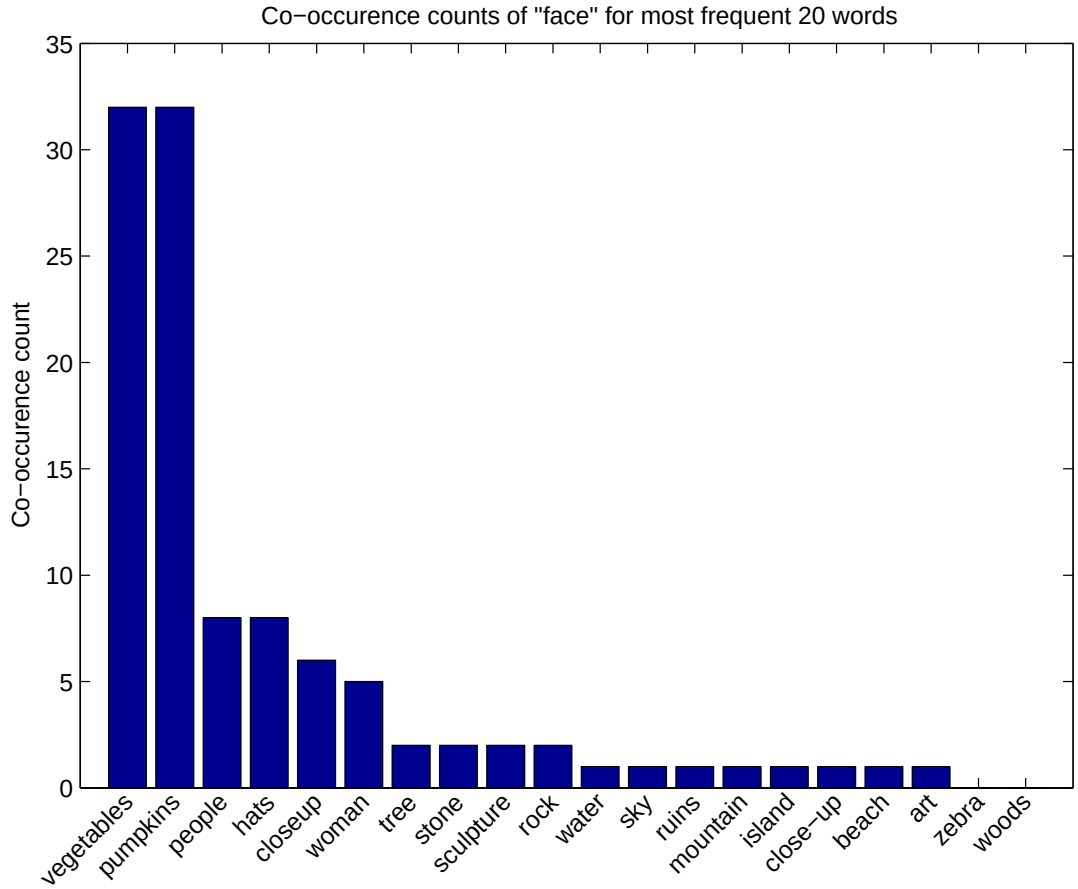

Figure 4.31: Co-occurrence counts of words for the word ”face”. ”Face” and ”vegetables” co-occur in many images. ”pumpkins” is the second most frequently co-annotated word for ”face”.

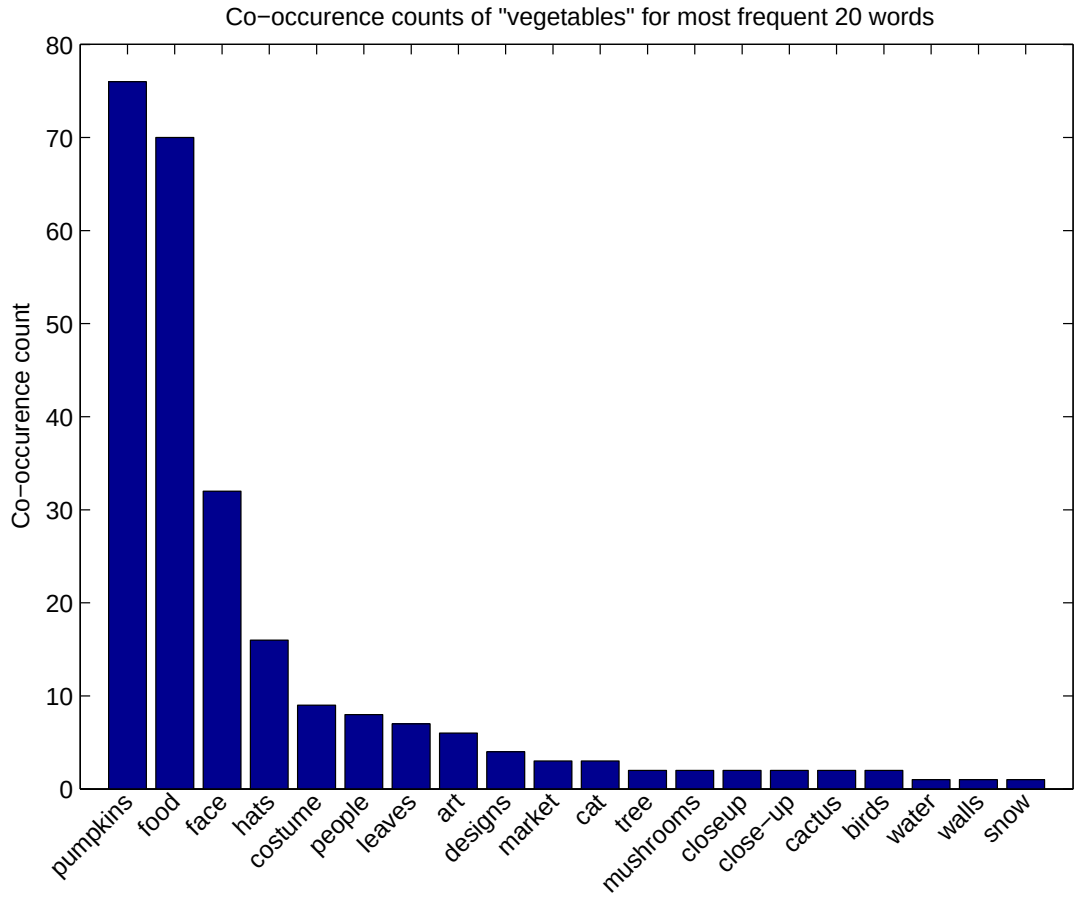

Figure 4.32: Co-occurrence counts of words for the word ”vegetable”. ”vegetable” and ”pumpkins” co-occur in many images. ”pumpkins” is the most frequently co-annotated word for ”vegetable”.

Figure 4.33: Testing set images for the word ”pillars”. Model probability of test images decrease left to right, top to bottom.

annotated with ”Windows”, ”Candy”, ”Light”, ”Snake” and ”Buildings” have a variety of colors. Although tracks have the same background color, usually they appear with cars with different colors.

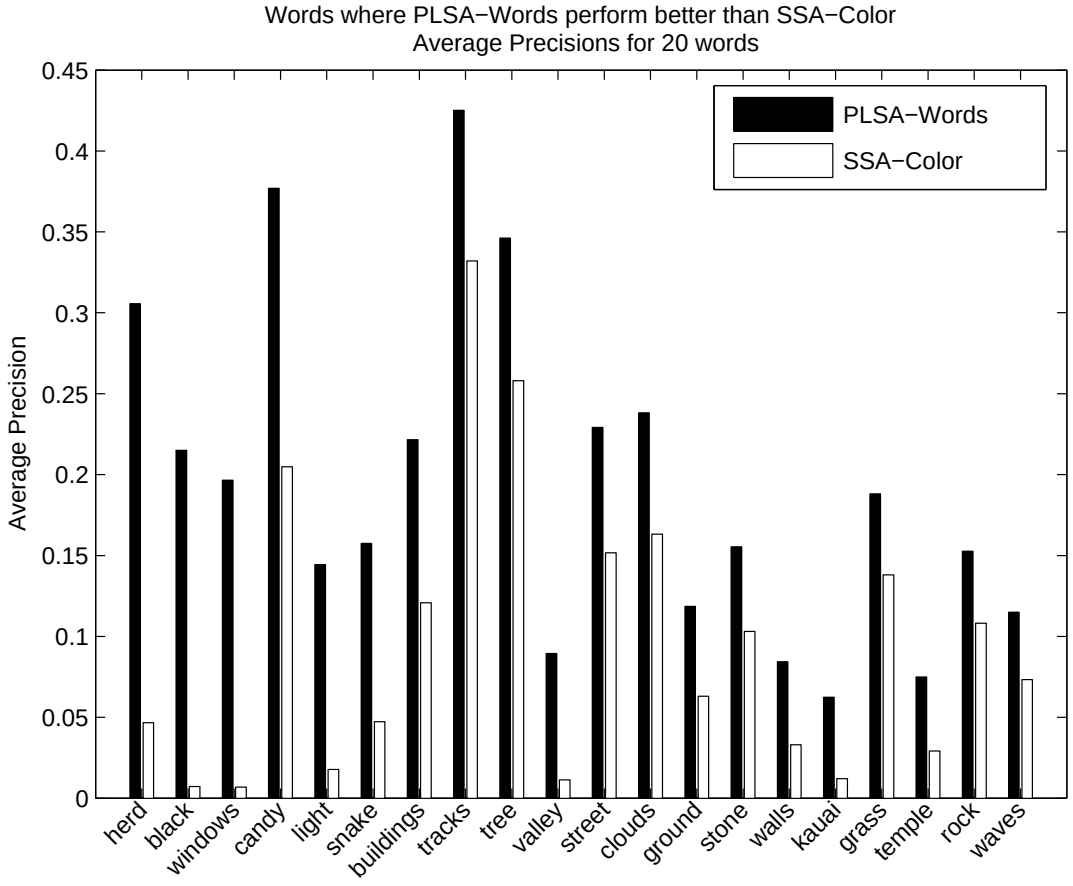

Figure 4.34: Relative average precision reduction for the worst 20 words for PLSA-Words vs. SSA-Color (500 clusters). Average precision difference is highest to lowest sorted from left to right.

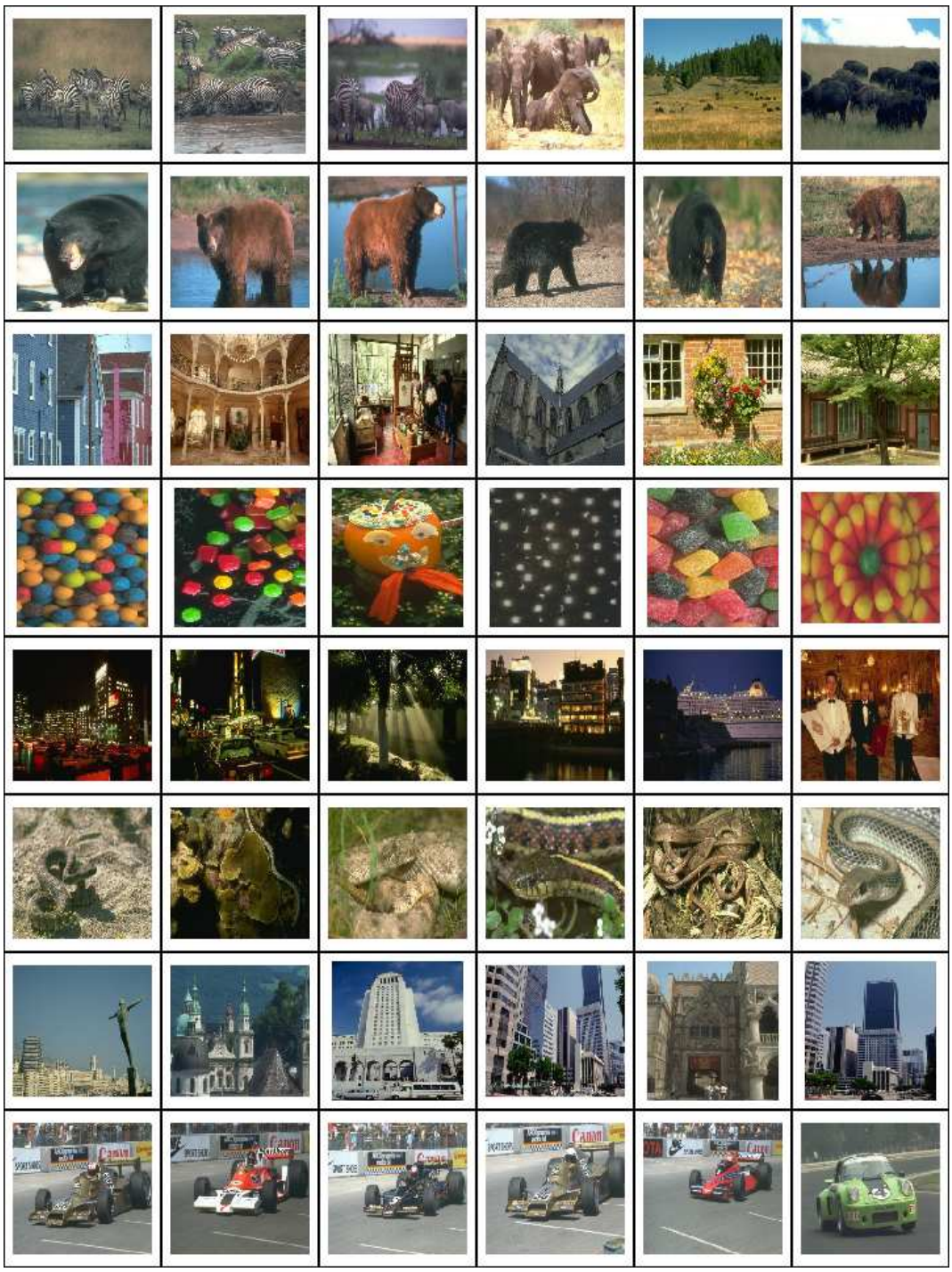

Figure 4.35: Test images with lowest average precision reduction for the worst 8 words for PLSA-Words vs. SSA-Color (500 clusters). Model probability reduction of test images decrease left to right, top to bottom. Each row corresponds to a word. Words top to bottom: herd, black, windows, candy, light, snake, buildings, tracks.

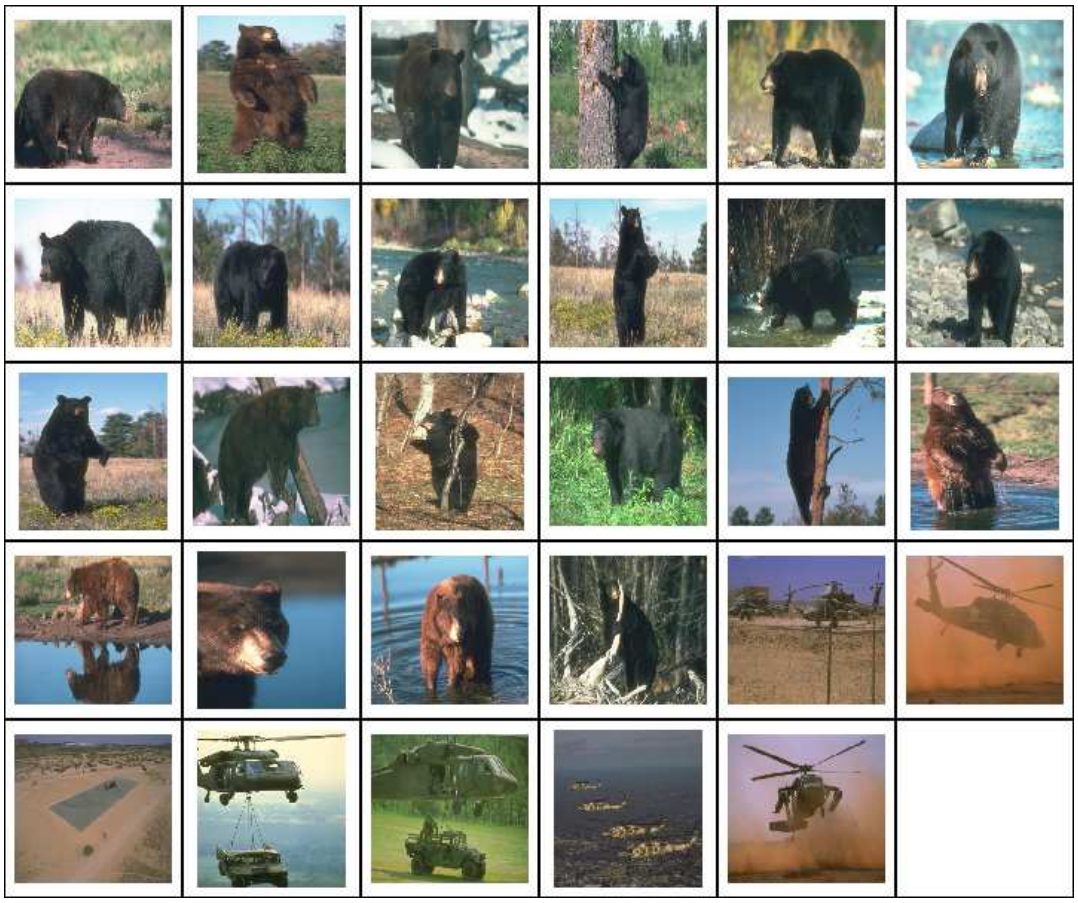

Figure 4.36: Training images for the word ”black”. ”Black” corresponds to different objects, namely, bears and helicopters.

4.6 Entropy Measure of SIFT, SSA-Color and SSA-Orientation Features

In the previous sections, we see that the overall performance increase is obtained for SSA-Color and SSA-Orientation features compared to PLSA-Words of [20]. In this section, we analyze and compare the proposed SSA system and PLSA-Words with respect to the entropy of the clusters, consisting of visterms. This analysis allows us to further understand the reason for the performance increase in the proposed SSA algorithm. We predict that this performance increase results from the fact that by using ”side information” available in the annotated images besides the visual features, clusters become more homogeneous with respect to the provided side information. Therefore, clusters become sharper and the overall entropy of the system is reduced.

We compute the entropy of SIFT for PLSA-Words, SSA-Color and SSA-Orientation features assuming that data points obey a Gaussian mixture distribution. Specifically, given a set of data points $Feature^{ci}$, where c is the cluster label obtained by standard K-means or semi-supervised clustering, $c = 1..K$, and i is the sample id in the cth cluster and $i = 1..n_c$, n_c is the number of data points within cth cluster, we first compute mean μ_c, covariance matrix Σ_c and the prior probability of cluster c by dividing the number of points within cluster c by the total number of points using,

$$prior_c = \frac{n_c}{\sum_{k=1}^{K} n_k} . \tag{4.5}$$

The probability density function of $Feature$ is computed using the Gaussian mixture assumption, from

$$p(Feature) = \sum_{c=1}^{K} prior_c \; p_c(Feature) , \tag{4.6}$$

and

$$p_c(Feature) = \frac{1}{(2\pi)^{d/2}|\Sigma_c|^{1/2}} \exp(-\frac{1}{2}(Feature - \mu_c)^T \Sigma_c^{-1}(Feature - \mu_c)) , \tag{4.7}$$

where d is the dimension of the $Feature$.

Given a set of data points $X = \{Feature_{jm}\}$, $j = 1..s$, $m = 1..f_j$, where f_j is the number

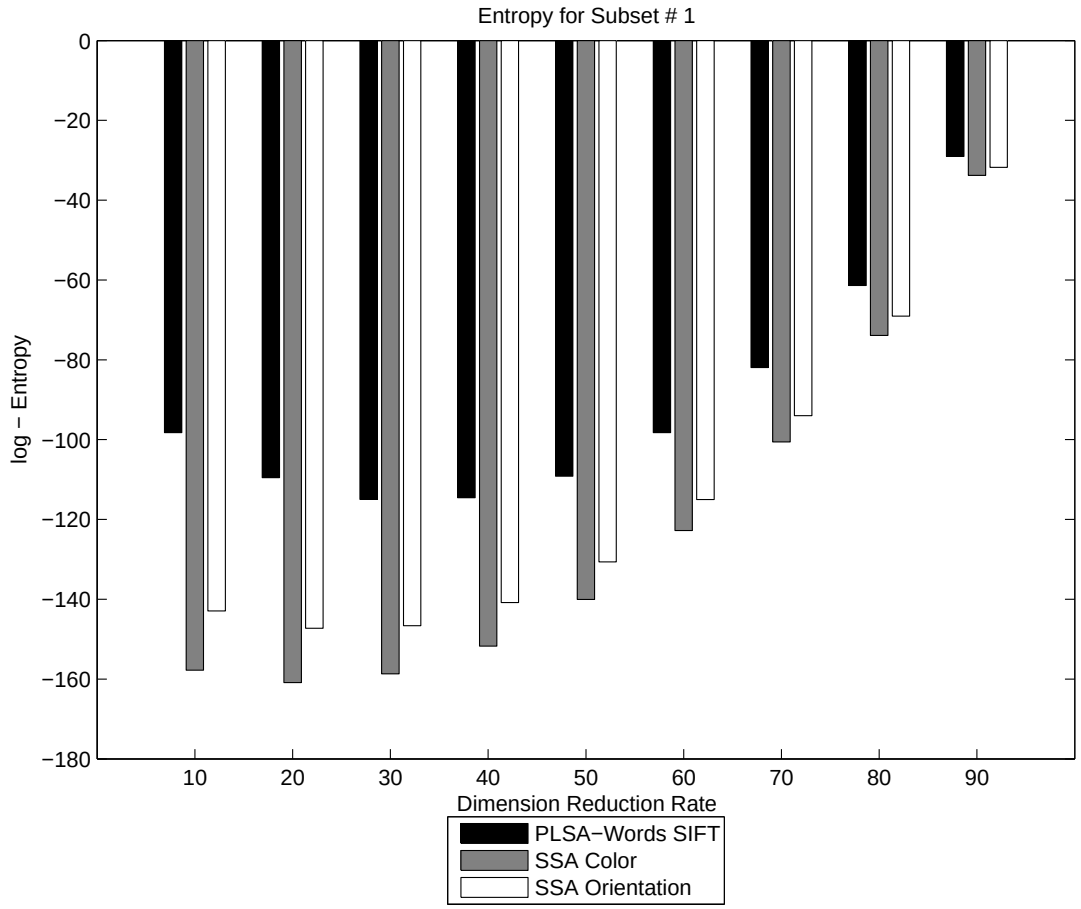

Figure 4.37: Log entropy of SIFT, SSA-Color and SSA-Orientation Features for Subset 1.

of regions in image I_j, corresponding to low level visual feature obtained from region m of image I_j, we compute entropy using:

$$entropy(Feature) = -\sum_{j=1}^{s}\sum_{m=1}^{f_j} p(Feature_{jm}) log(p(Feature_{jm})) \; . \tag{4.8}$$

Entropy computations of PLSA-Word SIFT, SSA-Color and SSA-Orientation features for the first subset of the training set can be seen in Figure 4.37. Entropy values for all the rest of 9 subsets are given in Appendix B. Since some of the covariance matrices become singular, we compute entropy after reducing feature dimension in the order of 10 to 90 percent in increments of 10 using principle component analysis [64]. In all the subsets, a similar behavior is observed. The entropies for SSA-Color and SSA-Orientation are reduced significantly compared to the entropy of PLSA-Words SIFT. The entropy for SSA-Color is lower compared

to SSA-Orientation. This result is consistent with the mean average precision performances of SSA-Color, SSA-Orientation and PLSA-Words. As can be seen in Figures 4.13 and 4.9, mean average precision values for SSA-Color, SSA-Orientation, and PLSA-Word SIFT are around 0.16, 0.13 and 0.11 after 100 number of hidden topics. We, also, observe that the entropy values stabilize after 30 percent of feature reduction rate and continue to increase, as the dimension reduction rate gets higher. This is an expected result since the majority of the information is lost during the dimension reduction.

4.6.1 Summary

In this chapter, we conduct a thorough numerical analysis for the proposed SSA algorithm. We also compare it to the state of the art PLSA-Words algorithm. First, the structure of dataset is explained. Then, in addition to the most frequently used precision, recall and MAP performance metrics, we introduce CAP Curve. Based on CAP Curve, we define three new metrics, namely, CAP-percent-better, CAP-total-better and CAP-total-worse metrics. Next, estimation of system hyper-parameters by cross-validation is provided. Then, performance of the proposed system is compared with PLSA-Words and analyzed in detail.

We observe that the proposed SSA-Topic, SSA-Color and SSA-Orientation algorithms perform better than PLSA-Words in cross validation tests. We, also, show that SSA outperforms PLSA-Words in Corel2003 data set based on precision, recall, MAP metrics as well as the proposed ones based on CAP Curve.

Finally, we measure the entropies of the low level feature spaces. We observe that the entropies for SSA-Color and SSA-Orientation are reduced significantly compared to the entropy of PLSA-Words SIFT. This reduction confirms our predictions that decreasing the randomness in the system enables us to get better annotation results since the side information introduced to the system makes clusters of visual features sharper.

CHAPTER 5

CONCLUSION AND FUTURE DIRECTIONS

Automatic image annotation can be defined as generating a set of annotation words for a given image database, using the training data consisting of images and their annotations. The recent availability of large annotated image data sets, require accurate and fast methods for image annotation.

In this thesis, we propose a new method to improve the image annotation process by enhancing the information content of the system. This task is achieved by introducing the ”side information” concept, which is used to close the semantic gap between the visual and textual information.

Side information is simply defined as the available, but unused information in the image, extracted during the representation of visual and/or textual features. This information is employed to improve the visual and/or textual features extracted from the images. One way of utilization of side information is to constrain the clustering of visual features. In most of the image annotation systems, visual features are clustered to quantize and then to be matched with the annotation words. It is well known that the correspondence between the clusters of visual features and the words is rather poor due to the huge semantic gap between the low level features based on color, texture and/or shape and high level context information of words. Side information is used in such a way that, while clustering visual features, those features with the same side information are constrained to fall in the same group of clusters.

To embed the available side information to the clustering process of visual features, first, we define and quantize the side information. For this purpose, we cluster the side information features collected from the annotated images. Each cluster label obtained this way corresponds to a group. We assign visual clusters to groups, so that each group is assigned approximately

equal number of clusters. Next, each visual feature is associated with a group or set of groups depending on its co-existing side information. For clustering of visual features, we use a modified version of K-Means. We constrain the visual feature clustering process with the available side information, in such a way that visual points falling in the same cluster should all have the same group associations.

Although it can be formulated in many different ways, in this thesis, we define three different types of side information. The first one is the annotation keywords, which is global, high level information for the annotated image. This information is associated with visual features extracted from all the regions of a given image and can be represented by the hidden topic probabilities obtained during the PLSA algorithm. Hidden topics of PLSA algorithm correspond to a presumed group of words, which corresponds to a topic. To associate with visual features, we select ”highly likely” topics after clustering the hidden topic probabilities into two ”likely” and ”not likely” clusters. Then, we associate the hidden likely topics to the ”Blob features” extracted from regions obtained through N-Cut region segmentation algorithm. To keep clusters as apart from each other as possible and visual features within each cluster as close in Euclidean distance as possible, we apply Linear Discriminant Analysis to the clustering results.

The second side information we define is the orientation information. This side information is used for semi-supervising the clustering of SIFT features. Orientation information is extracted from an interest point based on an orientation histogram computed from gradient orientations of sample points that are within a region around the interest point. The dominant direction obtained from the peaks in the orientation histogram is used as the side information. The orientation information based on the dominant direction is quantized into a number, depending on the number of groups desired for this side information.

The third side information we define, is the color information around each interest point. Color information is obtained through K-means clustering of Luv color features around interest points. We associate this side information with SIFT features as in the case for orientation side information. Both orientation and color side information provide additional cues for clustering of SIFT features resulting in better annotation results.

We compare the proposed system SSA to PLSA-Words based on precision, recall and MAP metrics that are most frequently used in the literature. In this thesis, we propose a new set

of metrics based on the CAP Curve. The proposed metrics are very handy in comparing the performances of two annotation systems. The distribution of relative per-word performances of two different annotation algorithms, can be seen by making use of CAP Curve. Moreover, metrics defined on CAP Curve enable one to see the percentage of words that are better estimated by any of the two algorithms and the total average performances of words that are estimated better/worse than the other algorithm. We demonstrate that SSA gives better results compared to PLSA-Words on all the metrics mentioned above.

Both standard K-means and semi-supervised K-Means algorithms proposed in this study have been implemented in a high performance parallel computation environment. Parallelism has been implemented with MPI library, based on message passing. Both sequential and parallel versions of semi-supervised clustering algorithms have less computational complexity in distance calculations based on the number of groups used for side information.

We obtain two major benefits by using the ”side information” available in the annotated images besides the visual features that are clustered. First, we show that the overall entropy of the system is reduced due to the information induced into visual features through the side information. Clusters become more homogeneous with respect to the provided side information and have sharper probability density functions. By reducing the randomness of visual features, we improve the annotation performance. Second, since we compare a visual feature with not all of the cluster centers, but with only a subset of it, depending on the constraints provided by the side information, we can complete clustering in shorter time compared to the classical clustering algorithms. The more the number of groups we choose for side information, the less computation overhead for distance calculations we obtain.

One should note that, the selection and definition of side information requires careful analysis of the application domain. The questions of what side information to use and which visual features to associate with do not have easy answers. Side information selected based on intuition needs to be validated through cross-validation tests in the training set. Moreover, it is difficult to find generic side information, which is valid for all the words in the vocabulary. Selected side information might give better annotation results for some words but not for the others depending on the visual representation of words in images. Therefore, selection of side information is a domain dependent process and needs to be defined to improve the information content of the low level visual feature clusters.

5.1 Future Directions

The following are some directions for future research:

- We only used some of the side information that is available in images together with their annotations. Other information such as position of regions can be used as side information to improve annotation performance. Some of the visual content always appears relatively at the same positions within an image. For example ”sky” is usually at the top pixels/areas whereas ”grass” usually appears at the bottom of images. Constraining clustering of visual features depending on the position might improve performance.

- Not all the textual words benefit equally for specific side information as far as the annotation performance is concerned. An annotation model, which uses a different set of visterms constructed from different side information for each word might give better annotation results. For example, for word ”pumpkin”, which is heavily represented by the orange color, it might be better to use visterms obtained from color side information whereas for ”walls” which usually have vertical edges, an annotation model based on visterms obtained from orientation side information could be used.

- Bag of words representation obtained from local features is often criticized for losing information about the spatial relationship between interest points. Using visual features obtained from segmented regions as side information for local features within the regions might be a remedy to this problem. Associating the local features with side information at the region level adds context to local features. Hence, local information is combined with more global ones elevating the information content of local features.

- It is possible to define a hierarchical SSA by using global side information such as annotation keywords, for visual features from segmented regions, and using visterms obtained this way as side information for local features within each segmented region. Propagating global side information towards the local features might increase the annotation performance.

- Another open issue in this thesis is to make use of not only the available but unused side information in images together with their annotations, but also, to employ other external information sources such as WordNet [65]. WordNet provides information

about semantically related words that can be integrated as a link with image annotations as side information. Wordnet can be used to extract additional side information based on the annotation keywords by finding semantically similar keywords. We expect that the hidden topics obtained from annotation keywords combined with the semantically related words provide better annotation results when used as side information.

- We have used semi-supervised clustering of visual features technique only in the context of image annotation. Recently, there has been an increasing interest of using visual codebooks in other problem domains such as object categorization and image retrieval [66], [67], [68], [69], [70], [71], [72], [73], [74] as well. We expect that semi-supervised clustering of visual features using visual side information, increases the performance of these methods.

As outlined above, there are many ways to improve the proposed method in this thesis. However, the crucial issue remains is how to improve the information content of the annotation system for closing the semantic gap.

REFERENCES

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

[2] R. Datta, D. Joshi, J. Li, and J. Z. Wang. Image retrieval: Ideas, influences, and trends of the new age. *ACM Computing Surveys*, 40(2):1–60, April 2008.

[3] C. Carson, S. J. Belongie, H. Greenspan, and J. Malik. Blobworld: Image segmentation using expectation-maximization and its application to image querying. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(8):1026–1038, August 2002.

[4] V. E. Ogle and M. Stonebraker. Chabot: Retrieval from a relational database of images. *IEEE Computer*, 28(9):40–48, September 1995.

[5] S. Sclaroff, L. Taycher, and M. La Cascia. Imagerover: A content-based image browser for the world wide web. In *Workshop on Content-Based Access of Image and Video Libraries*, page 2, 1997.

[6] J. Dowe. Content-based retrieval in multimedia imaging. In *Storage and Retrieval for Image and Video Databases (SPIE)*, pages 164–167, 1993.

[7] A. P. Pentland, R. W. Picard, and S. Sclaroff. Photobook: Content-based manipulation of image databases. *International Journal of Computer Vision*, 18(3):233–254, June 1996.

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

[9] C. Faloutsos, R. J. Barber, M. D. Flickner, J. Hafner, C. W. Niblack, and W. R. Equitz. Efficient and effective querying by image content. *Journal of Intelligent Information Systems*, 3(3-4):231–262, July 1994.

[10] C. Nastar, M. Mitschke, C. Meilhac, and N. Boujemaa. Surfimage: a flexible content-based image retrieval system. In *Proceedings of the 6th ACM International Conference on Multimedia (Multimedia-98)*, pages 339–344, September 12–16 1998.

[11] Welcome to flickr - photo sharing, www.flickr.com, last visited on december 2009.

[12] Google videos, video.google.com, last visited on december 2009.

[13] 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.

[14] P. Duygulu, K. Barnard, J. F. G. de Freitas, and D. A. Forsyth. Object recognition as machine translation: Learning a lexicon for a fixed image vocabulary. In *ECCV '02: Proceedings of the 7th European Conference on Computer Vision-Part IV*, pages 97–112, 2002.

[15] D. M. Blei and M. I. Jordan. Modeling annotated data. In *SIGIR '03: Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, pages 127–134, 2003.

[16] J. Jeon, V. Lavrenko, and R. Manmatha. Automatic image annotation and retrieval using cross-media relevance models. In *SIGIR '03: Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, pages 119–126, 2003.

[17] V. Lavrenko, R. Manmatha, and J. Jeon. A model for learning the semantics of pictures. In *Advances in Neural Information Processing Systems 16*. 2004.

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

[19] J. Li and J. Z. Wang. Real-time computerized annotation of pictures. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(6):985–1002, June 2008.

[20] F. Monay and D. Gatica-Perez. Modeling semantic aspects for cross-media image indexing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(10):1802–1817, October 2007.

[21] G. Carneiro, A. B. Chan, P. J. Moreno, and N. Vasconcelos. Supervised learning of semantic classes for image annotation and retrieval. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(3):394–410, March 2007.

[22] J. Liu, M. J. Li, Q. S. Liu, H. Q. Lu, and S. D. Ma. Image annotation via graph learning. *Pattern Recognition*, 42(2):218–228, February 2009.

[23] Ö. Öztimur and F. T. Yarman Vural. Hanolistic: A hierarchical automatic image annotation system using holistic approach. In *International Symposium on Computer and Information Sciences (ISCIS)*, 2008.

[24] X. J. Wang, L. Zhang, X. R. Li, and W. Y. Ma. Annotating images by mining image search results. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(11):1919–1932, November 2008.

[25] C. Julien. Automatic handling of digital image repositories: A brief survey. In *ISMIS*, volume 4994 of *Lecture Notes in Computer Science*, pages 410–416, 2008.

[26] T. Hofmann. Probabilistic latent semantic indexing. In *SIGIR*, pages 50–57, 1999.

[27] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2000.

[28] P. Quelhas and J. M. Odobez. Natural scene image modeling using color and texture visterms. In *CIVR*, pages 411–421, 2006.

[29] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.

[30] K. Barnard, P. Duygulu, D. Forsyth, N. de Freitas, D. M. Blei, and M. I. Jordan. Matching words and pictures. *J. Mach. Learn. Res.*, 3:1107–1135, 2003.

[31] C. Harris and M. J. Stephens. A combined corner and edge detector. In *Alvey Conference*, pages 147–152, 1988.

[32] H. Bay, T. Tuytelaars, and L. J. Van Gool. SURF: Speeded up robust features. In *ECCV*, pages I: 404–417, 2006.

[33] E. Rosten and T. Drummond. Machine learning for high-speed corner detection. In *ECCV*, pages I: 430–443, 2006.

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

[35] T. Pajdla, M. Urban, O. Chum, and J. Matas. Robust wide baseline stereo from maximally stable extremal regions. In *BMVC*, page 3D and Video, 2002.

[36] A. Hanbury. A survey of methods for image annotation. *J. Vis. Lang. Comput*, 19(5):617–627, 2008.

[37] J. Tang and P. H. Lewis. A study of quality issues for image auto-annotation with the corel dataset. *IEEE Transactions on Circuits and Systems for Video Technology*, 17(3):384–389, March 2007.

[38] J. B. MacQueen. Some methods for classification and analysis of multivariate observations. In *Proceedings of 5-th Berkeley Symposium on Mathematical Statistics and Probability*, 1967.

[39] A. Dempster, N. Laird, and D. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, Series B*, 39(1):1–38, 1977.

[40] A. Gelman, J. B. Carlin, H. S. Stern, and D. B. Rubin. *Bayesian Data Analysis*. Chapman and Hall, 2003.

[41] N. Vasconcelos. Image indexing with mixture hierarchies. In *CVPR*, pages 3–10, 2001.

[42] M.R. Gray J.M. Keller and J.A. Givens. A fuzzy k-nearest neighbor algorithm. *IEEE Transactions on Systems, Man, and Cybernetics*, 15(4):580–585, 1985.

[43] A. Demiriz, K. P. Bennett, and M. J. Embrechts. Semi-supervised clustering using genetic algorithms, October 25 1999.

[44] K. Wagstaff, C. Cardie, S. Rogers, and S. Schroedl. Constrained K-means clustering with background knowledge. In *Proc. 18th International Conf. on Machine Learning*, pages 577–584, 2001.

[45] S. Basu, A. Banerjee, and R. J. Mooney. Semi-supervised clustering by seeding. In *ICML*, pages 27–34, 2002.

[46] M. H. C. Law, A. P. Topchy, and A. K. Jain. Model-based clustering with probabilistic constraints. In *SDM*, 2005.

[47] M. R. Garey and D. S. Johnson. *Computers and Intractability*. Freeman and Company, 1979.

[48] D. Cohn, R. Caruana, and A. McCallum. Semi-supervised clustering with user feedback. Technical report, Cornell University, February 12 2003.

[49] D. Klein, S. D. Kamvar, and C. D. Manning. From instance-level constraints to space-level constraints: Making the most of prior knowledge in data clustering. In *ICML*, 2002.

[50] A. Bar-Hillel, T. Hertz, N. Shental, and D. Weinshall. Learning distance functions using equivalence relations. In *ICML*, pages 11–18, 2003.

[51] T. Hertz, A. Bar-Hillel, and D. Weinshall. Boosting margin based distance functions for clustering. In *ICML*, volume 69 of *ACM International Conference Proceeding Series*, 2004.

[52] H. Chang and D. Y. Yeung. Locally linear metric adaptation with application to semi-supervised clustering and image retrieval. *Pattern Recognition*, 39(7):1253–1264, July 2006.

[53] A. Sayar and F. T. Yarman Vural. Image annotation with semi-supervised clustering. In *IEEE 16th Signal Processing, Communication and Applications Conference*, 2008.

[54] A. Sayar and F. T. Yarman Vural. Image annotation by semi-supervised clustering constrained by sift orientation information. In *International Symposium on Computer and Information Sciences (ISCIS)*, 2008.

[55] A. Sayar and F. T. Yarman Vural. Image annotation with semi-supervised clustering. In *International Symposium on Computer and Information Sciences (ISCIS)*, 2009.

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

[57] X. Shen C. M. Brown M. Boutell, J. Luo. Learning multi-label scene classification. *Pattern Recognition*, 37(9):1757–1771, 2004.

[58] A. K. Jain, M. N. Murty, and P. J. Flynn. Data clustering: a review. *ACM Comput. Surv.*, 31(3):264–323, 1999.

[59] R. Xu and D. Wunsch. Survey of clustering algorithms. *IEEE Transactions on Neural Networks*, 16(3):645–678, May 2005.

[60] I. S. Dhillon and D. S. Modha. A data-clustering algorithm on distributed memory multiprocessors. *Lecture Notes in Computer Science*, 1759:245–260, 2000.

[61] M. Snir, S. W. Otto, S. Huss-Lederman, D. W. Walker, and J. Dongarra. *MPI: The Complete Reference*. Scientific and Engineering Computation Series. MIT Press, Cambridge, MA, 1996.

[62] W. Gropp, E. Lusk, and A. Skjellum. *Using MPI, Portable Parallel Programming with the Message Passing Interface*. MIT Press.

[63] D. E. Culler, R. M. Karp, D. Patterson, A. Sahay, E. E. Santos, K. E. Schauser, R. Subramonian, and T. von Eicken. LogP: A practical model of parallel computation. *Communications of the ACM*, 39(11):78–85, November 1996.

[64] I. T. Jolliffe. *Principal Component Analysis*. Series in Statistics. Springer Verlag, 2002.

[65] C. Fellbaum. *WordNet: An Electronic Lexical Database*. MIT Press, 1998.

[66] F. Monay, P. Quelhas, J. M. Odobez, and D. Gatica Perez. Contextual classification of image patches with latent aspect models. *EURASIP Journal on Image and Video Processing*, 2009.

[67] F. F. Li and P. Perona. A bayesian hierarchical model for learning natural scene categories. In *CVPR*, pages II: 524–531, 2005.

[68] P. Quelhas, F. Monay, J. M. Odobez, D. Gatica Perez, and T. Tuytelaars. A thousand words in a scene. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(9):1575–1589, September 2007.

[69] P. Quelhas and J. M. Odobez. Multi-level local descriptor quantization for bag-of-visterms image representation. In *CIVR*, pages 242–249, 2007.

[70] M. Marszaek and C. Schmid. Spatial weighting for bag-of-features. In *CVPR*, pages II: 2118–2125, 2006.

[71] G. Wang, Y. Zhang, and F. F. Li. Using dependent regions for object categorization in a generative framework. In *CVPR*, pages II: 1597–1604, 2006.

[72] J. Sivic and A. Zisserman. Video google: A text retrieval approach to object matching in videos. In *ICCV*, pages 1470–1477, 2003.

[73] C. Dance, J. Willamowski, G. Csurka, and C. Bray. Categorizing nine visual classes with bags of keypoints. In *International Conference on Pattern Recognition*, 2004.

[74] A. Bosch, A. Zisserman, and X. Munoz. Scene classification via pLSA. In *ECCV*, pages IV: 517–530, 2006.

APPENDIX A

WORD FREQUENCIES IN ALL 10 SUBSETS OF THE TRAINING SET

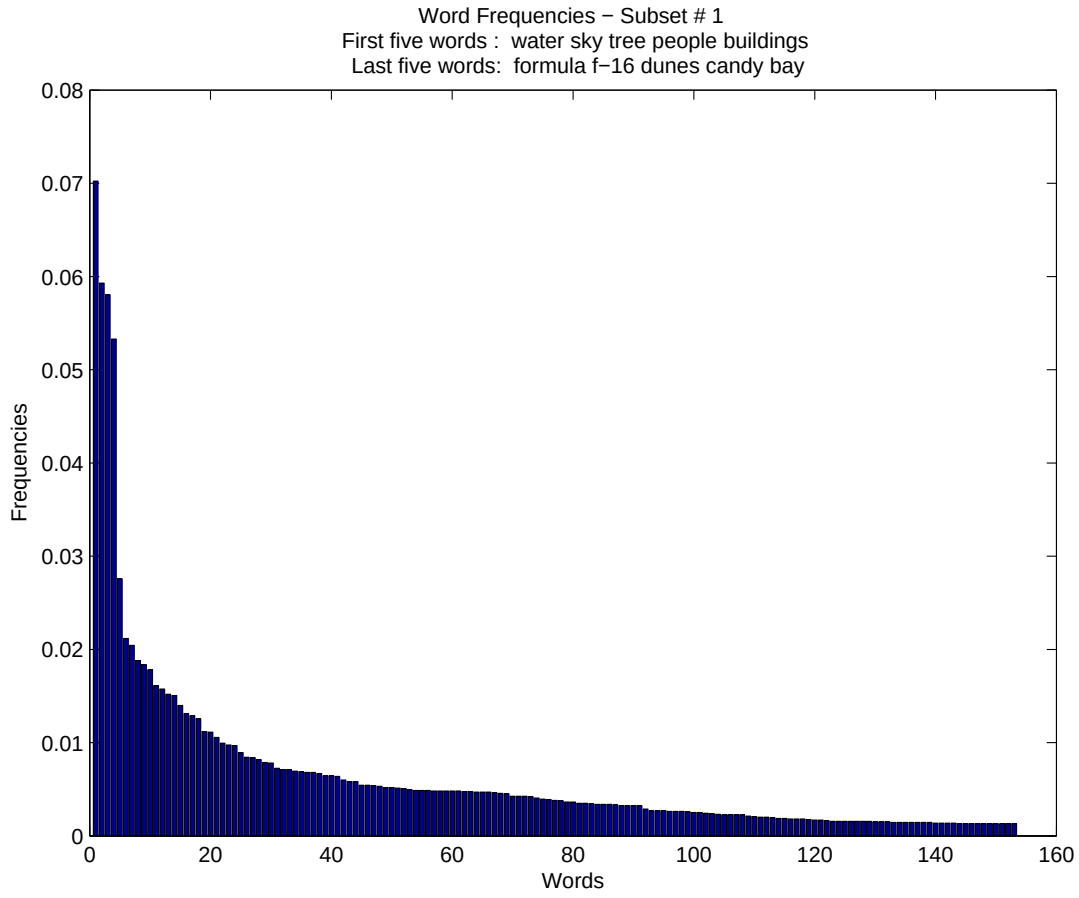

Figure A.1: Word frequencies in subset 1. Words are sorted based on their frequencies in decreasing order from left to right. Most frequent, and least frequent 5 words are listed at the top of the figure.

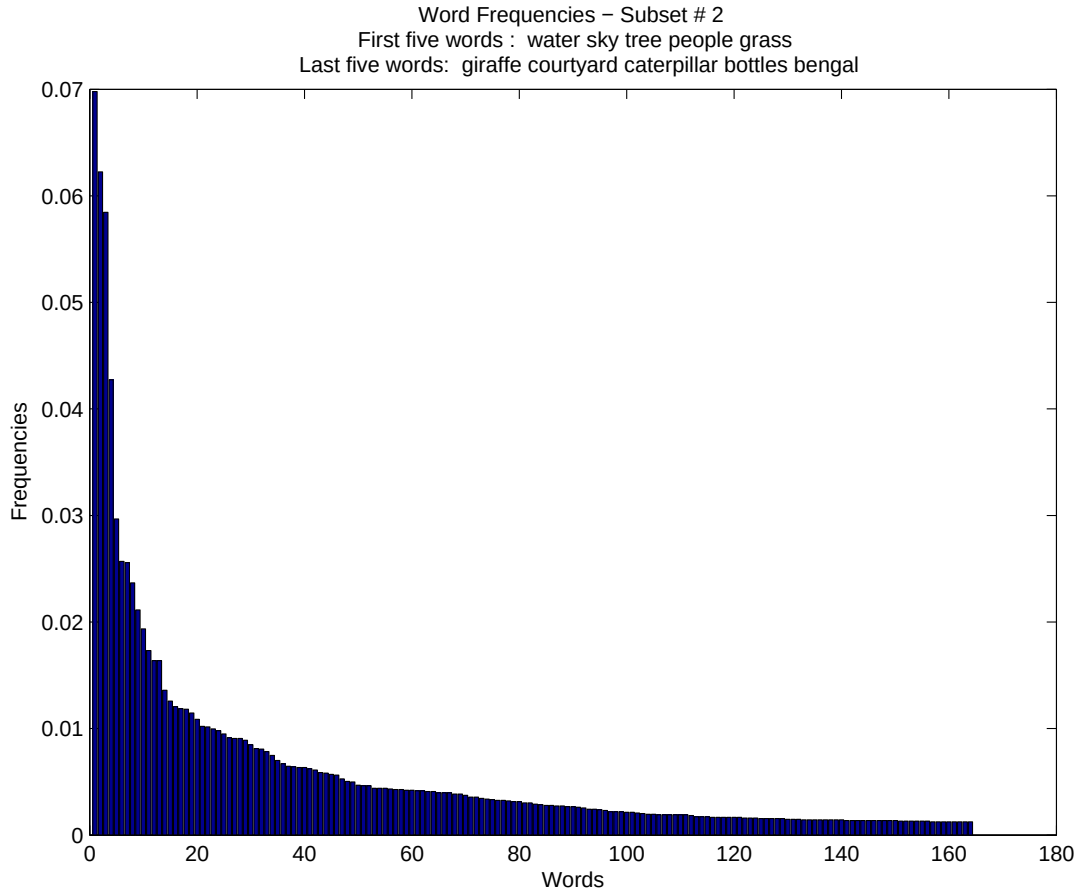

Figure A.2: Word frequencies in subset 2. Words are sorted based on their frequencies in decreasing order from left to right. Most frequent, and least frequent 5 words are listed at the top of the figure.

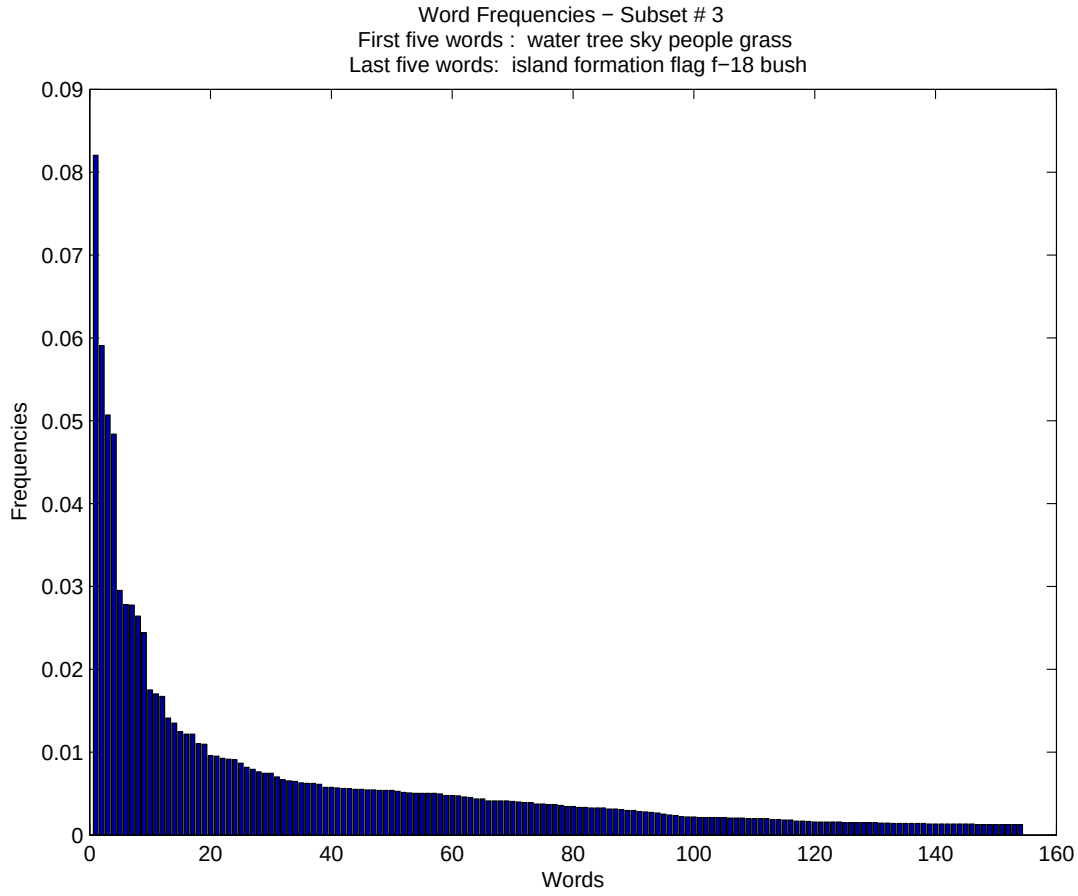

Figure A.3: Word frequencies in subset 3. Words are sorted based on their frequencies in decreasing order from left to right. Most frequent, and least frequent 5 words are listed at the top of the figure.

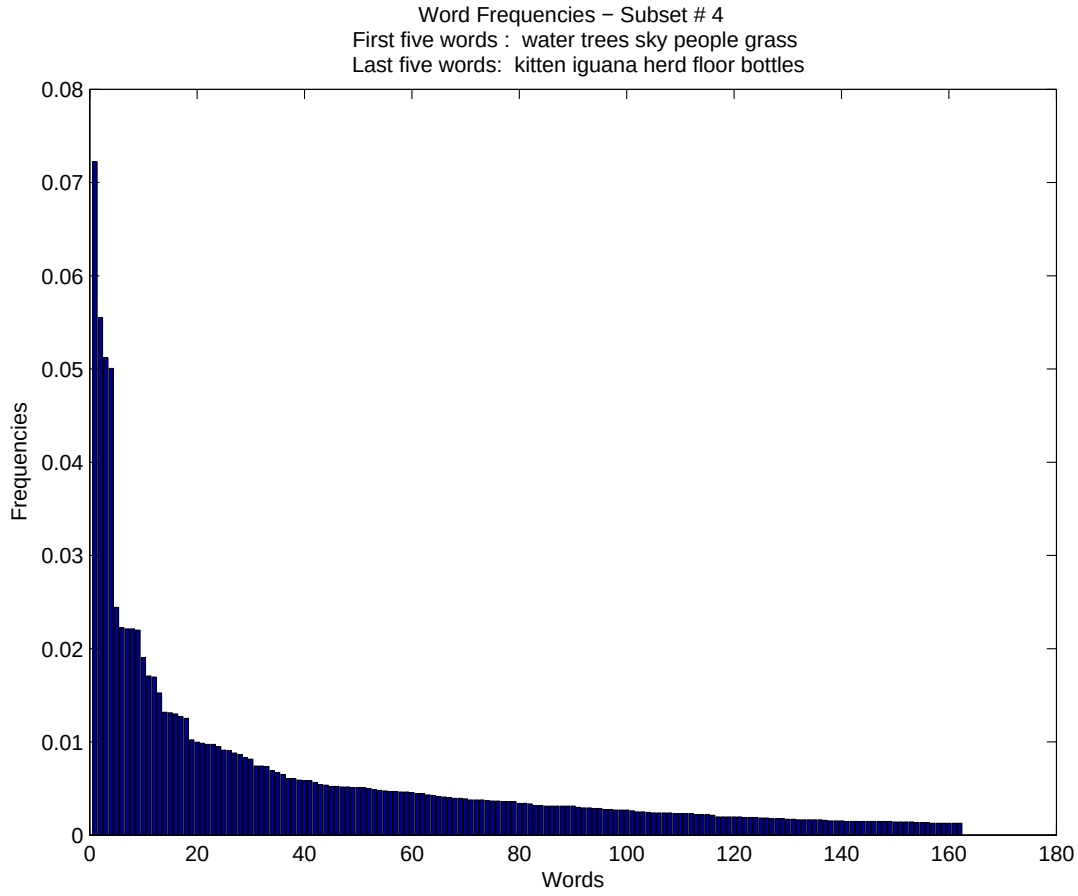

Figure A.4: Word frequencies in subset 4. Words are sorted based on their frequencies in decreasing order from left to right. Most frequent, and least frequent 5 words are listed at the top of the figure.

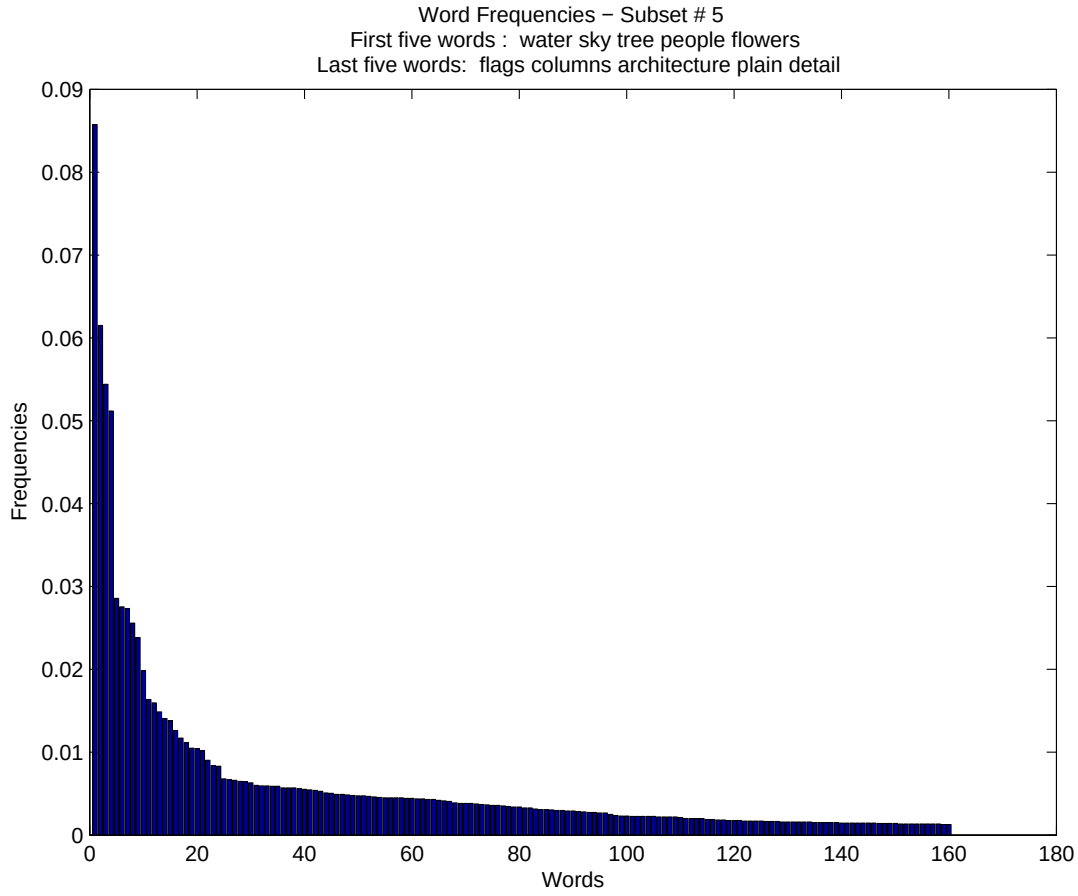

Figure A.5: Word frequencies in subset 5. Words are sorted based on their frequencies in decreasing order from left to right. Most frequent, and least frequent 5 words are listed at the top of the figure.

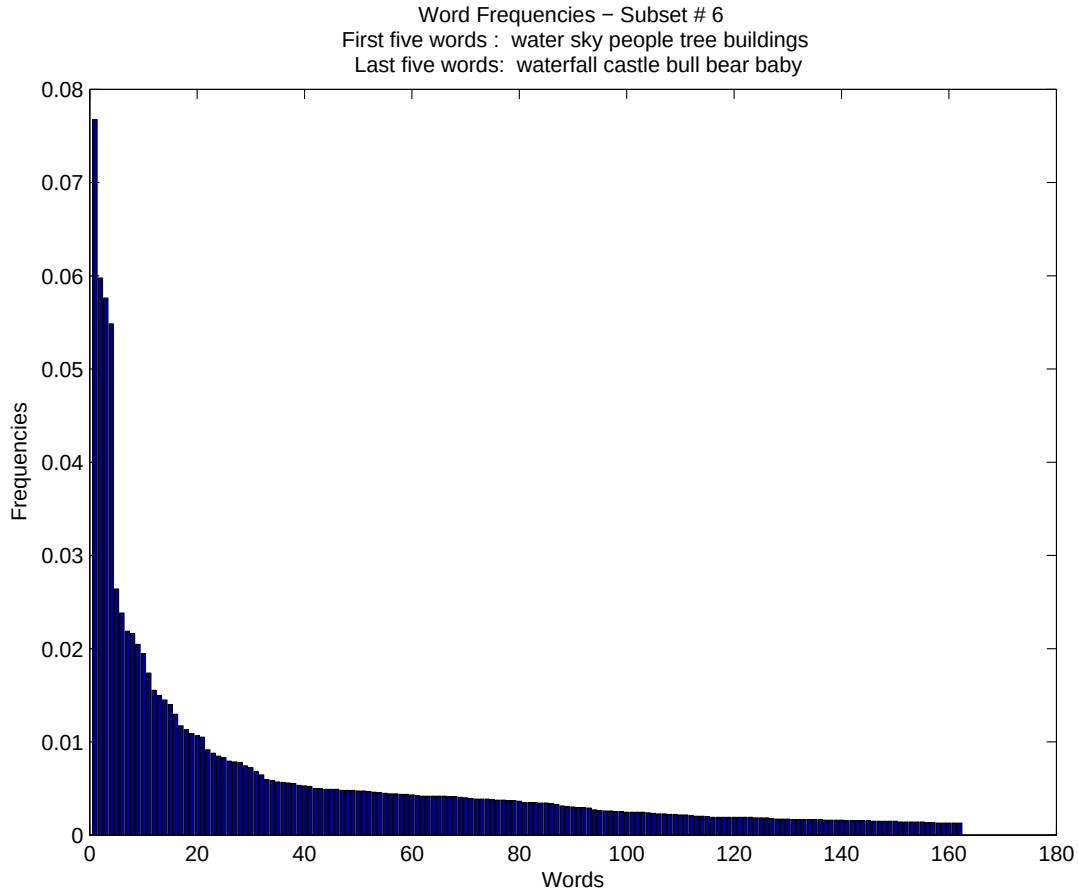

Figure A.6: Word frequencies in subset 6. Words are sorted based on their frequencies in decreasing order from left to right. Most frequent, and least frequent 5 words are listed at the top of the figure.

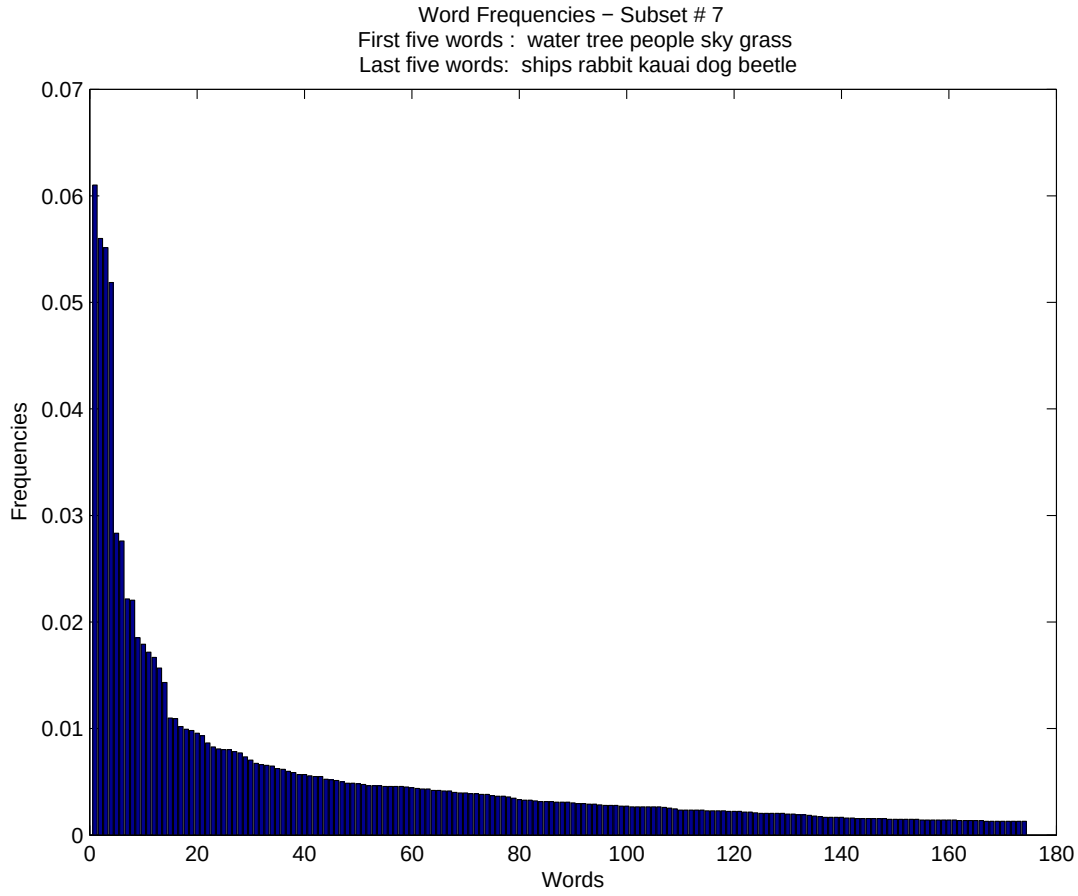

Figure A.7: Word frequencies in subset 7. Words are sorted based on their frequencies in decreasing order from left to right. Most frequent, and least frequent 5 words are listed at the top of the figure.

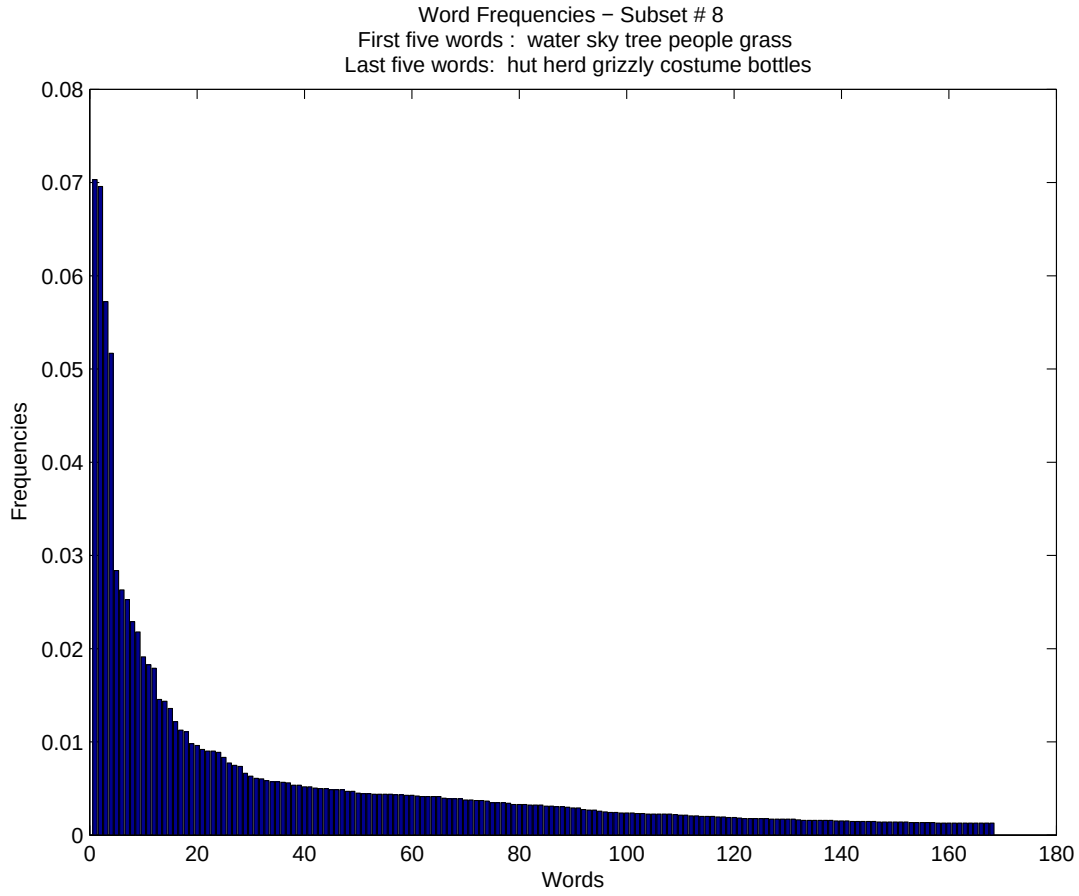

Figure A.8: Word frequencies in subset 8. Words are sorted based on their frequencies in decreasing order from left to right. Most frequent, and least frequent 5 words are listed at the top of the figure.

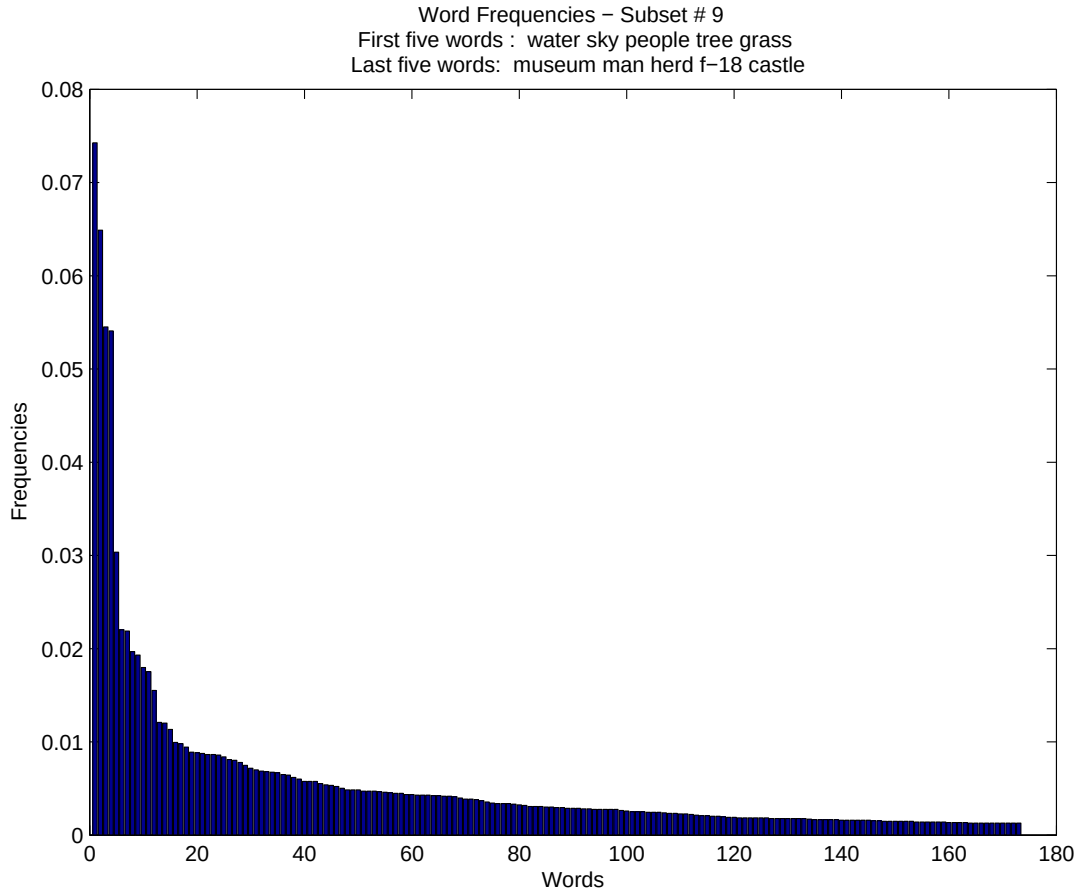

Figure A.9: Word frequencies in subset 9. Words are sorted based on their frequencies in decreasing order from left to right. Most frequent, and least frequent 5 words are listed at the top of the figure.

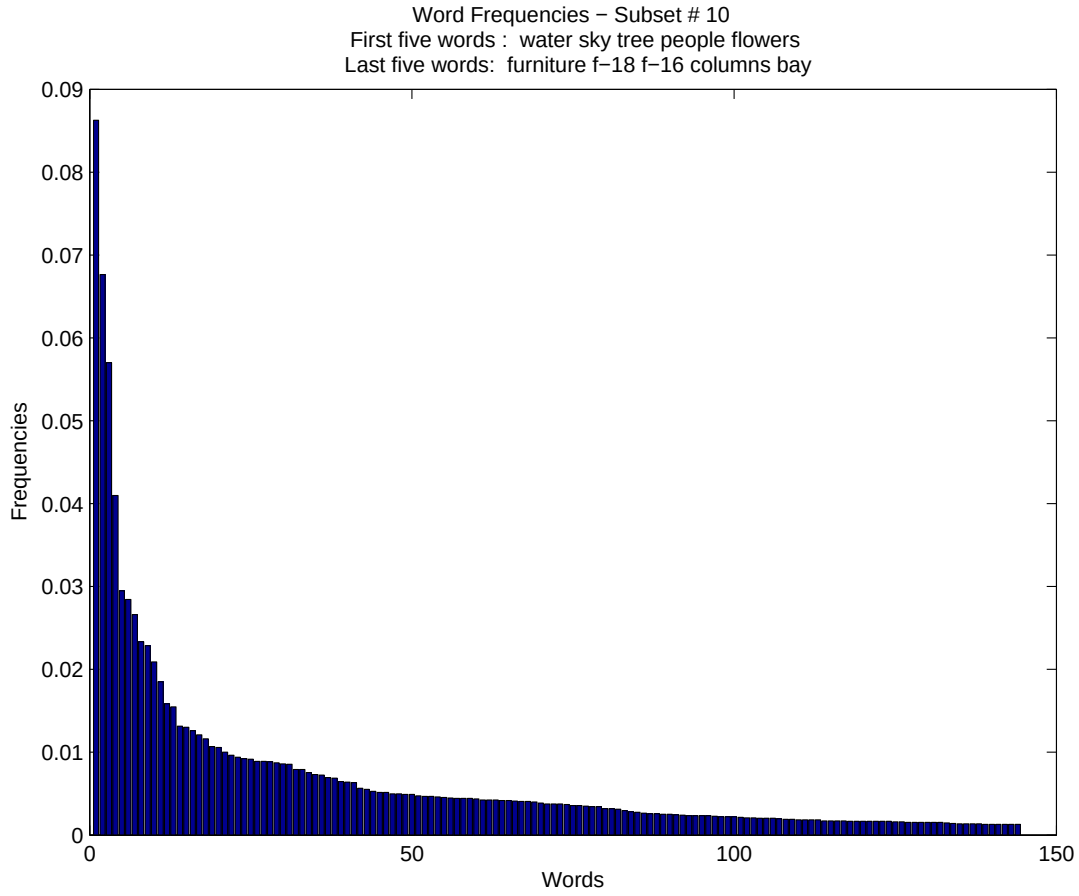

Figure A.10: Word frequencies in subset 10. Words are sorted based on their frequencies in decreasing order from left to right. Most frequent, and least frequent 5 words are listed at the top of the figure.

APPENDIX B

ENTROPY VALUES FOR SUBSETS 2-9 OF THE TRAINING SET

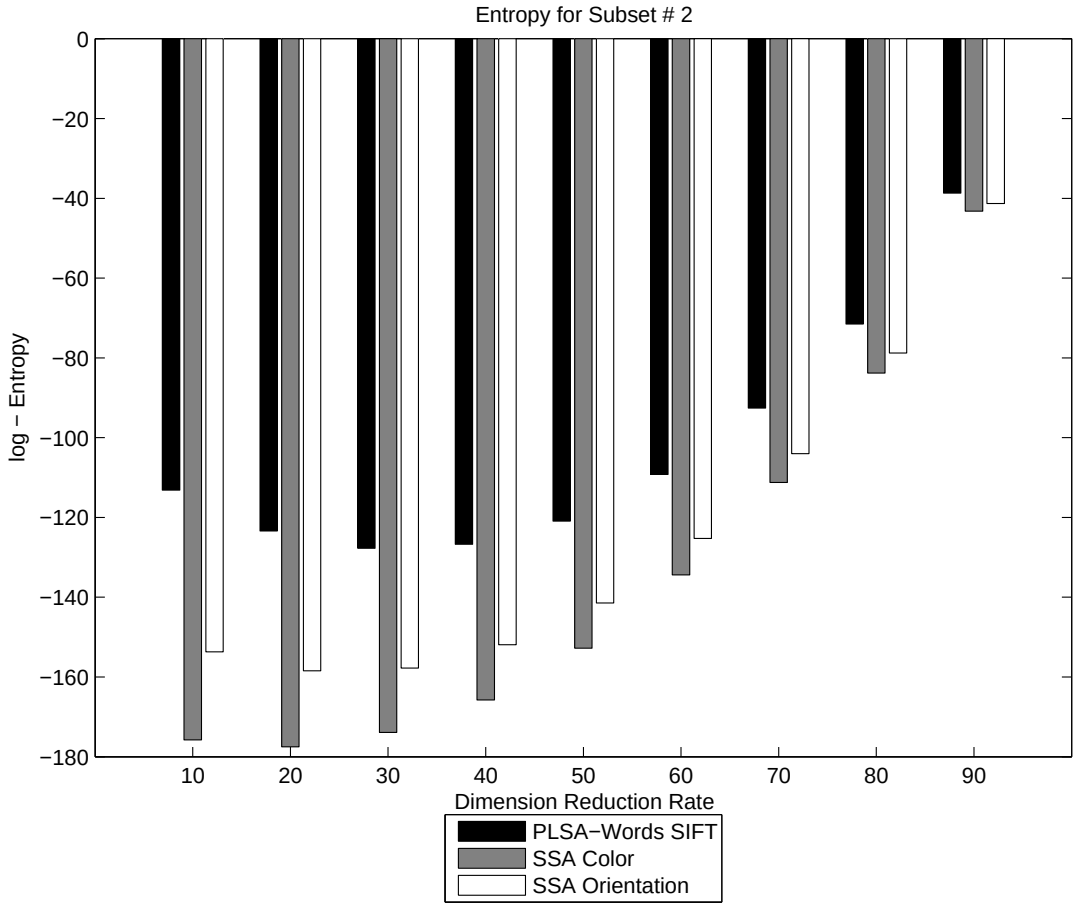

Figure B.1: Log entropy of SIFT, SSA-Color and SSA-Orientation Features for Subset 2.

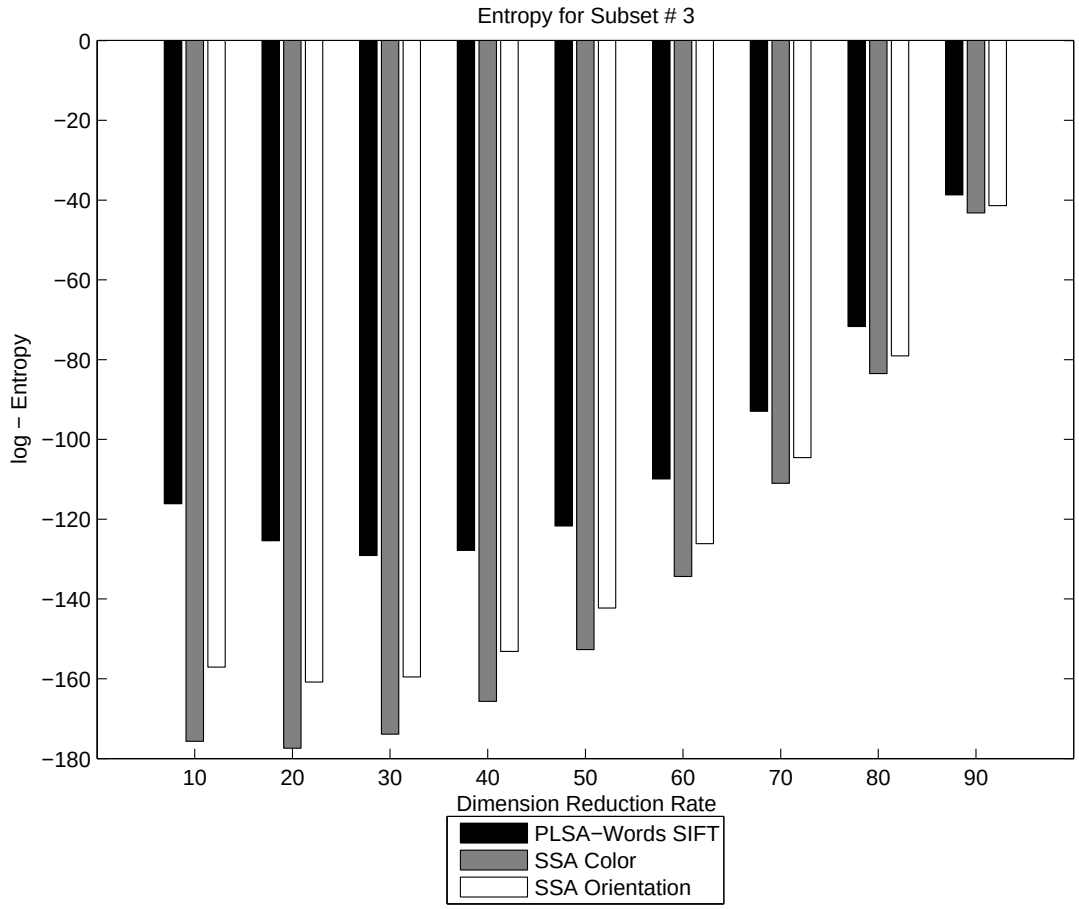

Figure B.2: Log entropy of SIFT, SSA-Color and SSA-Orientation Features for Subset 3.

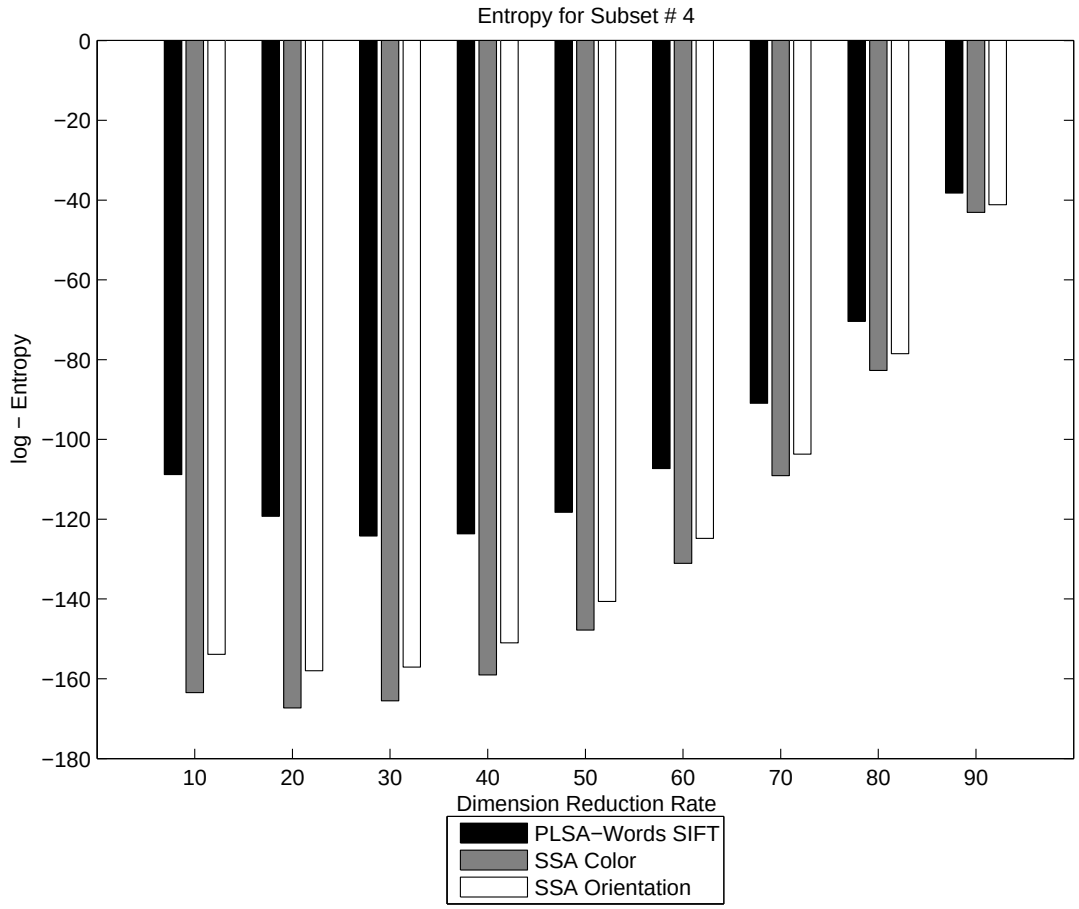

Figure B.3: Log entropy of SIFT, SSA-Color and SSA-Orientation Features for Subset 4.

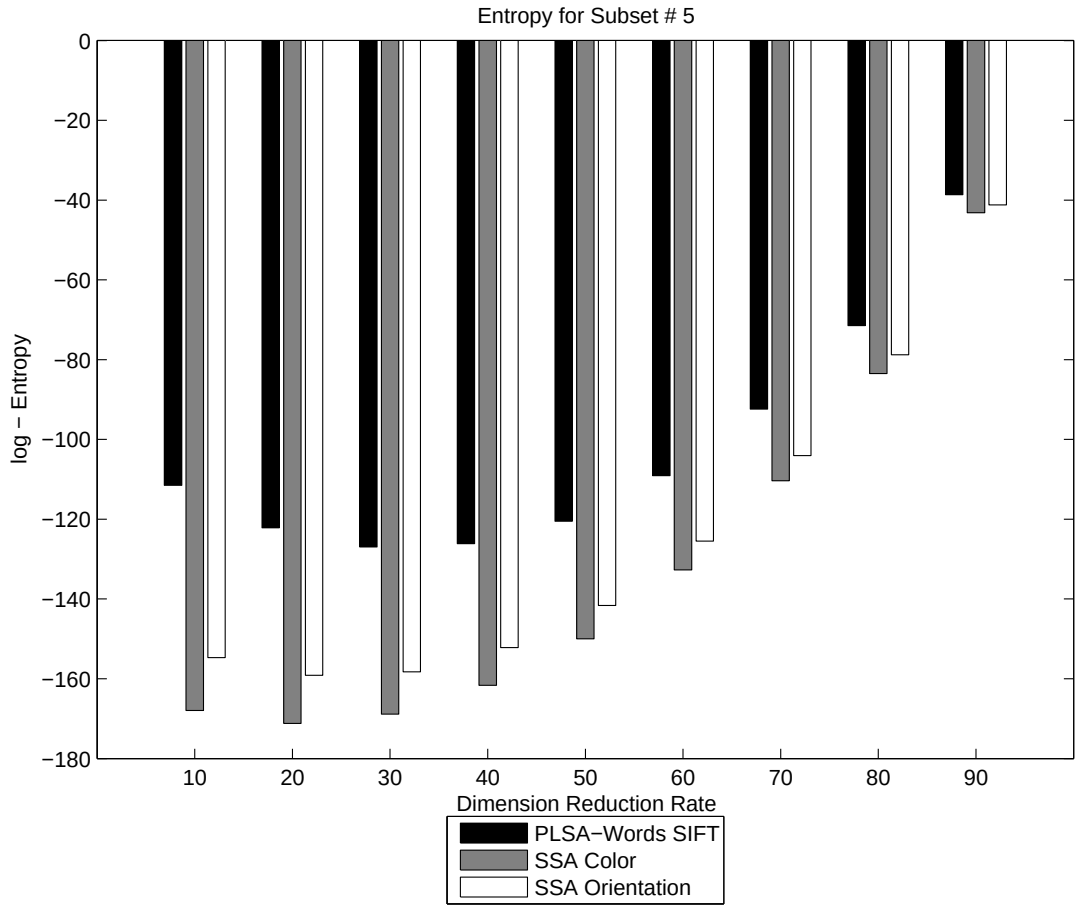

Figure B.4: Log entropy of SIFT, SSA-Color and SSA-Orientation Features for Subset 5.

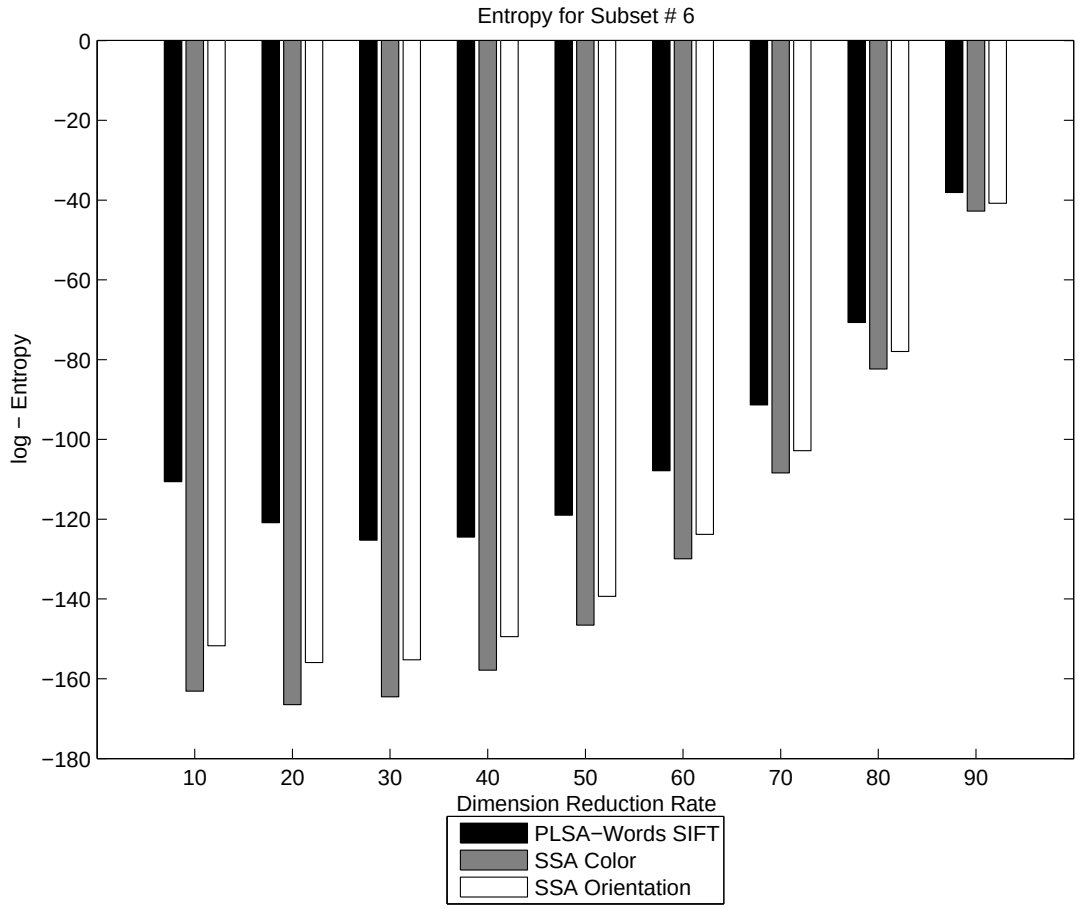

Figure B.5: Log entropy of SIFT, SSA-Color and SSA-Orientation Features for Subset 6.

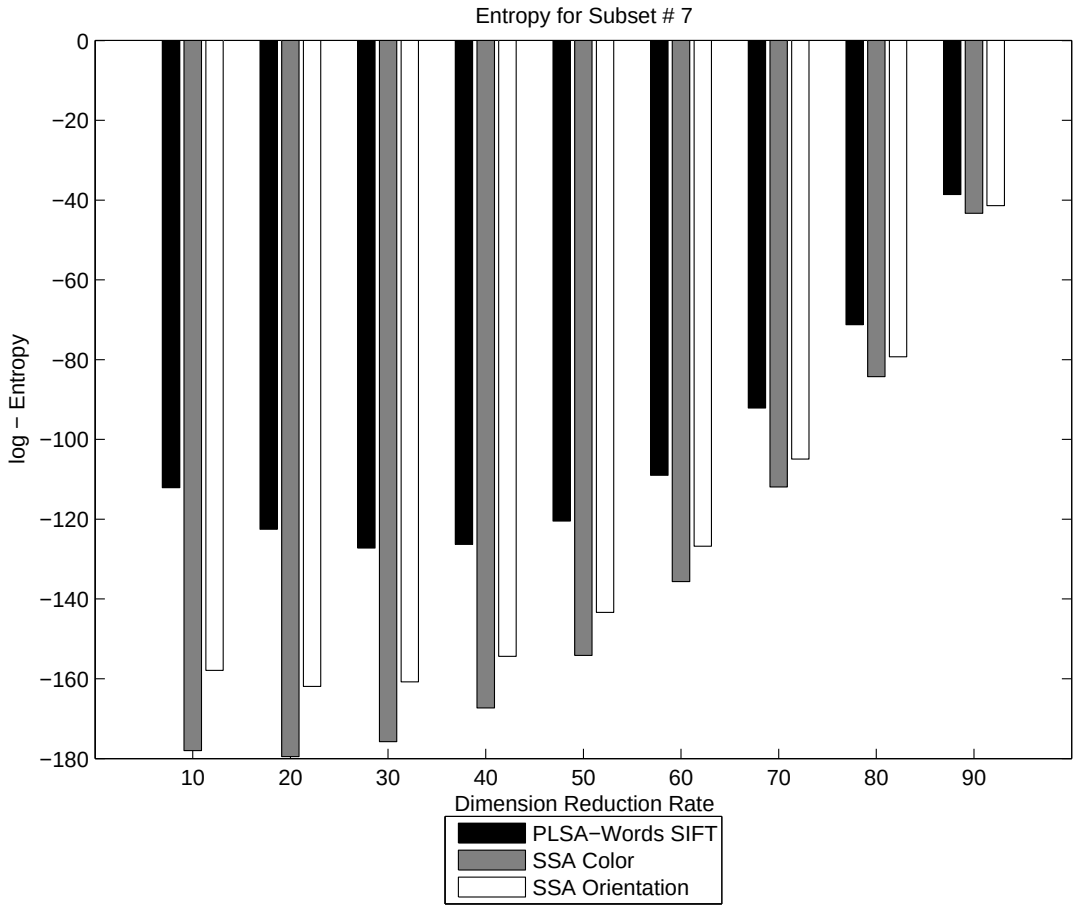

Figure B.6: Log entropy of SIFT, SSA-Color and SSA-Orientation Features for Subset 7.

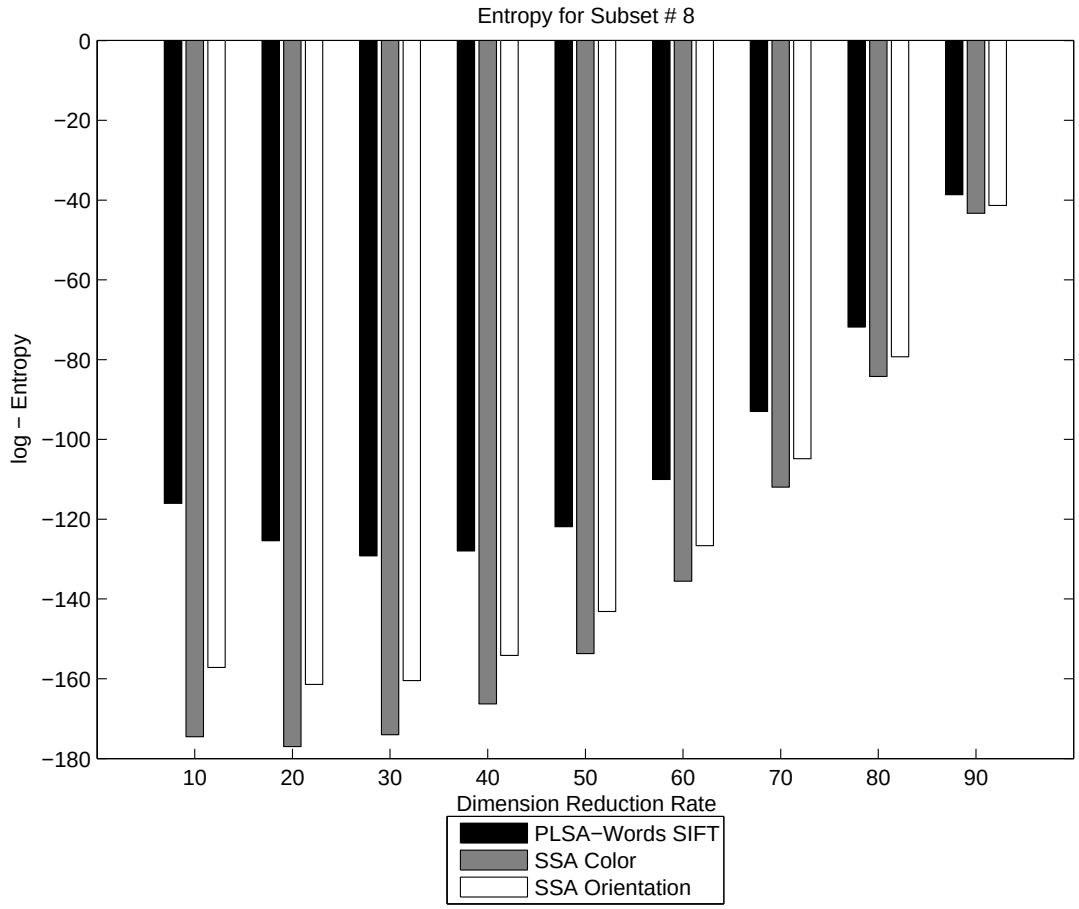

Figure B.7: Log entropy of SIFT, SSA-Color and SSA-Orientation Features for Subset 8.

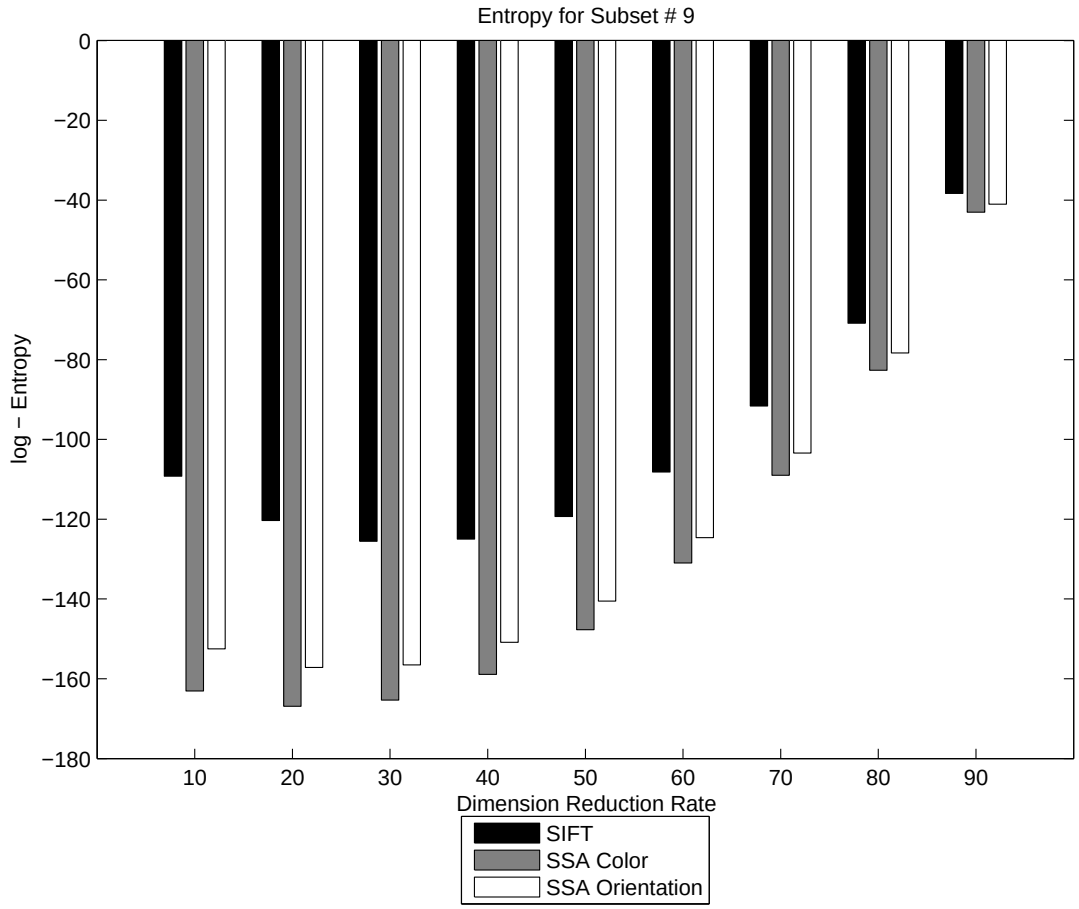

Figure B.8: Log entropy of SIFT, SSA-Color and SSA-Orientation Features for Subset 9.

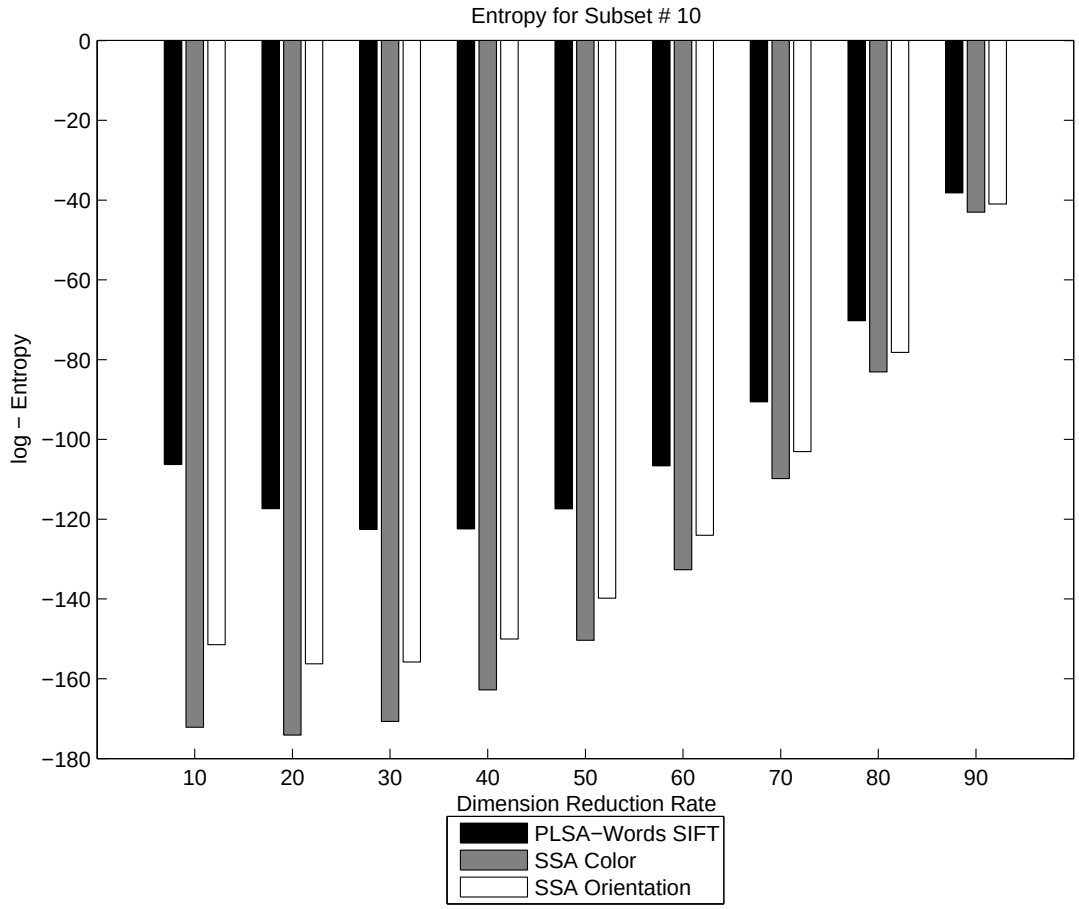

Figure B.9: Log entropy of SIFT, SSA-Color and SSA-Orientation Features for Subset 10.

VITA

Ahmet Sayar received his B.S. degree in Computer Engineering from the Middle East Technical University in 1987. He received his M.S. degree from the Department of Computer Science, University of Maryland in 1990. He was teaching/research assistant at the University of Maryland between 1987 and 1990. He worked as a software developer between 1990 and 1995 in the U.S.A. for AINS, a software development company. He was a consultant to United Nations in Turkey from 1996 to 1999. In 2000, he was the founding partner of Sayartek, a software company. He has been working for the Scientific and Technological Research Council of Turkey, Space Technologies Research Institute as a chief researcher since 2001.