Method of Adding Color Information to Spatially-Enhanced, Bag-of-Visual-Words Models

Robert Laurenson

A thesis

submitted in partial fulfillment of the

requirements for the degree of

Master of Science

University of Washington

2021

Committee:

Dr. Clark Olson

Dr. William Erdly

Dr. Min Chen

Program Authorized to Offer Degree:

Computing and Software Systems

University of Washington

Abstract

Method of Adding Color Information to Spatially-Enhanced, Bag-of-Visual-Words Models

Robert Laurenson

Chair of the Supervisory Committee:

Dr. Clark Olson

Computing and Software Systems

This thesis provides a late-fusing method, based on the HoNC (Histogram of Normalized Colors) descriptor, for combining color with shape in spatially-enhanced-BOVW models to improve predictive accuracy for image classification. The HoNC descriptor is a pure color descriptor that has several useful properties, including the ability to differentiate achromatic colors (e.g., white, grey, black), which are prevalent in real-world images, and to provide illumination intensity invariance. The method is distinguishable from prior late-fusing methods that utilize alternative descriptors, e.g., hue and color names descriptors, that are lacking with respect to one or both of these properties. The method is shown to boost the predictive accuracy by between about 1.9% - 3.2% for three different spatially-enhanced BOVW model types, selected for their suitability for real-time use cases, when tested against two datasets (i.e., Caltech101, Caltech256), across a range of vocabulary sizes. The method adds between about 150 – 190 mS to the models' total inference time.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENTS

Heartfelt thanks to Dr. Clark Olson, my thesis adviser, for his valuable guidance and numerous suggestions throughout the entire thesis process, which greatly improved the resultant work product.

DEDICATION

This thesis is dedicated to my wife Florence who provided unwavering support over the last five years as I transitioned my career into the interesting field of computer science.

Chapter 1. INTRODUCTION

The Bag-of-Visual-Words (BOVW) model is one of the most successful approaches for image classification available [1], [2], [3]. Despite the emergence of other machine learning models such as convolutional neural networks (CNN), the BOVW model remains relevant for real-time applications, such as traffic sign recognition and home intruder detection, where there is a need to generate inferences in real time. For these applications, spatially-enhanced BOVW models [5] provide performance that is competitive with, if not superior to, that provided by state-of-the art CNN models [6], [7].

In Abdi et al. [5], for example, a real-time traffic sign detector based on a spatially-enhanced BOVW model is reported as having a 90.48% recognition accuracy and 100 mS inference time when tested against the GTSRB dataset of German traffic sign images. This compares favorably with, if not exceeds, the performance of state-of-the-art CNN-based real-time object detectors such as, for example, R-FCN, currently ranked #2 on the Facebook AI leaderboard for real-time object detection using the PASCAL VOC 2007 dataset.[1] R-FCN is reported in Dai et al. [6] as having a mAP (mean average precision) of 80.5% and 9 FPS throughput (equivalent to a 111.11 mS inference time) when tested against the PASCAL VOC 2007 dataset.

Specific use cases for spatially-enhanced BOVW models include 1) the image classification stage of real-time object detection, which, according to the Facebook AI leaderboard website, is "the task of doing object detection in real-time with fast inference while maintaining a base level of accuracy"; and 2) machine learning applications where, due to privacy considerations or the like, there is a need or desire to train the machine learning model on a resource-constrained

[1] https://paperswithcode.com/task/real-time-object-detection

device, such as might be maintained in a home or at the edge of a cloud network, to avoid transmitting sensitive data or images into the cloud interior where the dissemination thereof is harder to control.

As illustrated in Figure 1.1, the traditional BOVW model represents an image with a histogram encoding the frequency of occurrence of quantized object features called visual words or codewords [4], [8]. To generate the histogram, the image is sampled at each of the intersection points of a dense grid overlaid (conceptually) over the image, and a descriptor representing the shape or intensity gradient of the image at the image patch located at that sample point is generated. Each descriptor is quantized by mapping it to the closest one of the visual words, and then the histogram bin corresponding to that visual word is incremented. This process is repeated at each of the sample points. The result is a histogram representing the image, as depicted in Figure 1.1.

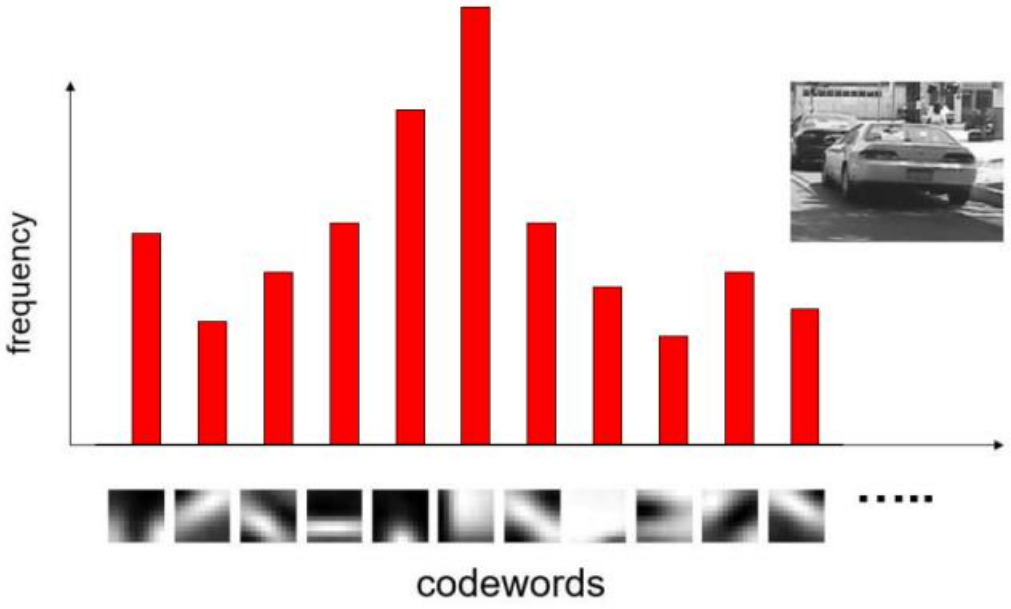

Figure 1.1. Traditional BOVW model[2].

[2] https://towardsdatascience.com/bag-of-visual-words-in-a-nutshell-9ceea97ce0fb

Quantization is required since all object features cannot be represented in the histogram, only a finite number *N*. The set of *N* visual codewords is referred to as the model's visual word vocabulary or codebook. (Hereinafter, either *N* or *m* will be used to refer to vocabulary size, depending on the context.) As illustrated in Figure 1.2, the process of quantizing descriptors of object features into visual codewords introduces quantization error which can detract from model performance.

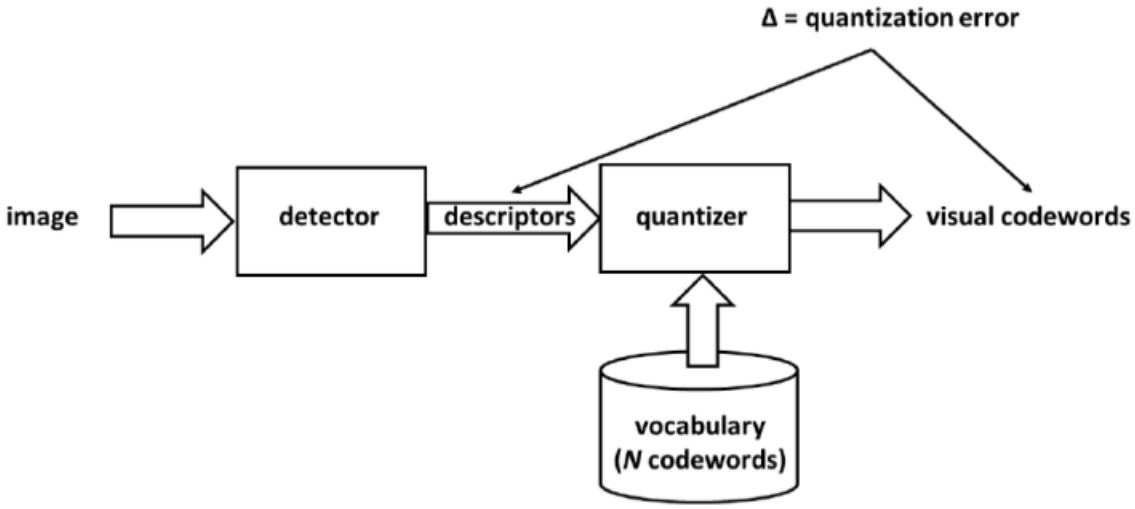

Figure 1.2. Quantization into codewords.

A significant drawback of traditional BOVW models is that they lack spatial information depicting the spatial distribution of object features in the image plane [1], [4]. The lack of spatial information limits the predictive accuracy provided by such models for image classification.

For example, as can be seen in Figure 1.3, where the red squares represent SIFT descriptors that map to the same visual word across four images from the Caltech101 dataset, without spatial information, the helicopter is indistinguishable from either of the two motorbike images as all have the same number (three) of instances of the visual codeword.

However, when spatial information is included, then the helicopter image is distinguishable from both motorbike images as the spatial distribution of the codewords in the helicopter image is quite different from that in either of the two motorbike images.

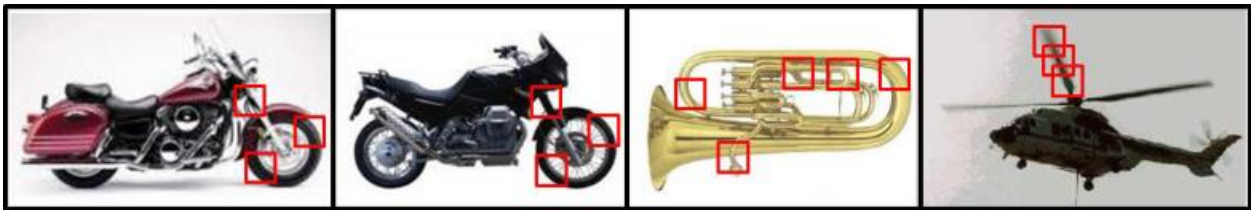

Figure 1.3. Discriminative power of identical object features (reproduced from [2]).

To provide this advantage, spatially-enhanced BOVW models were introduced, e.g., [2], [4], [5], [9]. Spatially-enhanced BOVW models add spatial information to the frequency information encoded by the traditional BOVW histogram for the purpose of improving predictive accuracy.

Examples of spatially-enhanced BOVW models include the multi-level spatial-pyramiding representation [9], where the spatial information is in the form of absolute locations of object features at multiple levels of granularity; the "single-level" variant of the spatial-pyramiding representation [9], where the spatial information is in the form of absolute locations of object features at a single level of granularity; and spatially-enhanced BOVW models [2], [4] where the spatial information is in the form of angle and/or distance histograms depicting the relative locations of object features as determined from pairwise comparisons of feature descriptors.

The inclusion of color offers yet another possibility for improving the predictive accuracy of spatially-enhanced BOVW models, which traditionally have relied on shape alone [8], [10], [14]. To this end, both early- and late-fusing methods have come to fruition for joining color with shape in BOVW and related models [1], [8], [10]. With early-fusing, color and shape descriptors are fused to form combination descriptors that are thereafter quantized and encoded into a

histogram, while, with late-fusing, the color and shape descriptors are separately quantized and encoded into distinct histograms, which are thereafter fused.

However, these methods have only incrementally and inconsistently been applied to spatially-enhanced BOVW models, perhaps because there is a mistaken perception that only spatial information relating to the spatial distribution of shape features adds discriminative power. For example, Abdi et al. [5] describe a spatially-enhanced BOVW model for use in real-time traffic sign recognition that renders image classification decisions based solely on shape information.

Elfiky et al. [1] report success from using an implementation of the late-fusing method to join color with shape in a specialized (compact) variant of the multi-level spatial pyramiding representation. However, Elfiky et al. [1] question whether this implementation is generalizable to other model types. Moreover, Khan et al. [4], published subsequent to Elfiky et al. [1], still identify the need for additional research regarding how to best incorporate "spatial information provided by multiple cues e.g. color and shape" into spatially-enhanced BOVW models.

In light of the foregoing, there is a need for a method of using color to improve the predictive accuracy of spatially-enhanced BOVW models for image classification that is generalizable to many if not all of the models in this class. A primary application of the method is the image classification phase of object detection [10].

This work is of interest to both practitioners and students of machine learning and computer vision, particularly those interested in exploring machine learning models for real-time use cases for which a convolutional neural network may not be optimal, or those who are interested in an image classification model with a high degree of transparency and optimization opportunities.

Chapter 2. LITERATURE REVIEW

Color has been shown to yield excellent results when combined with shape for image classification [10], [12]. Van de Weijer and Khan [8] compare and contrast early- and late-fusing methods for combining color and shape in traditional BOVW models.

Elfiky et al. [1] report that, for a specialized (compact) variant of the multi-level spatial pyramid model, late-fusing methods for combining color and shape outperform early-fusing methods. Khan et al. [2], [4] report that it is still an open research question regarding how to best use color and shape to improve performance of other types of spatially-enhanced BOVW models.

With respect to color descriptors, Khan et al. [10] recommend the use of a color descriptor that explicitly represents color rather than merely projecting shape information onto separate color channels. Kahn et al. [10] also report good results from using a particular color descriptor, the CN (color names) descriptor, for object detection compared to a hue descriptor. Elfiky et al. [1] similarly report use of the CN descriptor in their method. With a CN descriptor, RGB values are mapped to real-world color names using a mapping that is learned from Internet search results [13], rather than the specific dataset at hand.

Mansoori et al. [11] report good results from using a hue descriptor in a late-fusing method. Van de Weijer and Schmid [14] as well recommend the use of a hue descriptor in the case when the dataset colors are saturated.

Kopf et al. [15] describe an approach for enhancing the traditional BOVW model that does not explicitly use a hue or color descriptor. In this approach, an image is divided into sub-groups, each associated with a unique range of pixel hues, using hue-based masking, whereby a mask associated with a range of hues is used to filter out all but the pixels whose hue falls within the

range. A BOVW shape histogram is formed for each sub-group, and then all these histograms are concatenated to form the final histogram for the image.

Olson et al. [16] and Olson and Zhang [17] report excellent results from using a particular color descriptor (HoNC) for keypoint recognition. HoNC is distinguished from other color-related descriptors, such as RGB-SIFT and OpponentSIFT, on the basis that these latter descriptors merely provide shape information, in the form of intensity gradients, for individual color channels but not color information per se.

Olson and Zhang [17] further report that, while HoNC is invariant to illumination intensity, it is not invariant to illumination color, which improves its discriminatory power. Also, in the context of keypoint recognition, the concatenation of HoNC and SIFT descriptors is identified as a top-performing combination descriptor that captures both color and shape information.

Other more general or less salient references describe aspects of combining color with shape and/or texture for purposes of content-based image retrieval, see [18], [19], [20], [21], or for purposes of image classification, see [3], [22], [23], [24].

The work described herein builds upon the foregoing work to develop a novel late-fusing method, featuring the HoNC descriptor, for adding color to spatially-enhanced BOVW models to improve image classification performance.

The use of the HoNC descriptor distinguishes the method from prior late-fusing methods [1], [11] that employ a hue and CN descriptor, respectively. Compared to a hue descriptor, as detailed in Section 5.3, the HoNC descriptor better handles achromatic colors, e.g., white, grey, and black, which are prevalent in real-world images. Compared to a CN descriptor, the HoNC descriptor includes a formal mechanism, color normalization, for achieving illumination intensity invariance, while the CN descriptor lacks such a formal mechanism [28].

Through testing, we confirmed the method boosts the average-per-class predictive accuracy by between about 1.9 – 3.2% in three diverse spatially-enhanced BOVW model types over a range of vocabulary sizes, and also adds no more than about 150 – 190 mS to the models' total inference time. In many cases, that increase should be low enough to keep the total inference time with the range currently posted on the Facebook AI leaderboard for real-time object detection.[3] Based on the foregoing, we believe the method is generalizable to a broad range of model types, datasets and operating scenarios.

The method employs a late-fusion approach, which was found to outperform early-fusing in the context of our testing environment. We provide a theoretical explanation based on quantization error analysis to explain why the late-fusing approach outperforms early-fusing in this context.

Chapter 3. METHOD

For our method, rather than rely on color alone, we elected to join color with shape. We followed that approach because preliminary testing, summarized in Table 3.1, showed that a "single-level" spatial pyramiding model, utilizing only SIFT (shape) descriptors, out-performed that same model when configured to utilize only color-related descriptors (e.g., OpponentSIFT, RGB-SIFT, HoNC, HoWH).

For our shape descriptor, we used the SIFT descriptor given its excellent performance and invariance properties, as reported in Van de Weijer and Schmid [14]. The major decision points for our method were: 1) whether to use early- or late-fusing; and 2) which color descriptor to use.

[3] https://paperswithcode.com/task/real-time-object-detection

Table 3.1. Performance of color descriptors compared to SIFT.

Descriptor	Pred. Acc.
SIFT*	53.40%
OpponentSIFT**	53.22%
RGB-SIFT**	52.28%
HoNC**	36.19%
HoWH**	19.16%

*See Table 3.2
**Based on a single run@*N*=200

3.1 EARLY VS. LATE FUSING

In early fusing, as depicted at the top of Figure 3.1, prior to quantization, color and shape descriptors for a particular image patch are concatenated, optionally after applying relative weights given by β and (1 – β), respectively, to the color and shape descriptors. The combination descriptors are each quantized into a visual word using a combined shape/color vocabulary. The visual words are then encoded into a histogram that forms the basis for an image classification prediction for the image.

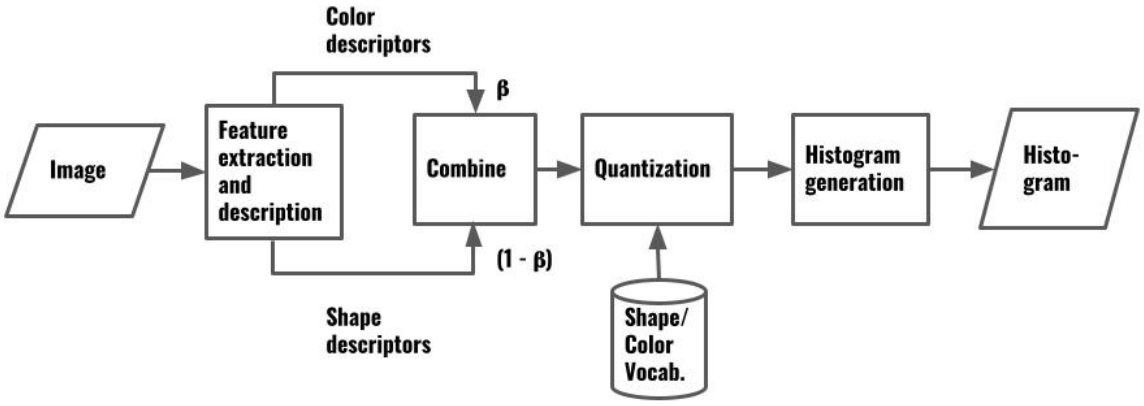

Figure 3.1. Early-fusing approach.

With late fusing, depicted in Figure 3.2, color and shape descriptors for a particular image patch are separately quantized into distinct color and shape visual words using different color and shape vocabularies, respectively. The color and shape visual words are separately encoded into corresponding histograms, which are then concatenated, optionally after applying relative weights given by α and (1 – α), respectively, to the color and shape histograms. The combination histogram forms the basis for an image classification prediction for the image.

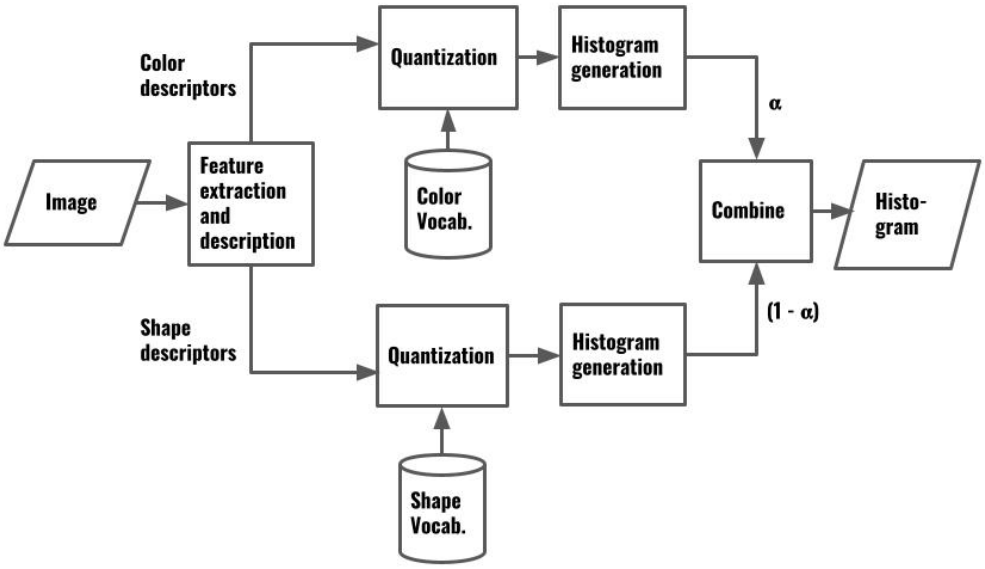

Figure 3.2. Late-fusing approach.

In choosing between early and late fusing for our method, we balanced design simplification considerations with performance. We found that, while early fusing results is a simpler, more efficient and more scalable design because it requires only a single vocabulary and single processing path rather than two as with late fusing, late fusing significantly outperforms early fusing due to the large quantization error that is introduced with early fusing, as explained in Section 3.2. Accordingly, we adopted late fusing for our method despite the additional complexity involved.

3.2 Quantization Error Considerations

We found that the introduction of color through early fusing to a shape-only model, while keeping the vocabulary size constant, significantly increases quantization error, as measured by mean squared error (MSE), and that this increase in quantization error is accompanied by a reduction in the model's predictive accuracy. For example, for a vocabulary size of 200, a "single-level" spatial pyramiding model based solely on the SIFT descriptor was found to have an MSE of 78,579, and a predictive accuracy of 52.00%. However, when early fusing of HoNC and SIFT descriptors was introduced to that model while keeping the vocabulary size constant at 200, the MSE increased to 140,591, and the predictive accuracy dropped from 52.00% to 47.01%, almost a 5% reduction. We conjecture from these results that the increase in quantization error through early fusing introduces error in the final histogram used to represent the image, which in turn reduces the predictive accuracy of the model.

We further found this increase in quantization error could not be mitigated by modest increases in the vocabulary size. For example, when we increased the vocabulary size from 200 to 800 in the previous example, the MSE dropped to only 121,878, far short of the original level of 78,579. In fact, using parametric modeling [25] of the inverse relationship (represented by equation 3.1) that has been determined to exist between the vocabulary size m and the MSE at that vocabulary size, W_m, we estimate that the vocabulary size in this example would have to be increased from 200 to roughly 55,000 in order to reduce the MSE to the original level of 78,579.

$$W_m \times m^{2/a} = Constant \tag{3.1}$$

As this increase would result in an infeasible histogram length, we investigated whether L2 normalization of the underlying descriptors could help lower the required increase in

vocabulary size. We found that, while L2 normalization did in fact lower the required increase in vocabulary size from 55,000 to 16,000, the increase in vocabulary size was still too large.

We next investigated the use of portmanteau vocabularies [8] [27] as a potential solution. With portmanteau vocabularies, as detailed in Khan et al. [27], individual color and shape vocabularies are first constructed in the traditional manner using K-means clustering. Then, the joint probability distribution of the multiple cues is empirically estimated assuming conditional independence between the multiple cues. Next, using the empirically-determined joint probability distribution, a compact portmanteau vocabulary is formed by applying a DITC compression algorithm to an expanded vocabulary that includes an entry for every possible combination of the two individual vocabularies.

Using this approach, it was reported that, for a multi-cue scenario involving a color vocabulary of 100 words and a shape vocabulary of 5,000 words, a portmanteau vocabulary was constructed that included only 100 multi-cue words, far less than the 500,000 words implied by enumerating every possible combination of the two individual vocabularies [27].

However, we ultimately rejected this approach because of its complexity, but also because the performance gains that were reported, 2.8% and 5.4%, did not appear sufficient to overcome the almost 5% deficit in performance we encountered due to early fusing of SIFT and HoNC descriptors, and add a reasonable boost.

For all the foregoing reasons, we concluded that, in our particular design, an early-fusing approach for joining the HoNC and SIFT descriptors was not tractable.[4] Accordingly, we opted for a late-fusing approach for joining color with shape.

[4] Although, in our case, quantization error was the dominant consideration in choosing late fusing over early fusing, in other cases, additional factors might come into play that favor one or the other approach [1], [8], [10].

3.3 Choice of Color Descriptor

In choosing a color descriptor for our method, we narrowed our selection down to the HoNC descriptor, given the successful use of that descriptor for keypoint matching as reported in Olson et al. [16], Olson and Zhang [17]; the HoWH descriptor, given the use of hue descriptors such as HoWH for content-based image retrieval as reported in Mansoori et al. [11]; and the CN descriptor, given the use of that descriptor for image classification as reported in Elfiky et al. [1], Khan et al. [10]. Of these three, we ultimately selected HoNC because it seemed to do the best job at balancing illumination intensity invariance and ability to distinguish achromatic colors, two desirable properties for a color descriptor that, according to Van de Weijer and Schmid [28], are often at cross-purposes.

More specifically, while HoWH, like HoNC, utilizes a normalization procedure to achieve illumination intensity invariance, HoWH weighs hue with saturation, to achieve robustness with respect to achromatic colors, which have an undefined hue [14]. However, this also causes achromatic colors to be ignored because these colors have low saturation [14]. Since achromatic colors are prevalent in real-world images, accounting for up to 44% of all pixels according to some estimates [28], we were concerned that selection of the HoWH descriptor would unduly limit the generalizability of our method. In the HSV color model shown in Figure 3.3, the achromatic colors appear along the value dimension of the model that runs vertically from the bottom to the top of the figure. They are to be distinguished from the chromatic colors, which appear around the periphery of the color wheel at the top of the figure.

Similarly, while the CN descriptor is able to distinguish between achromatic colors, it does so at the expense of achieving illumination intensity invariance [28]. No such tradeoff is required by the HoNC descriptor.

Another concern that weighed against the CN descriptor is that it utilizes a universal mapping, learned from Internet search results, to map RGB values into pre-selected categories of generic color names [10], [13], [28]. We were concerned that this universal mapping, not tied to the specific dataset at issue, could introduce error and subjectivity into the image classification process.

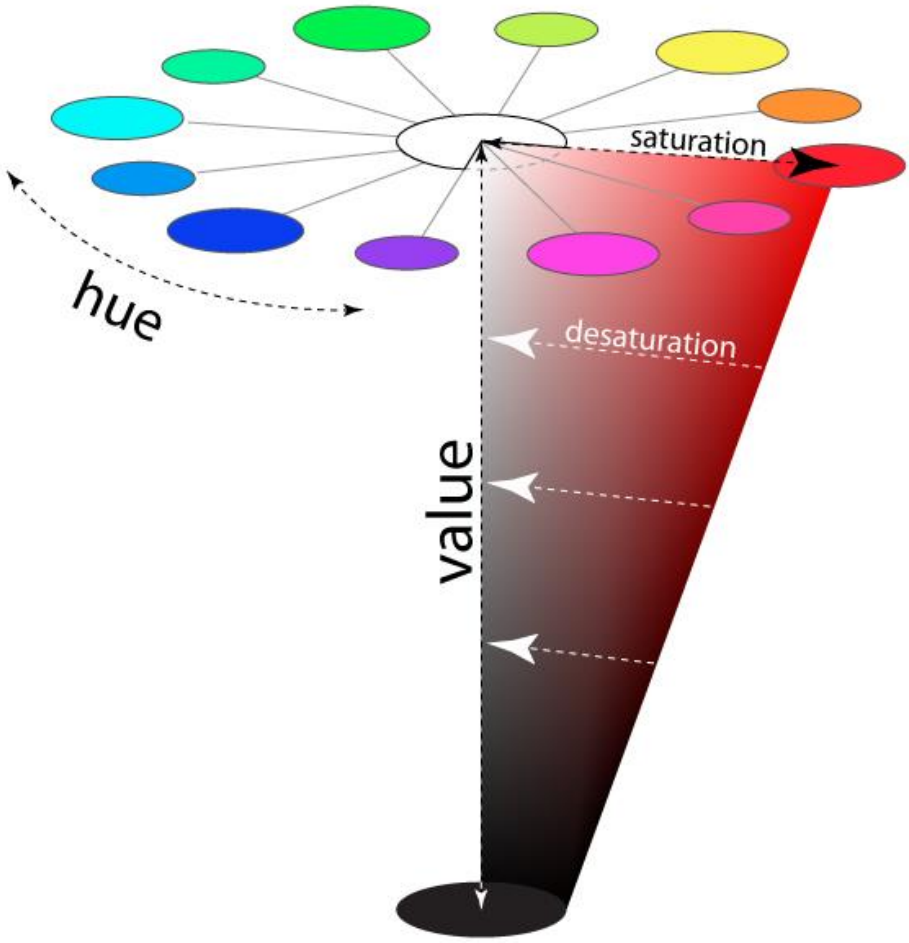

Figure 3.3. HSV color model[5].

[5] See http://learn.leighcotnoir.com/artspeak/elements-color/hue-value-saturation/

3.4 FINAL DESIGN

The final design has a training stage and a testing stage. A block diagram of the training stage is shown in Figure 3.4, and a block diagram of the testing stage is shown in Figure 3.5. Both stages mirror the late-fusing approach.

In the training stage, a training set forms the input, consisting of 30 randomly selected images from each class in the dataset. For each image in the training set, features are extracted and described through a two-step process. In the first step, keypoints are formed on a spaced, dense grid conceptually overlaying the image. In the second step, descriptors describing object features within image patches located at the keypoints are formed, with SIFT descriptors describing shape features, and HoNC descriptors describing color features.

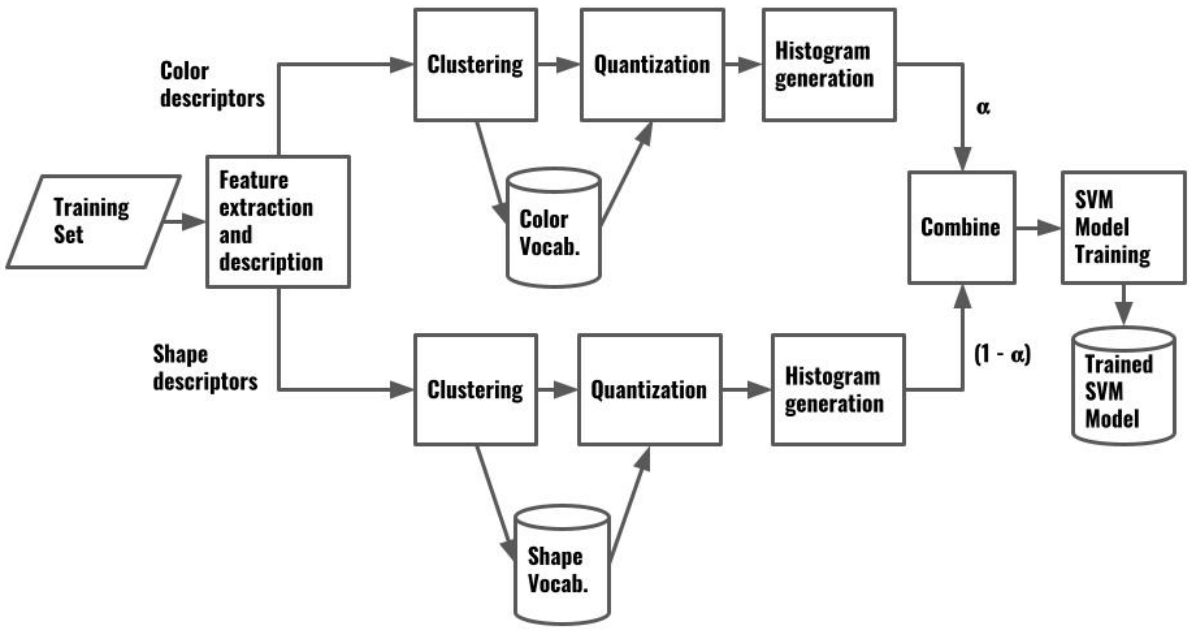

Figure 3.4. Block diagram of training stage.

Following first the upper path in the figure, K-means clustering is performed on the color descriptors for all the images in the training set. In this process, the descriptors are mapped into

N clusters in descriptor space, where *N* is a hyper-parameter set by the user. The centers of the resulting clusters form *N* color visual words, which become the color vocabulary. The color vocabulary is saved for later use during the testing stage. The color descriptors are then quantized by mapping each to the closest color visual word. The color visual words for each image are then encoded into a spatially enhanced color histogram using a spatially enhanced BOVW model.

Turning now to the lower path in the figure, similar to the color descriptors, K-means clustering is applied on the shape descriptors for all the images in the training set. In this process, the descriptors are mapped into *N* clusters in descriptor space, where *N* is the same hyper-parameter as before. The centers of the resulting clusters form *N* shape visual words, which become the shape vocabulary. The shape vocabulary is saved for later use during the testing stage. The shape descriptors are then quantized by mapping each to the closest visual word. The shape visual words for each image are then encoded into a spatially enhanced shape histogram using the same spatially enhanced BOVW model used for the color descriptors.

The color and shape histograms for each image are then weighted, respectively, with the values α and $(1 - \alpha)$, which are values set by the user, and the weighted histograms then concatenated to form the histogram for the image. (In our final design based on the HoNC color descriptor, we set α to 0.20 after determining through testing that value optimized performance compared to alternative settings of 0.40 and 0.10). The histograms for each of the images in the training set, accompanied by corresponding image labels, are then used to train the SVM machine learning model. The trained SVM model is stored for use during the testing stage.

Turning to Figure 3.5, the testing set that forms the input to the testing stage consists of 50 randomly selected images from each class that are independent of the training set. In the event

that there are less than 50 such images available for a particular class, then only the remaining, available images are included in the testing set for that class.

For each image in the testing set, feature extraction and description are performed through the same two-step process described above, i.e., dense keypoint generation followed by color and shape descriptor generation at the keypoint locations. The color descriptors follow the upper processing path shown in the figure, while the shape descriptors follow the lower path.

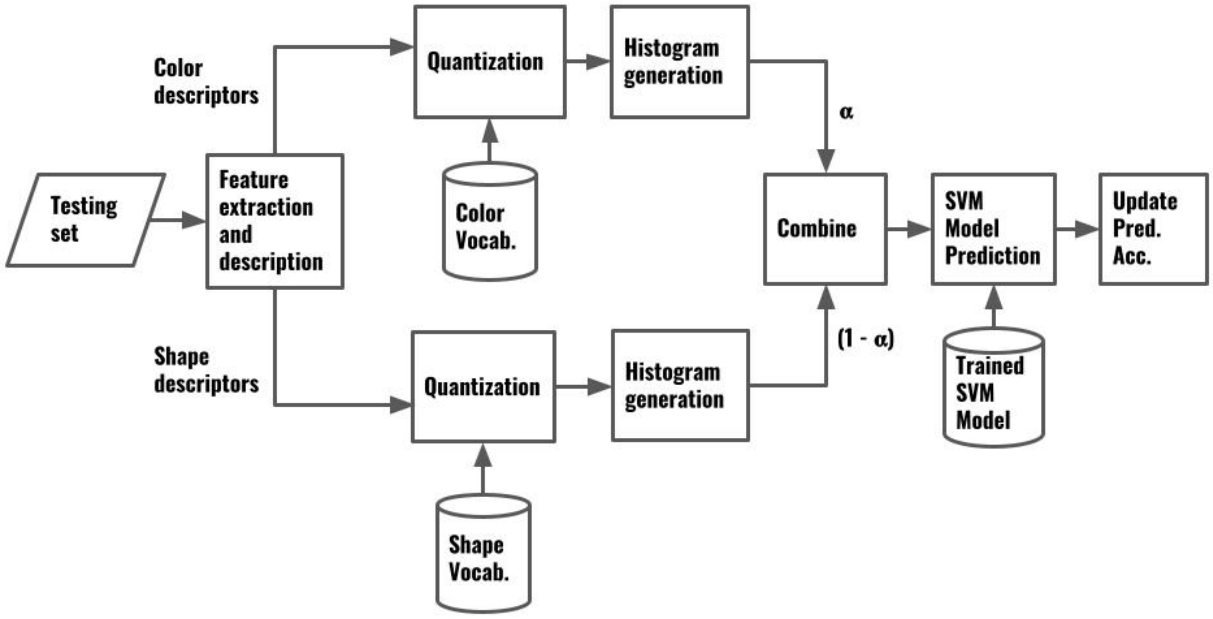

Figure 3.5. Block diagram of testing stage.

Beginning with the upper path, the color descriptors are quantized using the color vocabulary from the training stage by mapping each color descriptor to the closest color visual word. The color visual words for each image are then encoded into a spatially enhanced color histogram using the same spatially enhanced BOVW model used during the training stage.

Turning to the lower path, the shape descriptors are quantized using the shape vocabulary from the training stage by mapping each shape descriptor to the closest shape visual word. The

shape visual words for each image are then encoded into a spatially enhanced shape histogram using the same spatially enhanced BOVW model used above.

The color and shape histograms for each image are then weighted, respectively, with the values α and (1 – α), and the weighted histograms then concatenated to form the histogram for the image. The histogram for the image is then input to the trained SVM model, which generates a prediction of the image class. The predictions for each of the images are then compared to the corresponding class labels to determine which predictions are correct and which are not. This information is then used to determine the predictive accuracy for each class in the testing set.

3.5 Representative Examples of Spatially-Enhanced BOVW Models

In this section, we describe three representative examples of spatially-enhanced BOVW models with which to test our method. Our goal was to select three representative examples that would be appropriate for real-time use cases. In these descriptions, it is important to keep in mind the distinction between the image histogram, which represents the entire image, and a spatial histogram, which is a component of the image histogram associated with a particular codeword.

3.5.1 *Single-Level SPR*

Our first model is the "single-level" variant of the spatial pyramiding representation described in Lazebnik et al. [9], where only the pyramid base is represented not the entire pyramid. The image histogram generated for each image associates a single spatial histogram with each visual codeword in the vocabulary. An example of this spatial histogram, suitable for a 300 x 300-pixel image, is shown in Figure 3.6. The spatial histogram depicts the absolute X-Y locations of the descriptors that map to the corresponding codeword. While a single spatial histogram is shown

in Figure 3.6, it should be appreciated that, in an actual implementation, a spatial histogram will be associated with each codeword in the vocabulary.

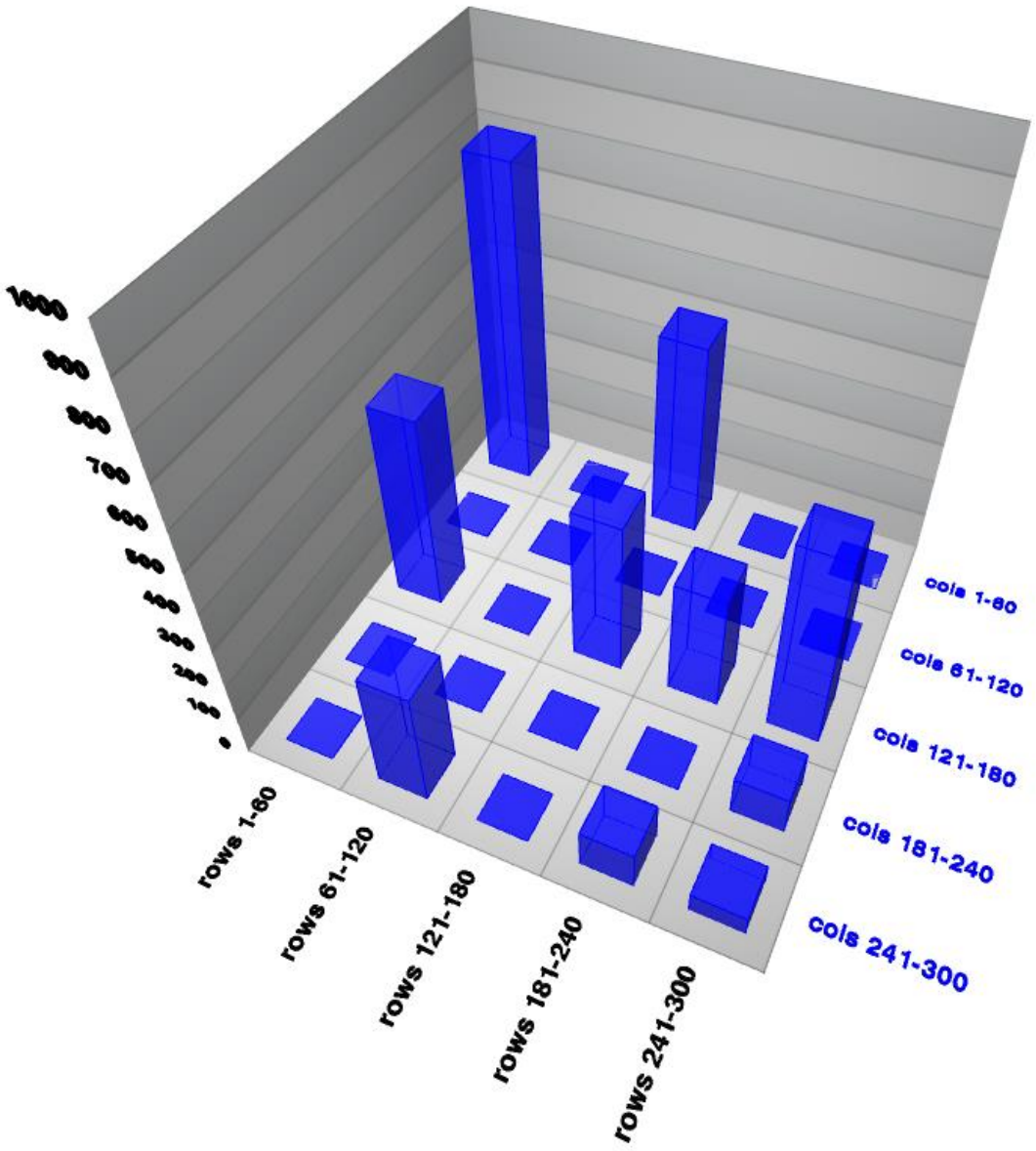

Figure 3.6. Example of spatial histogram for Single-Level SPR model.

As shown, the base of the spatial histogram is a 5 x 5 two-dimensional grid, where the horizontal, or X, dimension of the base grid represents rows of the image plane, and the vertical, or Y, dimension of the base grid represents columns of the image plane. Each cell of the grid

corresponds to a range of spatial (X-Y) locations on the image plane and the height of the spatial histogram at that cell indicates a count of the number of descriptors located at or within these spatial locations that map to the visual word corresponding to the spatial histogram.

To populate the image histogram, each descriptor for the image is quantized by mapping it to its closest visual word. The spatial histogram for that visual word is then located, and the frequency count of the cell corresponding to the X- and Y-locations of the descriptor on the image plane is incremented.

To determine the X-coordinate of that cell in the grid, the bucket size in the X-direction is determined by dividing the width of the image by 5, the width of the grid in terms of number of cells. The X-coordinate of the cell in the grid in then determined by taking the integer floor of the result of dividing the X-component of the location of the center of the keypoint for the descriptor by this bucket size.

Similarly, to determine the Y-coordinate of that cell in the grid, the bucket size in the Y-direction is determined by dividing the height of the image by 5, the height of the grid in terms of number of cells. The Y-coordinate of the cell in the grid is then determined by taking the integer floor of the result of dividing the Y-component of the location of the center of the keypoint for the descriptor by this bucket size.

This process is repeated for each of the descriptors corresponding to the image. The result is an image histogram representing the image that is infused with spatial information (the spatial histograms corresponding to each visual word in the vocabulary) encoding the absolute X, Y locations of the descriptors on the image plane.

The complexity of the algorithm for generating the histogram is $O(n)$, where n is the number of descriptors for the image. The dimensionality of the histogram is $25m$, where m is the

size of the vocabulary in terms of number of visual words. Hereinafter, this model type will be referred to as "Single-Level SPR" where SPR stands for "Spatial Pyramid Representation".

We selected this model type because of its performance relative to the traditional BOVW model, its computational efficiency, and also because of its unique design. Beginning with model performance, Table 3.2 shows the predictive accuracy of the Single-Level-SPR model relative to the traditional BOVW model. Although these test results do not reflect later optimizations that are reflected in the data reported in Chapter 5, they still provide valuable insight as to why this model type was selected.

Table 3.2. Performance of Single-Level SPR model.

Dataset	Vocab size	Single-Level SPR					
		Pred. Acc*		Hist gen time (mS)*		Boost	Dim.
Caltech101		μ	σ	μ	σ		
	100	51.13	1.04	9.35	0.06	**10.86**	2500
	200	53.40	1.51	10.65	0.04	**8.39**	5000
	400	54.77	1.11	13.62	0.41	**6.09**	10000
	1000	55.28	0.66	20.97	0.30	**3.84**	25000

*Computed over 10 runs

The bolded values in Table 3.2 show the spatial information included in the Single-Level-SPR model substantially boosts the predictive accuracy at each of the vocabulary sizes tested compared to the traditional BOVW model. Although the size of the boost drops as vocabulary size increases, the boost in performance at each level of vocabulary size is still substantial, which weighs in favor of including this model.

The second factor that weighs in favor of including this model is its computational efficiency, meaning that the spatially-enhanced model is capable of achieving any particular predictive accuracy at a reduced vocabulary size compared to the traditional BOVW model. This computational efficiency is implied by the positive boost values at each of the vocabulary sizes in

Table 3.2. If the spatial information responsible for these boosts were to be eliminated by reverting to a traditional BOVW model, the only way to maintain the predictive accuracy at the current level would be to increase the vocabulary size.

The third factor that weighs in favor of including this model is its unique design that differentiates it from the other models, namely, that it provides only a single spatial histogram for each codeword in the vocabulary. This factor facilitates the real-time performance of the model relative to that of the other models, which maintain more than one spatial histogram per codeword.

3.5.2 *Single-Level-SPR/HPS-Lite*

Our second model combines the foregoing Single-Level SPR representation with a modification of the HPS_{ad}^{intra} representation, described in Khan et al. [4], which will hereinafter be referred to as "HPS-Lite", where HPS stands for "Hard Pairwise Similarity."[6] The combination will be referred to as Single-Layer SPR/HPS-Lite.

As shown in Figure 3.7, the image histogram generated for each image associates two spatial histograms with each visual word in the vocabulary, the Single-Level SPR spatial histogram shown in blue and the HPS-Lite spatial histogram shown in red. The structure and manner of populating the Single-Level SPR spatial histogram was described in the previous section and will not be repeated here. Instead, we focus our attention on the HPS-Lite spatial histogram.

The HPS-Lite spatial histogram is generated from a pairwise comparison of all the descriptors that map to the visual codeword to which the spatial histogram corresponds. It depicts the relative locations (i.e., in terms of distance and angle) between these pairs of descriptors, and is intended to complement the information represented by the Single-Level SPR histogram, i.e.,

[6] The modification consists of the elimination of a hard Gaussian weighting step described in Khan et al. [4] that we considered unnecessary and to detract from real-time performance.

absolute X-Y locations of the descriptors that map to the visual codeword to which the spatial histogram corresponds.

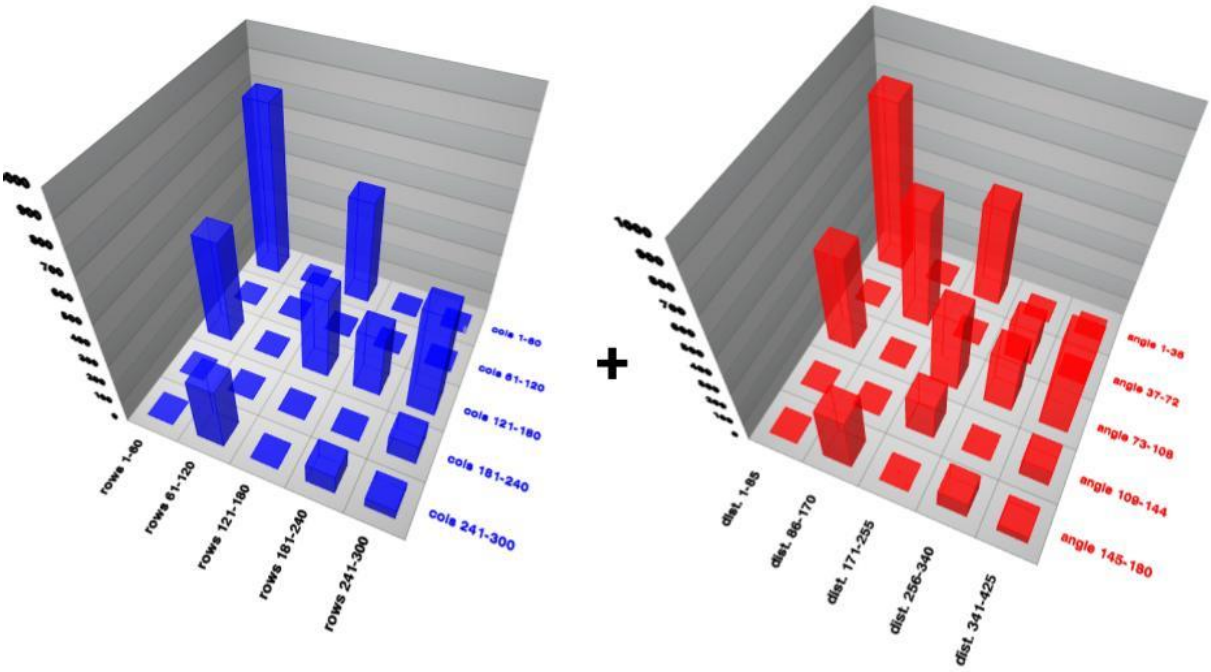

Figure 3.7. Concatenated spatial histogram for Single-Level-SPR/HPS-Lite model.

Turning to Figure 3.8, depicting an expanded view of the HPS-Lite spatial histogram shown in Figure 3.7, the base of the HPS-Lite spatial histogram consists of a 5 x 5 two-dimensional grid, where the horizontal (or X) dimension represents the distance between pairs of descriptors that both map to the visual word to which the spatial histogram corresponds, and the vertical (or Y) dimension represents the angle between such pairs of descriptors.

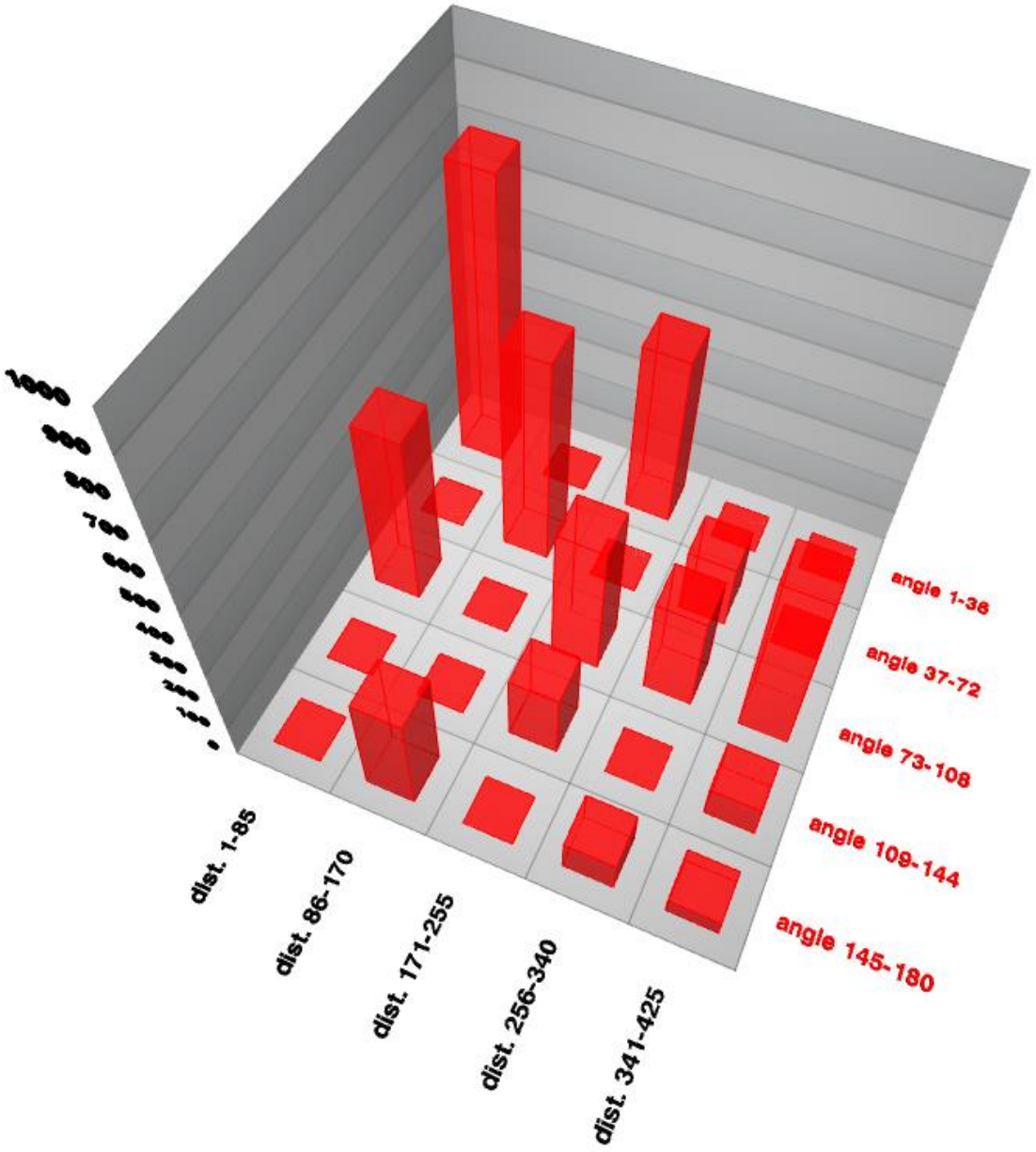

Figure 3.8. Expanded view of spatial histogram for HPS-Lite.

The process for populating such spatial histograms is depicted in Figure 3.9. As illustrated, it is assumed that each of the incoming descriptors have each been mapped to its closest visual

word, and then ordered. An outer loop and an inner loop are executed to compare each descriptor d_i in the ordered list with all descriptors d_j ahead of d_i in the ordered list (i.e., all d_j, where $j > i$).

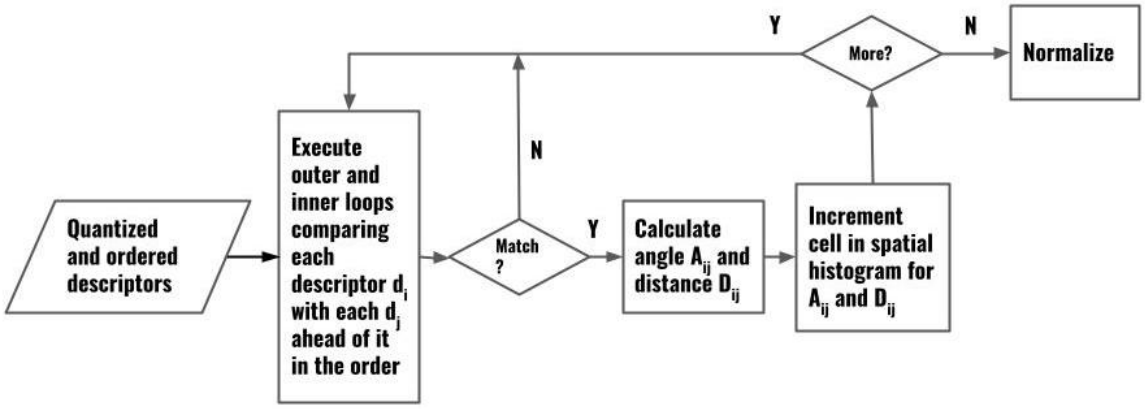

Figure 3.9. Process flow diagram for populating HPS-Lite spatial histogram.

If two such descriptors match, i.e., both map to the same visual word, the Euclidean distance D_{ij} between the pair in image space is calculated, as well as the angle A_{ij} between the pair relative to the horizontal. These distance and angle values are calculated by "constructing" a vector between the pair and then taking D_{ij} as the length of this vector, and A_{ij} as the angle of this vector relative to the horizontal. The HPS-Lite spatial histogram corresponding to the visual word to which the pair of descriptors match is retrieved, and then the count in the cell corresponding to these distance and angle values is incremented.

To determine the X-coordinate of that cell, the bucket size in the X-direction is determined by dividing the length of the image diagonal, which sets the maximum distance between any two descriptors, by 5, the width of the grid in terms of number of cells. The X-coordinate of the cell is then determined by taking the integer floor of the result of dividing the distance between the pair of descriptors D_{ij} by this bucket size.

To determine the Y-coordinate of that cell, the bucket size in the Y-direction is determined by dividing the maximum angle between any pair of descriptors, which is 180°, by 5, the height of the grid in terms of number of cells. The Y-coordinate of the cell is then determined by taking the integer floor of the result of dividing the angle between the pair of descriptors A_{ij} by this bucket size. Once the X- and Y- coordinates of the cell are determined, the count in this cell is incremented.

The specifics of the angle calculation are shown in Figure 3.10. In the figure, *V* denotes the vector that is constructed between the pair of descriptors d_i and d_j in the X-Y image plane. The left portion of the figure represents the situation where the dot product of *V* and i_y, the unit vector in the Y-direction, is greater than 0 (i.e., *V* is pointing downwards from d_i to d_j) and the right portion of the figure represents the situation where this dot product is less than or equal to 0 (i.e., *V* is horizontal or pointing upwards from d_j to d_i).

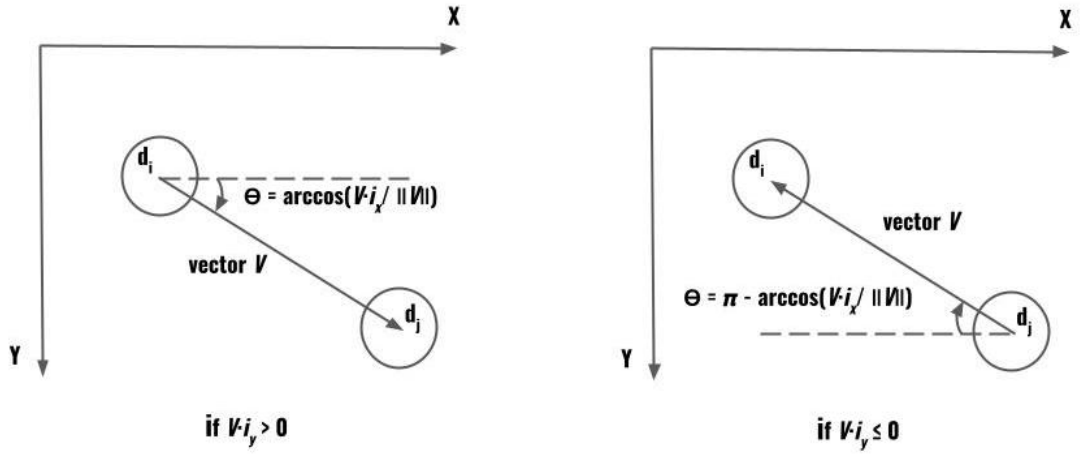

Figure 3.10. Specifics of angle calculation.

In the situation on the left, the angle Θ is given by the following equation, where i_x is the unit vector in the X-direction, $V \cdot i_x$ is the dot product between *V* and i_x, and $\|V\|$ is the magnitude of *V*:

$$\Theta = arccos\left({V \cdot i_x}/{\|V\|}\right) \tag{3.2}$$

In the situation on the right, the angle Θ is given by the following equation, where again $V \cdot i_x$ is the dot product between *V* and i_x, and $\|V\|$ is the magnitude of *V*:

$$\Theta = \pi - \ arccos\left({V \cdot i_x}/{\|V\|}\right) \tag{3.3}$$

These calculations ensure that the numerical value of the angle Θ will be invariant to whichever direction the vector *V* is pointing, from d_i to d_j or vice-versa.

After an HPS-Lite spatial histogram corresponding to a visual codeword has been populated, it is normalized, as described in Khan et al. [4], so that it is given equal weight compared to the corresponding Single-Level SPR spatial histogram. If b_i represents the number of descriptors that map to this visual word, normalization occurs by multiplying each count in the grid by the ratio of b_i, and $(b_i \times (b_i - 1))/2$, the number of pairs of those descriptors, which simplifies to $2/(b_i - 1)$.

Once normalized, the HPS-Lite spatial histogram is concatenated with the corresponding Single-Level SPR spatial histogram for the same visual codeword, after which L1 normalization of the concatenation is performed. The result is a concatenated spatial histogram for the corresponding visual word.

This normalization and concatenation procedure is repeated for each of the HPS-Lite spatial histograms. The result is an image histogram representing the image that is infused with spatial information encoding both the absolute and relative locations of the descriptors on the image plane.

The complexity of the algorithm for generating the image histogram is $O(n^2)$, where n is the number of descriptors for the image. The dimensionality of the final histogram is $50m$, where m is the vocabulary size in terms of number of visual words.

As with the Single-Level SPR model, we selected this model due to its performance relative to the traditional BOVW model, its computational efficiency, and its unique design. Beginning with performance, Table 3.3 shows the performance, in terms of predictive accuracy, of the Single-Level-SPR/HPS-Lite model relative to the traditional BOVW model. As with Table 3.2, although these test results do not reflect later optimizations that are reflected in the data reported in Chapter 5, they still provide valuable insight as to why this model was selected.

Table 3.3. Performance of Single-Level-SPR/HPS-Lite.

Dataset	Vocab size	Single-Level-SPR/HPS-Lite					
		Pred. Acc*		Hist gen time (mS)*		Boost	Dim.
Caltech101		μ	σ	μ	Σ		
	100	52.75	1.11	13.34	0.25	**11.68**	5000
	200	54.15	1.36	13.89	0.20	**9.14**	10000
	400	55.63	1.11	16.08	0.49	**6.95**	20000
	1000	56.03	0.66	22.98	0.56	**4.59**	50000

*Computed over 10 runs

The bolded figures show the Single-Level-SPR/HPS-Lite model substantially boosts the predictive accuracy compared to the traditional BOVW model at each of the vocabulary sizes tested, which weighs in favor of including this model.

In fact, as can be seen by comparing the bolded values of Table 3.3 with those of Table 3.2, the boost from the Single-Level SPR/HPS-Lite model exceeds that from the Single-Level SPR model by between about 0.75 – 1.62%, suggesting here is a synergistic effect resulting from the complimentary nature of the spatial information encoded by the two histograms that are concatenated in this model, consisting, respectively, of the absolute and relative locations of object

features on the image plane. The complimentary nature of this information, which is noted in Khan et al. [4], further weighs in favor of including this model.

Turning to computational efficiency, once again, compared to the traditional BOVW model, the Single-Level-SPR/HPS-Lite model is capable of achieving any particular predictive accuracy at a reduced vocabulary size compared to the traditional BOVW model. As with Table 3.2, this computational efficiency is implied by the positive boost values at each level of vocabulary size for the Single-Level SPR/HPS-Lite model.

The third factor that weighs in favor of including this model is its unique design that differentiate it from the other models, namely, that it provides two spatial histograms for each codeword in the vocabulary, one depicting the absolute locations in the image plane of the descriptors that map to this codeword, the other depicting the relative locations in the image plane of the descriptors that map to this codeword.

3.5.3 *Multi-Level SPR*

Our third model is the Multi-Level Spatial Pyramiding Representation of Lazebnik et al. [9], where, in our particular implementation, the parameter L, indicating the number of levels beyond the base level, is set to 2, for a total of 3 levels. At this setting, an image histogram is generated that associates each visual word in the vocabulary with three spatial histograms, a 1 x 1 spatial histogram, corresponding to level 0, a 2 x 2 spatial histogram, corresponding to level 1, and a 4 x 4 spatial histogram, corresponding to level 2. Through this association between visual words and spatial histograms, a total of 21 bins are allocated in the image histogram for each visual word.

An example of the three-level spatial pyramiding representation depicting this association between visual words and spatial histograms is shown in Figure 3.11. In this example, there are three visual words, indicated by circles, diamonds, and crosses. At the bottom left, three 1 x 1

spatial histograms are depicted corresponding, respectively, to the circle, diamond, and cross visual words. At the bottom center, three 2 x 2 spatial histograms are depicted corresponding, respectively, to the circle, diamond, and cross visual words. Lastly, at the bottom right, three 4 x 4 spatial histograms are depicted corresponding, respectively, to the circle, diamond, and cross visual words.

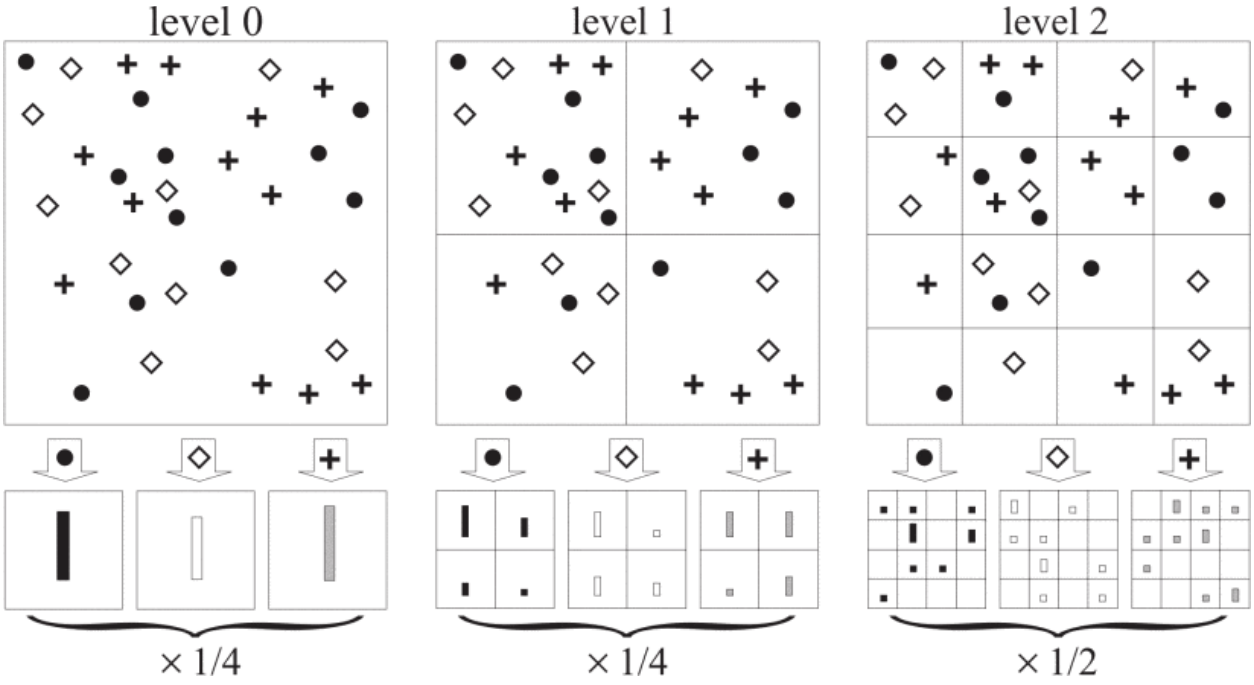

Figure 3.11. Example of a three-level spatial pyramid (reproduced from [9]).

To populate the image histogram, each descriptor from the image is quantized by mapping it to its closest visual word, and then all three of the spatial histograms corresponding to that visual word are located. Next, the appropriate cell in each of these spatial histograms is incremented.

In the case of the 1 x 1 spatial histogram, there is only one cell, so the count for that single cell is incremented. In the case of the 2 x 2 histogram, of the 4 possible cells, the cell corresponding to the (X, Y) location of the keypoint center for the descriptor in the image plane is incremented. In the case of the 4 x 4 spatial histogram, of the 16 possible cells, as with the 2 x 2 spatial histogram,

the cell corresponding to the (X, Y) location of the keypoint center for the descriptor in the image plane is incremented.

Each of the spatial histograms is also weighted based on their granularity. Since the 4 x 4 spatial histogram is the most granular, it is weighted by 1/2, while the 1 x 1 and 2 x 2 spatial histograms are both weighted by 1/4. This process is repeated for each of the descriptors for the image.

As a result of the foregoing, an image histogram representing the image is generated that is infused with the absolute locations of the descriptors in the image plane, at multiple levels of granularity. The complexity of the algorithm for generating this histogram is $O(n)$, where n is the number of descriptors for the image.

The dimensionality of the resulting histogram is $21m$, where m is the vocabulary size in terms of number of visual words. Hereinafter, this model type will be referred to as "Multi-Level SPR".

We included this model because it is the most notable paradigm in this area [4], and also because of its unique design that differentiates it from the other models, namely, that it provides three spatial histograms, at multiple levels of granularity, for each codeword in the vocabulary, depicting the absolute locations in the image plane of descriptors that map to this codeword. The multiple levels of granularity are reported to provide a statistically significant boost in performance compared to the Single-Level SPR model [9].

Chapter 4. EXPERIMENTS

In this section, we discuss the datasets used for testing, our platform, implementation details and our experimental protocol.

4.1 THE DATASETS

Two datasets were used for testing, Caltech101 and Caltech256. Caltech101 contains 9,146 images split between 102 image categories.[7] There are between 40 and 800 images per category; most categories have about 50 images. Caltech101 has been widely tested, [4], [9], and is used here to facilitate comparison with these earlier works. Caltech256 consists of 30,607 images distributed over 257 image categories.[8] The minimum number of images in any category is 80.

The complexity of each of these datasets, in terms of number of classes, is believed comparable with that of the two datasets, PASCAL VOC 2007 and COCO, that are currently being use for benchmarking by Facebook AI in the field of real-time object detection[9]. PASCAL VOC 2007 in particular has 20 classes, while COCO has 80 classes. We considered Caltech101 and Caltech256 to be better suited to the problem at hand, which is image classification not object detection per se.

4.2 PLATFORM

The hardware platform is a Windows 10 PC, with a multi-core Intel 1.6GHz processor, 12GHz of RAM, and a 64-bit operating system. Our software platform is based on an OpenCV 4.1.1 library including the extra modules for SIFT and SURF. To this library, we added HoNC, HoWH, RGB-SIFT, OpponentSIFT, SPIN, HoNI, CHoNI, and CSPIN descriptor code modules and several utility code modules, which we obtained from GitHub.

[7] http://www.vision.caltech.edu/Image_Datasets/Caltech101/
[8] http://www.vision.caltech.edu/Image_Datasets/Caltech256/
[9] https://paperswithcode.com/task/real-time-object-detection

For our software development environment, we used the Community version of Visual Studio 2019. In this development environment, we built our testing software in C++. We started with C++ code for testing the traditional BOVW model obtained from GitHub[10], and added the following functionality in order to facilitate testing of spatially-enhanced BOVW models, with and without color included:

- the three spatially-enhanced BOVW models outlined in Section 3.5
- a front end allowing a user to select model type and set hyper-parameters and/or operating parameters
- early- and late-fusing methods to join color and/or texture with shape in any of these models
- code that randomly selects independent training and test sets from the dataset at hand for a particular run, with 30 images in the test set for each category, and up to 50 images in the training set for each category
- code for accumulating statistics from a predetermined number, e.g., 10, of runs of a particular model
- code for implementing a dense detector, which was not available in OpenCV 4.1.1

4.3 Implementation Details

For Caltech101, following the experimental setup of Khan et al. [4] and Lazebnik et al. [9], we used a single-scale dense detector for sampling the image. As a dense detector is not native to

[10] https://github.com/hkhojasteh/Bag-of-Visual-Words

OpenCV 4.1.1, to implement the dense detector, we added the following C++ code to our testing software:

```
vector<KeyPoint> keypoints;
for (int y = STEP; y < test_img.rows - STEP; y += STEP) {
        for (int x = STEP; x < test_img.cols - STEP; x += STEP) {
                keypoints.push_back(KeyPoint(float(x), float(y), KP_SIZE));
        }
}
```

For Caltech101, we set STEP to 8 and KP_SIZE to 6, to effect sampling of an image every 8 pixels in the X- and Y- directions, using a keypoint size of 6 pixels, which, we understand, translates in OpenCV to a 24-pixel descriptor patch. This marks a difference with the experimental setup of Lazebnik et al. [9], which sampled over 16-pixel descriptor patches.

For training, we randomly selected 30 images per category. For testing, we selected all the remaining images for the category if there was less than 50, and otherwise randomly selected 50 images from the remainder for the category. We also followed the recommendation in Lazebnik et al. [9] of suspending the usual SIFT normalization procedure for weak patches.

We used K-means clustering to construct the vocabularies. For our machine learning prediction model, we used the multi-class SVM model with the intersection kernel selected and with the cost parameter *C* optimized using a grid search and 10-fold cross-validation on the training set.

For Caltech256, we used the same, single-scale dense detector as for Caltech101, but with the STEP parameter set to 12 pixels rather than 8, in order to keep the number of descriptors per image at about 1,050, which we experienced with Caltech101, but that was exceeded with Caltech256 using 8-pixel spacing. We avoided using the multi-scale dense descriptor recommended in Khan et al. [4] for Caltech256 because that would have required us to change our

platform from OpenCV to VLFeat. We found the single-scale dense detector was sufficient for us to achieve our objective of measuring the boost in performance from adding color when testing the Caltech256 dataset.

For training, as with Caltech101, we randomly selected 30 images per category, and, for testing, 50 images per category from the remaining images. We also followed the recommendation in Lazebnik et al. [9] of suspending the usual SIFT normalization procedure for weak patches.

As with Caltech101, we used K-means clustering to construct the vocabularies. For our machine learning prediction model, we again used the multi-class SVM model with the intersection kernel selected and with the cost parameter *C* optimized using a grid search and 10-fold cross-validation on the training set.

4.4 Experimental Protocol

For each dataset and model, over a range of vocabulary sizes, we measured the model's average-per-class predictive accuracy[11] without including color, measured the model's average-per-class predictive accuracy with color included through our late-fusing method, and then took the difference between these two measurements to determine the boost in the model's performance from including color.

For selected ones of the model configurations, i.e., those we considered to be top-performing, we also measured average-test-time-per-image without including color, measured average-test-time-per-image with color included using our late-fusing method, and then took the

[11] The alternative performance measurement, the percentage of all test images classified correctly, can give biased results if the size of test set varies considerably by size, as is the case with Caltech101 [9].

difference between these two measurements to determine the increase in the model's inference time from including color.[12]

Chapter 5. RESULTS

5.1 BOOST IN PERFORMANCE - CALTECH101

Table 5.4 shows the boost in performance that results from using our late-fusing method to include color in the Single-Level SPR model; Table 5.5, the boost in performance that results from using our late-fusing method to include color in the Single-Level-SPR/HPS-Lite model; and Table 5.6, the boost in performance that results from using our late-fusing method to include color in the Multi-Level-SPR model. The test results show the boost in performance from using our late-fusing method to include color ranges between about 1.90 – 2.50% for all three model types.

Table 5.4. Test results for Single-Level SPR relative to Caltech101.

Dataset	Vocab size	Single-Level-SPR				
		Pred. Acc. w/o color*		Pred. Acc. w/ color*		Boost
Caltech101		μ	σ	μ	σ	
	100	58.65	1.05	60.95	1.26	**2.30**
	200	59.42	0.72	61.76	0.78	**2.34**
	400	59.56	0.76	61.64	0.91	**2.08**

*Computed over 10 runs

Compared to the test data presented in Section 3.5, the test data here reflects the results of several optimizations and fine-tuning to improve performance. However, one effect of these improvements is that they mask the differences between the model types that were more apparent

[12] Since the average-test-time per image includes the I/O time required to retrieve the test image from disk or cloud storage plus the inference time, and this I/O time is invariant to whether color is included or not, the difference between these two measurements is the increase in the model's inference time.

in the test data reported in Section 3.5. For example, in Section 3.5, Single-Level-SPR/HPS-Lite is reported as outperforming Single-Level SPR by 0.75% at *N*=200, which we attribute to the complementary nature of the spatial information provided in that model, consisting of absolute and relative locations of object features.

Table 5.5. Test results for Single-Level-SPR/HPS-Lite relative to Caltech101.

Dataset	Vocab size	Single-Level-SPR/HPS-Lite				
		Pred. Acc w/o color*		Pred. Acc. w/ color*		Boost
Caltech101		μ	σ	μ	σ	
	100	58.64	1.25	60.92	1.03	**2.28**
	200	59.38	0.91	61.33	0.68	**1.95**
	400	58.28	1.31	60.33	1.08	**2.05**

*Computed over 10 runs

Table 5.6. Test results for Multi-Level SPR relative to Caltech101.

Dataset	Vocab size	Multi-Level SPR				
		Pred. Acc w/o color*		Pred. Acc. w/ color*		Boost
Caltech101		μ	σ	μ	σ	
	100	57.31	1.01	59.33	0.94	**2.02**
	200	58.27	0.68	60.72	0.50	**2.45**
	400	59.02	1.15	60.95	0.93	**1.93**

*Computed over 10 runs

However, in the current test results, at *N*=200, Single-Level-SPR/HPS-Lite is reported as slightly underperforming Single-Level SPR. We conjecture that one of our optimizations, which resulted in reducing the keypoint size in the dense detector from 16 to 6 pixels, is responsible for flipping the results between these two model types.

Another effect of these improvements is that the performance of several of the model configurations now peaks at *N*=200, for example, Single-Level-SPR with color, and Single-Level-SPR/HPS-Lite with and without color. This is actually more consistent with what is reported in the literature. For example, Khan et al. [4] reports their HPS_{ad}^{intra} model as peaking at *N*=200

when tested against Caltech101, and Lazebnik et al. [9] also reports their spatial pyramiding models as peaking at *N*=200.

This peaking occurs because, at *N*=200, the model histograms have high dimensionality, at least 5000 in each case. Since only about 1050 descriptors are generated for each image, the model histograms start to become sparsely populated at *N*=200. We conjecture that the reason this peaking was not more apparent in the data in Section 3.5 is because, with that data, each descriptor represented a larger patch.

Another observation from these results is that here, Multi-Level SPR is reported as underperforming Single-Level SPR, whereas in Lazebnik et al. [9], their multi-level spatial pyramiding model is reported as outperforming single-level spatial pyramiding. We conjecture that the flipping of these results can be explained by the fact that, in our models, the sole level in Single-Level SPR has a 5 x 5 granularity that exceeds the 4 x 4 granularity of the base level in Multi-Level SPR, whereas, in Lazebnik et al. [9], the granularity of the base layer of their multi-level model and that of the sole level of the single-level model are the same, i.e., 4 x 4.

5.2 BOOST IN PERFORMANCE - CALTECH256

Table 5.7 shows the boost in performance that results from using our late-fusing method to include color in the Single-Level-SPR model; Table 5.8, the boost in performance that results from using our late-fusing method to include color in the Single-Level-SPR/HPR-Lite model; and Table 5.9, the boost in performance that results from using our late-fusing method to include color in the Multi-Level-SPR model. The test results show the boost in performance from using our late-fusing method to include color ranges between about 1.90 – 3.20% for all three model types.

An interesting observation is that, for all but one model instance, Multi-Level SPR without color, model performance peaks at *N*=100 rather than *N*=200 as with the Caltech101

results. We believe this difference is due to the 12-pixel spacing used in the dense detector for our Caltech256 testing, compared to the 8-pixel spacing used with our Caltech101 testing, which causes the histograms to become sparely populated at a lower vocabulary size with Caltech256 compared to Caltech101.

Table 5.7. Test results for Single-Level SPR relative to Caltech256.

Dataset	Vocab size	Single-Level SPR		
Caltech256		Pred. Acc w/o color*	Pred. Acc. w/ color*	Boost
	50	21.09	23.78	**2.69**
	100	22.48	25.17	**2.69**
	200	22.36	24.48	**2.12**

*Computed over a single run

Table 5.8. Test results for Single-Level-SPR/HPS-Lite relative to Caltech256.

Dataset	Vocab size	Single-Level-SPR/HPS-Lite		
Caltech256		Pred. Acc w/o color*	Pred. Acc. w/ color*	Boost
	50	22.38	24.79	**2.41**
	100	23.41	25.64	**2.23**
	200	22.74	24.62	**1.88**

*Computed over a single run

Table 5.9. Test results for Multi-Level SPR relative to Caltech256.

Dataset	Vocab size	Multi-Level SPR		
Caltech256		Pred. Acc w/o color*	Pred. Acc. w/ color*	Boost
	50	21.26	23.87	**2.61**
	100	22.77	25.98	**3.21**
	200	22.98	25.78	**2.80**

*Computed over a single run

5.3 Comparison with Alternative Color Descriptors

When a hue descriptor, HoWH, is substituted for HoNC in our late-fusing method, as shown in Table 5.10, the boost in performance for each of the models drops significantly. In particular, the boost in performance for the Single-Level SPR model drops from 2.34 to 0.82, that

for the Single-Level SPR/HPS-Lite model drops from 1.95 to 0.77, and that for the Multi-Level SPR model drops from 2.45 to 0.65.

To test whether these differences are statistically significant, for each model type, we performed a two-sample t-test to compare the mean boost in performance from two samples. The first sample consisted of 7 independent runs at $N = 200$ to measure the mean boost in performance from the HoNC descriptor, which was 2.24, 1.99 and 2.46, respectively, for the Single-Level SPR, Single-Level SPR/HPS-Lite, and Multi-Level SPR models. The second sample consisted of 3 independent runs at $N = 200$ to measure the mean boost in performance from the HoWH descriptor, which was 0.82, 0.77, and 0.64, respectively, for the Single-Level SPR, Single-Level SPR/HPS-Lite, and Multi-Level SPR models. All the requirements for the two-sample t-test were met. In particular, each run was independent of all the others, the data appeared to be normally distributed as confirmed by histogram and Q-Q plots, the data was generated using a random sampling method, and the data in the two samples had about the same sample variance.

Table 5.10. Test results for Caltech101 with HoWH substituted for HoNC.

Dataset	Model	Performance		
Caltech101		Pred. Acc w/o hue*	Pred. Acc. w/ hue*	Boost from hue
	Single-Level SPR	59.44	60.26	**0.82** [drop from 2.34]
	Single-Level SPR/HPS-Lite	59.07	59.84	**0.77** [drop from 1.95]
	Multi-Level SPR	58.27	58.92	**0.65** [drop from 2.45]

*Computed over three runs at N=200 with α set to 0.05

The null hypothesis is that the mean boost in performance from the two samples is the same, and the alternative hypothesis is that the mean boost from HoWH is less than that for HoNC. The one-tailed p values were 0.000216, 0.007487, 0.002118, respectively, for the Single-SPR, Single-Level SPR/HPS-Lite, and Multi-Level SPR models, indicating that the null hypothesis can

be rejected in all three cases at a 1% significance level. These results confirm that HoNC significantly outperforms HoWH in our method relative to Caltech101.

When color normalization in HoNC is disabled, as shown in Table 5.11, the boost in performance for each of the models again drops significantly. In particular, the boost in performance for the Single-Level SPR model drops from 2.34 to 0.89, that for the Single-Level SPR/HPS-Lite model drops from 1.95 to 1.05, and that for the Multi-Level SPR model drops from 2.45 to 1.75.

To test whether these differences are statistically significant, we again performed a two-sample t-test to compare the mean boost in performance from two samples. The first sample consisted of 7 independent runs at $N = 200$ to measure the mean boost in performance from the HoNC descriptor, which was 2.24, 1.99, and 2.46, respectively, for the Single-Level SPR, Single-Level SPR/HPS-Lite, and Multi-Level SPR models. The second sample consisted of 3 independent runs at $N = 200$ to measure the mean boost in performance from the HoNC descriptor with color normalization disabled, which was 0.89, 1.05, and 1.75, respectively, for the Single-Level SPR, Single-Level SPR/HPS-Lite, and Multi-Level SPR models. As before, all the requirements for the two-sample t-test were met. In particular, each run was independent of the others, the data appeared to be normally distributed as confirmed by histogram and Q-Q plots, the data was generated using a random sampling method, and the data in the two samples had about the same sample variance.

The null hypothesis is that the mean boost from the two samples is the same, and the alternative hypothesis is that the mean boost from HoNC with color normalization disabled is less than that for HoNC. The one-tailed p values were 0.000224, 0.027864, 0.060417, respectively, for the Single-SPR, Single-Level SPR/HPS-Lite, and Multi-Level SPR models, indicating that the

null hypothesis in the first case can be rejected at a 1% significance level, that for the second case can be rejected at a 5% significance level, and that for the third case can be rejected at the 10% significance level. These results confirm that, at least for the first two cases, HoNC significantly outperforms HoNC with color normalization disabled in our method relative to Caltech101, and that the illumination intensity invariance provided by color normalization in HoNC is responsible for a statistically significant performance boost.

Table 5.11. Test results for Caltech101 with color normalization disabled in HoNC.

Dataset	Model	Performance		
Caltech101		Pred. Acc w/o color*	Pred. Acc. w/ color*	Boost
	Single-Level SPR	59.44	60.33	**0.89** [drop from 2.34]
	Single-Level SPR/HPS-Lite	59.07	60.12	**1.05** [drop from 1.95]
	Multi-Level SPR	58.27	60.02	**1.75** [drop from 2.45]

*Computed over three runs at N=200 with α set to 0.20

Table 5.12 and Table 5.13 shows the results of substituting HoWH and HoNC-none for HoNC relative to Caltech256. Table 5.12 shows that, when HoWH is substituted from HoNC in our late-fusing method, the boost in performance again drops significantly for each of the three models, confirming that, for Caltech256 as well, HoNC significantly outperforms HoWH.

Table 5.13 shows that, when color normalization in HoNC is disabled, the boost in performance for each of the three models again drops for Caltech256, although the results are not as pronounced here as with Caltech101. These results suggest that, for Caltech256, the illumination intensity invariance resulting from the color normalization in HoNC has a more muted effect than for Caltech101.

We did not test the CN descriptor in our method, but instead relied on test results for that descriptor as reported in Elfiky et al. [1], which are summarized in Table 5.14. More specifically, that table shows the combined boost in performance, computed from data reported in Elfiky et al. [1], resulting from jointly using the CN descriptor in combination with a second, self-similarity descriptor in a late-fusing method.

Table 5.12. Test results for Caltech256 with HoWH substituted from HoNC.

Dataset	Model	Performance		
Caltech256		Pred. Acc. w/o hue	Pred. Acc. w/ hue*	Boost*
	Single-Level SPR	22.48	23.71	**1.23** [drop from 2.69]
	Single-Level SPR/HPS-Lite	23.41	24.32	**0.91** [drop from 2.23]
	Multi-Level SPR	22.77	24.09	**1.32** [drop from 3.21]

*Computed over single run at N=100 with α set to 0.05

Table 5.13. Test results for Caltech256 with color normalization disabled in HoNC.

Dataset	Model	Performance		
Caltech256		Pred. Acc. w/o color	Pred. Acc. w/ color*	Boost*
	Single-Level SPR	22.48	24.88	**2.40** [drop from 2.69]
	Single-Level SPR/HPS-Lite	23.41	25.45	**2.04** [drop from 2.23]
	Multi-Level SPR	22.77	25.80	**3.03** [drop from 3.21]

*Computed over single run at N=100 with α set to 0.20

As can be seen from Table 5.14, the 1.9 – 3.2 % boost in performance we found for the HoNC descriptor exceeds that of the CN descriptor, since the latter will only be a fraction of the boost values reported in the table.

Table 5.14. Boost in performance from using CN and self-similarity descriptor.

Dataset	No. of image classes	Boost
Sports	8	**1.5**
15 Scenes	15	**2.3**
Butterflies	7	**2.4**
Pascal 2007	20	**2.3**
Pascal 2009	20	**2.4**

5.4 REAL-TIME PERFORMANCE

Table 5.15 and Table 5.16 shows the increase in inference time that results from using our late-fusing method to include color in the highest performing model configurations.

Table 5.15. Increase in inference time for selected model configurations - Caltech101.

Model	Increase in Inf. Time (mS) *	Pred. Acc.	Dataset
Single-level SPR w/color@N=400	**147.08**	61.64	Caltech101
Single-level SPR w/color@N=200	**150.56**	60.95	Caltech101
Single-level SPR/HPS-Lite w/ color@N=100	**152.25**	60.92	Caltech101
Multi-level SPR w/color@N=400	**165.87**	60.95	Caltech101
Multi-level SPR w/color@N=200	**181.73**	60.95	Caltech101

*Computed over 10 runs

Table 5.16. Increase in inference time for selected model configurations - Caltech256.

Model	Increase in Inf. Time (mS) *	Pred. Acc.	Dataset
Multi-level SPR w/color@N=100	**174.46**	25.98	Caltech256
Single-level SPR w/color@N=100	**190.54**	25.17	Caltech256
Multi-level SPR w/ color@N=200	**192.50**	25.78	Caltech256

*Computed over a single run

As can be seen, in all instances, the increase in inference time resulting from including color using our late-fusing method ranges between about 150 – 190 mS.

5.5 COMPARISON WITH STATE-OF-THE-ART

Table 5.17 compares a theoretical detector, constructed by modifying the detector reported in Abdi et al. [5] to include color using the subject method, with two state-of-the-art detectors, the unmodified detector from Abdi et al. [5], and a CNN-based detector, NAS-FPN AmoebaNet [7]. It is assumed that, by including color in the unmodified detector using the subject method, the predictive accuracy jumps by 2.55%, which is the midpoint of the 1.9 – 3.2 % range we measured for the HoNC descriptor, and the inference time jumps by 170 mS, which again is the midpoint of the 150 – 190 mS range we measured for the HoNC descriptor.

Table 5.17. Comparison with state-of-the-art models.

Model	Perf.	Inf. Time (mS)	Dataset	No. of classes	Date
SE-BOVW w/o color	90.48% (Pred. Acc.)	100	GTSRB	43	2018 [5]
SE-BOVW w/ color	93.03% (Pred. Acc.)	270	GTSRB	43	Theoretical
NAS-FPN AmoebaNet*	48.3% (mAP)	278.9	COCO	80	2019 [7]

*Currently ranked #15 on the Facebook AI leaderboard for real-time object detection - COCO dataset[13].

While the 270 mS total inference time may disqualify the modified detector from some real-time applications, for example, the Intelligent Driver Assistance System (IDAS) described in Abdi et al. [5] where the total inference time cannot exceed 100 mS, the total inference time is still within the range currently posted on the Facebook AI leaderboard for state-of-the-art real-time object detectors such as NAS-FPN AmoebaNet. The modified detector should be appropriate for other real-time applications where more relaxed total inference time requirements apply, such as

[13] https://paperswithcode.com/task/real-time-object-detection

intruder detection in home security systems. Of note is that further reductions in the total inference time should be possible when moving to an optimized, production-oriented hardware environment that provides CPU/GPU parallelism, such as that used to test the CNN-based detectors.

5.6 Testing of Texture Descriptors

In this section, we extend our previous work to include testing a late-fusing method for joining shape and texture in a spatially-enhanced BOVW model. Our architecture, shown in Figure 5.1, is identical to that depicted in Section 3.4, but with the color descriptor replaced by a texture descriptor, such as HoNI, SPIN, CHoNI, or CSPIN.

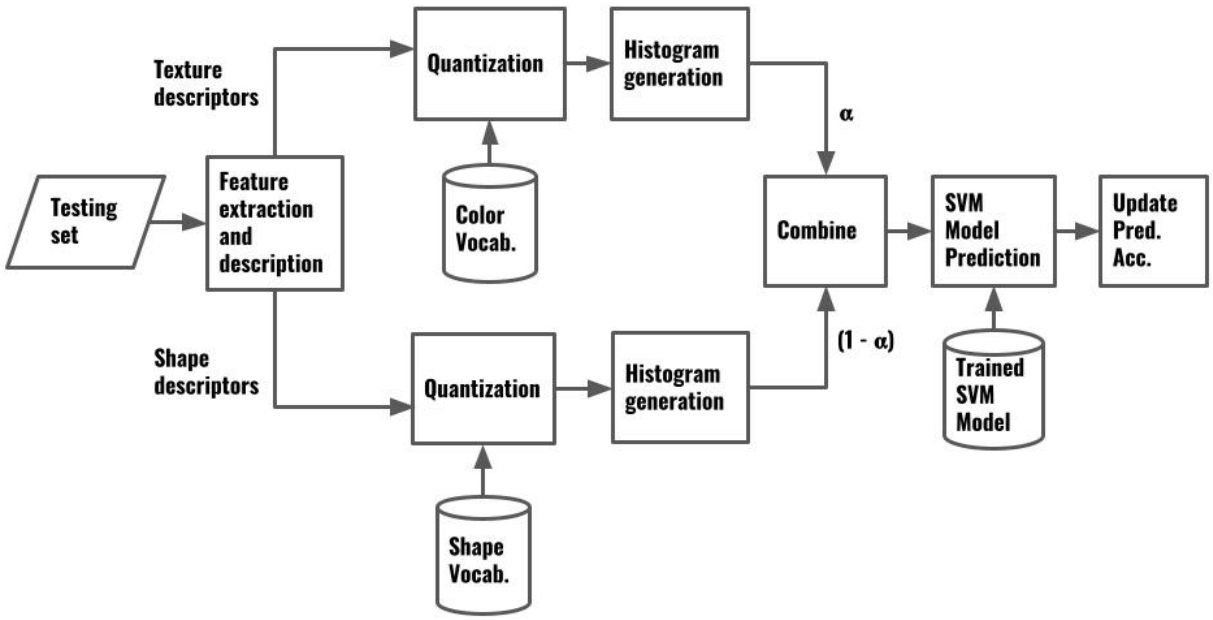

Figure 5.1. Late-fusing architecture for including texture or texture/color descriptors.

An application that might benefit from such a model is image classification in a home-security system equipped with security cameras for capturing images, which can be black-and-white or color depending on time of day and quality of lighting, of a sensitive area (such as a

backyard). In this application, the image classification takes place on a resource-constrained device such as an on-premises computer or a cloud edge device. The images captured by the security cameras are input to the image classification model, which is tasked with triggering an alarm in real time if a human intruder is detected from the images. To avoid false positives, the image classification should be capable of distinguishing human intruders from pets and wildlife such as raccoons or deer largely on the basis of shape and texture, which are assumed to be the dominant cues. Color, if present, is assumed to be a secondary cue.

In our first set of tests, we included HoNI as the texture descriptor and SIFT as the shape descriptor. We set α to 0.20 after determining through testing that value maximized performance compared to alternative settings at 0.10, 0.30, and 0.40, respectively. We also set *N* to 200, having previously determined that value optimized performance relative to the Caltech101 dataset. The test results are shown in Table 5.18.

Table 5.18. Test results with HoNI selected as the texture descriptor.

Dataset	Model	Performance		
Caltech101		Pred. Acc w/o texture*	Pred. Acc. w/ texture*	Boost
	Single-Level SPR	59.44	59.36	**(0.08)**
	Single-Level SPR/HPS-Lite	59.07	58.96	**(0.11)**
	Multi-Level SPR	58.27	58.79	**0.52**

*Computed over three runs at *N*=200

As can be seen, for the first two models, there is a negative boost in performance, meaning the addition of texture hurts performance, while, for the third model, the boost in performance is modest, about 0.5%.

In our second set of tests, we included SPIN as the texture descriptor and SIFT as the shape descriptor. We set α to 0.10, after determining through testing that value optimized performance

relative to alternative settings at 0.05 and 0.20, respectively. As before, we set *N* to 200, having previously determined that setting optimized performance relative to the Caltech101 dataset for the majority of models. The test results are shown in Table 5.19.

As can be seen, for the first two models, the boost in performance is negative, meaning the addition of texture hurts performance, while, for the third model, the boost in performance is nominal, i.e., less than 0.50%.

Table 5.19. Test results with SPIN selected as the texture descriptor.

Dataset	Model	Performance		
Caltech101		Pred. Acc w/o texture*	Pred. Acc. w/ texture*	Boost
	Single-Level SPR	59.44	59.38	**(0.06)**
	Single-Level SPR/HPS-Lite	59.07	58.81	**(0.26)**
	Multi-Level SPR	58.27	58.48	**0.21**

*Computed over three runs at *N*=200

For our third set of tests, we performed testing of pre-fused texture/color descriptors. We considered both CHoNI and CSPIN in this regard, but elected to go with CHoNI based on preliminary testing suggesting it would perform better. For our shape descriptor, we elected to go with SIFT as before. Table 5.20 shows the test results. As can be seen, the boost in performance ranges between about 0.50 – 1.00 % across the board and exceeds that of HoNI and SPIN for all three models. Accordingly, it appears to be the best choice for this application.

Table 5.20. Test results with CHoNI selected as the texture descriptor.

Dataset	Model	Performance		
Caltech101		Pred. Acc w/o texture*	Pred. Acc. w/ texture*	Boost
	Single-Level SPR	59.44	60.34	**0.90**
	Single-Level SPR/HPS-Lite	59.07	59.57	**0.50**
	Multi-Level SPR	58.27	59.23	**0.96**

*Computed over three runs at *N*=200

An interesting observation is that, in this testing, CHoNI performed better than HoNI and SPIN even though it represents an example of early fusing. We conjecture that CHoNI works in this context because of the high cross-correlation between the information that is fused with CHoNI, consisting of the same texture information projected onto different color channels, which renders the quantization error introduced by CHoNI low and manageable. The situation described in Section 3.2 is different because there, the low cross-correlation between the fused information greatly increases the quantization error that is introduced.

Chapter 6. CONCLUSION

In conclusion, we were successful in meeting our goal of providing a novel method for joining color with shape, featuring the HoNC descriptor, to improve the predictive accuracy of spatially-enhanced BOVW models for image classification.

Moreover, as shown in Table 6.21, summarizing the test results of Sections 5.1 and 5.2, we were also successful in showing that the method is generalizable to different model types, vocabulary sizes and datasets in that it boosts performance across the board in all such cases by about 1.9 – 3.2 %.

We also provided a theoretical explanation, based on quantization error analysis, that better explains why late-fusing outperforms combination descriptors/early fusing in the field of image classification. Prior work [6] had suggested that early-fusing should outperform late-fusing for image classification because it preserves the association between color and shape descriptors at the pixel level that is lost with late-fusing. We showed that any benefit from preserving this association is often outweighed by the quantization error that is introduced with early fusing.

Table 6.21. Summary of boost test results.

Dataset	Vocab. Size	Single-Level SPR	Single-Level SPR/HPS-Lite	Multi-Level SPR
Caltech101*	100	**2.30**	**2.28**	**2.02**
	200	**2.34**	**1.95**	**2.45**
	400	**2.08**	**2.05**	**1.93**
Caltech256†	50	**2.69**	**2.41**	**2.61**
	100	**2.69**	**2.23**	**3.21**
	400	**2.12**	**1.88**	**2.80**

*Results computed over 10 runs
†Results computed over singe run

A potential limitation of our work is that the late-fusing approach featured in our method does not scale well with additional cues such as texture.

An area for future research is investigating the use of bounded soft-assignment techniques [26] for reducing quantization error in general, which would appear to be suitable to real-time use cases.

BIBLIOGRAPHY

[1] N. M. Elfiky, F. Shahbaz Khan, J. van de Weijer, and J. Gonzàlez, "Discriminative compact pyramids for object and scene recognition," *Pattern Recognition*, vol. 45, no. 4, pp. 1627–1636, Apr. 2012, doi: 10.1016/j.patcog.2011.09.020.

[2] R. Khan, C. Barat, D. Muselet, and C. Ducottet, "Spatial orientations of visual word pairs to improve Bag-of-Visual-Words model," in *Proceedings of the British Machine Vision Conference 2012*, Surrey, 2012, p. 89.1-89.11, doi: 10.5244/C.26.89.

[3] D. A. R. Vigo, F. S. Khan, J. van de Weijer, and T. Gevers, "The Impact of Color on Bag-of-Words Based Object Recognition," in *2010 20th International Conference on Pattern Recognition*, Aug. 2010, pp. 1549–1553, doi: 10.1109/ICPR.2010.383.

[4] R. Khan, C. Barat, D. Muselet, and C. Ducottet, "Spatial histograms of soft pairwise similar patches to improve the bag-of-visual-words model," *Computer Vision and Image Understanding*, vol. 132, pp. 102–112, Mar. 2015, doi: 10.1016/j.cviu.2014.09.005.

[5] L. Abdi and A. Meddeb, "Spatially Enhanced Bags of Visual Words Representation to Improve Traffic Signs Recognition," *J Sign Process Syst*, vol. 90, no. 12, pp. 1729–1741, Dec. 2018, doi: 10.1007/s11265-017-1324-9.

[6] J. Dai, Y. Li, K. He, and J. Sun, "R-FCN: Object Detection via Region-based Fully Convolutional Networks," *arXiv:1605.06409 [cs]*, Jun. 2016, Accessed: May 09, 2021. [Online]. Available: http://arxiv.org/abs/1605.06409.

[7] G. Ghiasi, T.-Y. Lin, R. Pang, and Q. V. Le, "NAS-FPN: Learning Scalable Feature Pyramid Architecture for Object Detection," *arXiv:1904.07392 [cs]*, Apr. 2019, Accessed: May 09, 2021. [Online]. Available: http://arxiv.org/abs/1904.07392.

[8] J. Van De Weijer and F. S. Khan, "Fusing color and shape for bag-of-words based object recognition," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 7786 LNCS, pp. 25–34, 2013, doi: 10.1007/978-3-642-36700-7_3.

[9] S. Lazebnik, C. Schmid, and J. Ponce, "Spatial Pyramid Matching," in *Object Categorization*, S. J. Dickinson, A. Leonardis, B. Schiele, and M. J. Tarr, Eds. Cambridge: Cambridge University Press, 2009, pp. 401–415.

[10] F. Khan, R. Anwer, J. Weijer, A. Bagdanov, M. Vanrell, and A. López, "Color Attributes for Object Detection," *Proceedings / CVPR, IEEE Computer Society Conference on Computer*

Vision and Pattern Recognition. IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Jun. 2012, doi: 10.1109/CVPR.2012.6248068.

[11] N. S. Mansoori, M. Nejati, P. Razzaghi, and S. Samavi, "Bag of visual words approach for image retrieval using color information," in *2013 21st Iranian Conference on Electrical Engineering (ICEE)*, Mashhad, Iran, May 2013, pp. 1–6, doi: 10.1109/IranianCEE.2013.6599562.

[12] K. E. A. van de Sande, T. Gevers, and C. G. M. Snoek, "Evaluating Color Descriptors for Object and Scene Recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, no. 9, pp. 1582–1596, Sep. 2010, doi: 10.1109/TPAMI.2009.154.

[13] J. van de Weijer, C. Schmid, J. Verbeek, and D. Larlus, "Learning Color Names for Real-World Applications," *IEEE Transactions on Image Processing*, vol. 18, no. 7, pp. 1512–1523, Jul. 2009, doi: 10.1109/TIP.2009.2019809.

[14] J. van de Weijer and C. Schmid, "Coloring Local Feature Extraction," in *Computer Vision – ECCV 2006*, Berlin, Heidelberg, 2006, pp. 334–348, doi: 10.1007/11744047_26.

[15] S. Kopf *et al.*, "Enhancing bag of visual words with color information for iconic image classification," in *IPCV 2016 : proceedings of the 2016 International Conference on Image Processing, Computer Vision, & Pattern Recognition : WORLDCOMP '16, July 25-28, 2016, Las Vegas, Nevada, USA*, Las Vegas, NV, 2017, pp. 206–209, [Online]. Available: https://madoc.bib.uni-mannheim.de/42193/.

[16] C. F. Olson, S. A. Hoover, J. L. Soltman, and S. Zhang, "Complementary Keypoint Descriptors," in *Advances in Visual Computing*, vol. 10072, G. Bebis, R. Boyle, B. Parvin, D. Koracin, F. Porikli, S. Skaff, A. Entezari, J. Min, D. Iwai, A. Sadagic, C. Scheidegger, and T. Isenberg, Eds. Cham: Springer International Publishing, 2016, pp. 341–352.

[17] C. F. Olson and S. Zhang, "Keypoint Recognition with Histograms of Normalized Colors," in *2016 13th Conference on Computer and Robot Vision (CRV)*, Victoria, BC, Canada, Jun. 2016, pp. 311–318, doi: 10.1109/CRV.2016.37.

[18] C. Wengert, M. Douze, and H. Jégou, "Bag-of-colors for improved image search," in *Proceedings of the 19th ACM international conference on Multimedia*, New York, NY, USA, Nov. 2011, pp. 1437–1440, doi: 10.1145/2072298.2072034.

[19] J. M. dos Santos, E. S. de Moura, A. S. da Silva, and R. da Silva Torres, "Color and texture applied to a signature-based bag of visual words method for image retrieval," *Multimed Tools Appl*, vol. 76, no. 15, pp. 16855–16872, Aug. 2017, doi: 10.1007/s11042-016-3955-4.

[20] A. Khokher and R. Talwar, “A fast and effective image retrieval scheme using color-, texture-, and shape-based histograms,” *Multimed Tools Appl*, vol. 76, no. 20, pp. 21787–21809, Oct. 2017, doi: 10.1007/s11042-016-4096-5.

[21] K.-K. Seo, “An application of one-class support vector machines in content-based image retrieval,” *Expert Systems with Applications*, vol. 33, no. 2, pp. 491–498, Aug. 2007, doi: 10.1016/j.eswa.2006.05.030.

[22] Y. Alqasrawi, D. Neagu, and P. I. Cowling, “Fusing integrated visual vocabularies-based bag of visual words and weighted colour moments on spatial pyramid layout for natural scene image classification,” *SIViP*, vol. 7, no. 4, pp. 759–775, Jul. 2013, doi: 10.1007/s11760-011-0266-0.

[23] M. Soniya, P. Suhasini, “Integrated SURF and Spatial Augmented Color Feature Based Bovw Model with Svm for Image Classification,” *ijeat*, vol. 8, no. 6, pp. 1875–1877, Aug. 2019, doi: 10.35940/ijeat.F7900.088619.

[24] H. Hoashi, T. Joutou, and K. Yanai, “Image Recognition of 85 Food Categories by Feature Fusion,” in *2010 IEEE International Symposium on Multimedia*, Taichung, Taiwan, Dec. 2010, pp. 296–301, doi: 10.1109/ISM.2010.51.

[25] A. Kolesnikov, E. Trichina, and T. Kauranne, “Estimating the number of clusters in a numerical data set via quantization error modeling,” *Pattern Recognition*, vol. 48, no. 3, pp. 941–952, Mar. 2015, doi: 10.1016/j.patcog.2014.09.017.

[26] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman, “Lost in quantization: Improving particular object retrieval in large scale image databases,” in *2008 IEEE Conference on Computer Vision and Pattern Recognition*, Jun. 2008, pp. 1–8, doi: 10.1109/CVPR.2008.4587635.

[27] F. S. Khan, J. van de Weijer, A. D. Bagdanov, and M. Vanrell, “Portmanteau vocabularies for multi-cue image representation,” in *Proceedings of the 24th International Conference on Neural Information Processing Systems*, Red Hook, NY, USA, Dec. 2011, pp. 1323–1331, Accessed: Feb. 14, 2021. [Online].

[28] J. van de Weijer and C. Schmid, “Applying Color Names to Image Description,” in *2007 IEEE International Conference on Image Processing*, San Antonio, TX, USA, 2007, p. III-493-III–496, doi: 10.1109/ICIP.2007.4379354.

APPENDIX A

BOVW	Bag of Visual Words
CN	color names
CNN	Convolutional Neural Network
DITC	Divisive Information-Theoretic feature Clustering
FPS	frames per second
HoNC	Histogram of Normalized Colors
HoNI	Histogram of Normalized Intensities
HoWH	Histogram of Weighted Hues
mAP	mean average precision
MSE	Mean Squared Error
SIFT	Scale Invariant Feature Transform
SPR	Spatial Pyramiding Representation
SVM	Support Vector Machine