Bag of Visual Words Classifier

Computer Vision and Pattern Recognition Exam
University of Trieste (UniTS)

Marco Tallone

January 2025

Abstract

This report presents the implementation of a Bag of Visual Words (BoW) image classifier. The objective is to build a classifier for scene recognition by building a visual vocabulary from a set of images and performing multi-class classification. The visual vocabulary is built by clustering SIFT descriptors extracted from the images, and classification is performed comparing K-Nearest Neighbors (KNN) and Support Vector Machines (SVM) classifiers with different kernels. Results show that the best performance is achieved by the SVM classifiers, in particular when implemented with a spatial pyramid feature representation to add spatial information to the classic BoW approach.

1 Introduction

The Bag of Visual Words (BoW) model is a popular computer vision technique used for image classification or retrieval. Inspired by the analogous model used in natural language processing, this approach it's based on the idea of treating images as collections of visual words belonging to a visual vocabulary, which is obtained by clustering local features extracted from a (possibly different) set of representative images.
This report presents an implementation the BoW image classifier for scene recognition by first building a visual vocabulary from a set of test images and then performing multi-class classification using K-Nearest Neighbors (KNN) and Support Vector Machines (SVM) classifiers.
In particular, the visual vocabulary is built by clustering SIFT descriptors extracted from the test images using the K-Means algorithm. Descriptors are computed both from keypoints detected with the SIFT detector and from ones obtained by dense sampling of the images with a fixed grid.
In the classification phase, the performance of the KNN classifier is compared with that of different SVM classifiers, all using the "*one-vs-all*" strategy with different kernels.
Additionally, different ways to represent images as input feature vectors are tested. These include the classic representation as normalized histograms of visual words, the implementation of the *soft assignment* techniques proposed by *Van Gemert et al.* [7] and the use of the *spatial pyramid feature representation* proposed by *Lazebnik et al.* [3].
The objectives of this study are to compare the performance of the different classifiers and image representations and to reproduce the results obtained by *Van Gemert et al.* [7] and *Lazebnik et al.* [3] on the *15-Scenes* dataset.
The report is organized as follows. Section 2 presents the dataset used and analyzes the images distribution. Section 3 explains how the visual vocabulary is built from the sampled descriptors, while in Section 4 the different image representation techniques are compared. Then, Section 5 presents the classifiers used for the multi-class classification task, while in Section 6 the results obtained in this study are summarized, followed by some final considerations in the last section.
From a technical point of view, all the classifiers have been implemented in Python 3.12 adopting the `scikit-learn` library [4] (version `1.5.2`). In particular, the SVM classifier have been implemented using the `SVC` class, which internally relies on `libsvm` [1]. However, with the aim of maintaining a clear and concise report of the work, the following sections will only present concepts from a theoretical point of view and the most noticeable results of the study, while further analysis and implementation details will be available on the author's GitHub repository [6].

2 Dataset Description

The dataset used in this study is the *15-Scenes* dataset from *Lazebnik et al.* [3]. This dataset consists of a total of 4485 grayscale images, each belonging to one of 15 different scene categories (*Office, Kitchen, LivingRoom, Bedroom, Store, Industrial, TallBuilding, InsideCity, Street, Highway, Coast, OpenCountry, Mountain, Forest, Suburb*). The images are divided into a training set of 1500 images, with 100 images per category, and a test set of 2985 images with a non-uniform distribution as shown in Figure 2.1.
Moreover, it's worth mentioning that the images in the dataset have different sizes and aspect ratios, with the majority of them being 256×256 pixels.

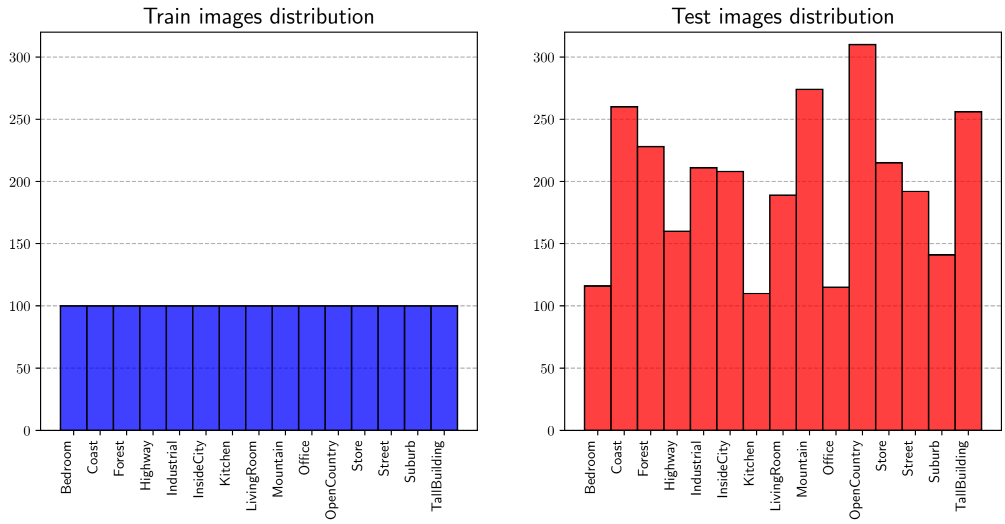

Figure 2.1: Distribution of images in train (*blue*) and test (*red*) sets.

3 Visual Vocabulary Construction

The process of building a visual vocabulary can be split in two main steps:

1. **Feature extraction**: sampling of SIFT descriptors from train images.
2. **Clustering**: grouping of the extracted descriptors into K clusters.

3.1 Feature Extraction

The first step involves the extraction of local features from the images in the training set. In this case, SIFT descriptors have been used as features, and they have been extracted following two different approaches:

1. Using the **SIFT detector**: the SIFT detector is used to detect keypoints in the images and the corresponding SIFT descriptors are computed on these keypoints.
2. Sampling on a **dense regular grid**: following the approach adopted by *Lazebnik et al.* [3], SIFT descriptors can also be computed from keypoints sampled on a regular dense grid over each image. Following the approach of the paper, a regular grid with spacing of 8 pixels between each keypoint has been used.

Both sampling methods have been tested and compared, with the results shown in later section 6. For the first approach, the SIFT detector has been initialized to retain only the best 500 features in each image, resulting in a total of 593 006 keypoints, while for the second approach 1 482 434 keypoints have been collected.
Notice that with both approaches the exact number of keypoints extracted in each image can change from one image to another. In the first case, this is due to the content of the images, i.e. the actual number of features that can be detected by the SIFT detector. In the second case instead, this is due to the different sizes and aspect ratios of the images. Figure 3.1 shows an example of this phenomenon, where a different number of keypoints has been detected by the SIFT detector in two different images, while Appendix A1 shows the distribution descriptors sampled from the train set with the two methods.

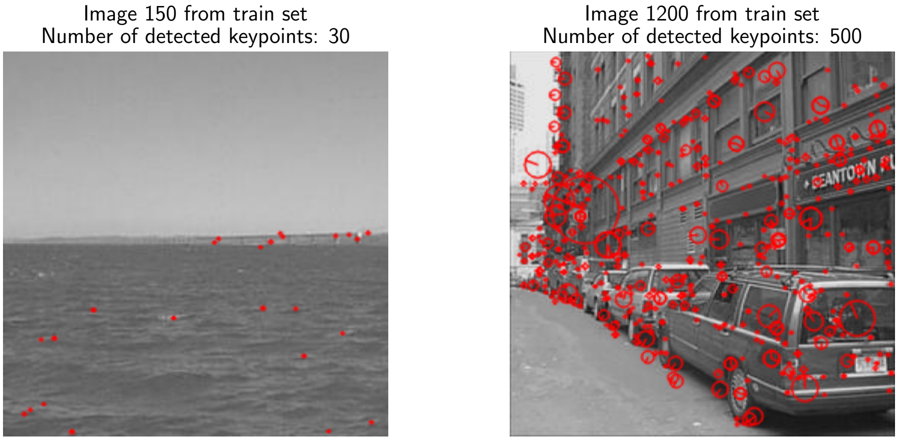

Figure 3.1: Example of keypoints detection with the SIFT detector. In the left side image only 30 keypoints were detected, while in the right side one all the desired 500 keypoints were found.

3.2 Clustering

Following feature extraction, clustering has been performed on a random subset of extracted SIFT descriptors to build the visual vocabulary. Hence, the resulting centroids, representing the visual words of the vocabulary, are 128-dimensional vectors.
The size of the subset has been chosen to maintain about the same ratio between the total number of descriptors and the number of descriptors used for clustering in the two feature extraction methods. Hence, this has been set to 10 000 for descriptors sampled using the SIFT detector and to 25 000 for descriptors sampled on a regular grid.
Clustering has then been performed using the *k-means* algorithm to group descriptors into K clusters, where K becomes the size of the visual vocabulary. The number of clusters is an hyperparameter of the clustering process which has to be chosen. An analysis of the average silhouette score [5] has been performed for values of K ranging from 50 to 1000 with the results shown in figure 3.2. Ultimately, the value of $K = 400$ has been chosen as a good trade-off between the quality of the clustering and the overall computational cost of the clustering process. Moreover, the choice is also motivated by the final results obtained in Section 6 and by the fact that, given the objectives of this project, this value allows for a direct comparison with the results reported by *Lazebnik et al.* [3], where the same vocabulary size was used.

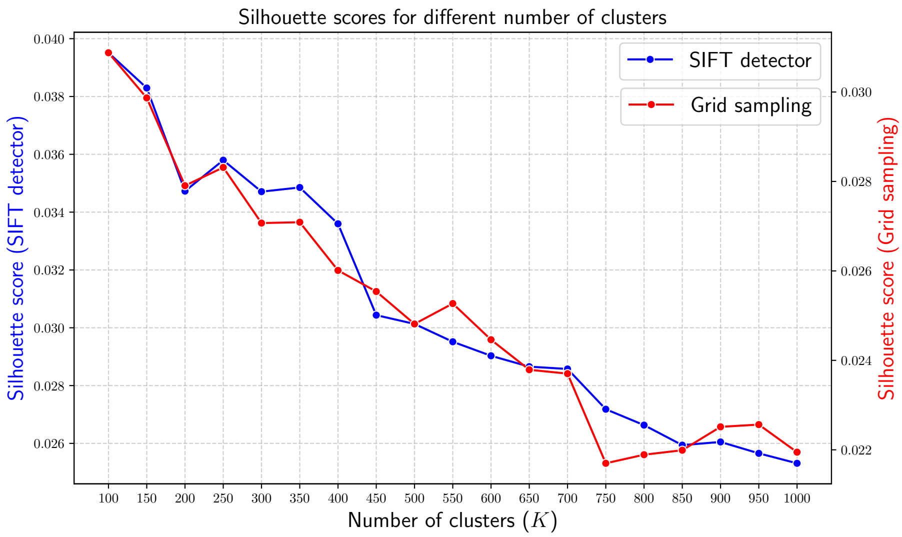

Figure 3.2: Silhouette scores resulting from clustering SIFT descriptors sampled using the SIFT detector (*blue*) and on a regular grid (*red*). As expected, the value decreases with the increase of K. Noticeable decreases occur after $K \approx 400$ and $K \approx 700$.

4 Image Representation

A important step in the Bag of Visual Words (BoW) pipeline is the representation of images as fixed-length feature vectors. In this study, 4 different techniques have been implemented to perform this task.
The first, most common approach is to represent images as *normalized histograms of visual words (HIST)*. For each image, the previously extracted SIFT descriptors are assigned to the closest visual word in the vocabulary, and a histogram is built counting the occurrences of each visual word in the image. Formally the i-th bin of the histogram for image d is computed as

$$h_i = \frac{n_{id}}{n_d}, \quad \forall i \in \{0, \ldots, K-1\}, \forall d \in \{0, \ldots, N-1\} \tag{4.1}$$

where n_{id} is the number of occurrences of the i-th visual word in image d, n_d is the total number of visual words in image d and N is the total number of images in the dataset. After normalization, the result is a fixed-length representation in which images are seen as normalized histograms having K bins, corresponding to visual words frequencies.
In order to account for the relevance of the visual words, another idea is to use the *term frequency-inverse document frequency (TF-IDF)* weighting scheme. In this case, the elements t_i of the histogram representation are computed as

$$h_i = \frac{n_{id}}{n_d} \cdot \log\left(\frac{N}{N_i}\right), \quad \forall i \in \{0, \ldots, K-1\}, \forall d \in \{0, \ldots, N-1\} \tag{4.2}$$

where n_{id} is the number of occurrences of the i-th visual word in image d, n_d is the total number of visual words (descriptors) in the image, N is the total number of images in the dataset, and N_i is the number of images containing the i-th visual word.
A third approach is to use the *soft assignment* techniques proposed by *Van Gemert et al.* [7], which argue that the hard assignment of visual words to descriptors gives rise to two types of ambiguity: *codeword uncertainty* when a descriptor is similarly close to two or more visual words, and *codeword plausibility* when a descriptor is relatively far from all visual words. To address this issue, the authors propose the usage of a Gaussian kernel $K_\sigma(x)$ density estimator[1] based on the Euclidean distance $D(w_i, x_j)$ between the i-th visual word w_i and the j-th descriptor x_j to approximate the probability density function of the descriptors around the visual words under the assumption that descriptors are normally distributed around visual words with standard deviation σ. Hence, by replacing the histogram estimator with the kernel density estimator the result is the *kernel codebook (KCB)* representation, in which each descriptor can contribute to multiple bins according to the equation

$$h_i = \frac{1}{n_d} \sum_{j=0}^{n_d} K_\sigma(D(w_i, x_j)), \quad \forall i \in \{0, \ldots, K-1\}, \forall d \in \{0, \ldots, N-1\} \tag{4.3}$$

According to the authors, this soft assignment strategy models the two types of ambiguity at the same time, but it's also possible to consider them separately by adopting either the *codeword uncertainty (UNC)* representation

$$h_i = \frac{1}{n_d} \sum_{j=0}^{n_d} \frac{K_\sigma(D(w_i, x_j))}{\sum_{l=0}^{K-1} K_\sigma(D(w_l, x_j))}, \tag{4.4}$$

or the *codeword plausibility (PLA)* representation

$$h_i = \frac{1}{n_d} \sum_{j=0}^{n_d} \begin{cases} K_\sigma(D(w_i, x_j)), & \text{if } w_i = \underset{l \in \{0, \ldots, K-1\}}{\operatorname{argmin}} D(w_l, x_j) \\ 0, & \text{otherwise} \end{cases} \tag{4.5}$$

both $\forall i \in \{0, \ldots, K-1\}$, $\forall d \in \{0, \ldots, N-1\}$.
The last approach used to fulfill the image representation task is the use of *spatial pyramid matching (SPM)* method proposed by *Lazebnik et al.* [3]. The idea behind the spatial pyramid is to repeatedly subdivide each image into subsequently finer grids at different resolution levels $\ell \in \{0, \ldots, L-1\}$, and to compute the histograms $H_\ell(g)$ of visual words for each g^{th} grid at level ℓ. Precisely, at level ℓ each image is divided into 2^ℓ cells along each dimension, and the count of visual words is computed for each cell.

[1] See Appendix A2.

All the histograms are then weighted according to a multiplicative weighting scheme that favors features counts computed at higher (finer resolution) levels of the pyramid while penalizing those at lower levels. In particular, if L is the number of levels in the pyramid, the weights multiplied to all the histograms at level ℓ are computed as

$$\omega_0 = \frac{1}{2^L}, \quad \omega_\ell = \frac{1}{2^{L-\ell+1}}, \quad \forall \ell \in \{0, \dots, L-1\} \tag{4.6}$$

The resulting weighted histograms are then stacked together to form, for each image, a single *extended multi-level descriptor* that can be used as input to the SVM classifiers presented in Section 5. Such descriptor, not only contains the information about the visual words, but also encodes positional information at different scales which was missing in the standard BoW approach.
The idea of using SVM classifiers specifically for this representation comes from the posibility of adopting the *spatial pyramid kernel* in order to compute the similarity between these extended descriptors by correctly measuring the information at different levels of the pyramid. Formally, given two images X and Y represented by their extended descriptors

$$\begin{aligned} H^X &= \{H^X_{\ell,i}(g) \mid \forall \ell \in \{0, \dots, L-1\}, \forall i \in \{0, \dots, K-1\}, \forall g \in \{0, \dots, 2^{2\ell} - 1\}\} \\ H^Y &= \{H^Y_{\ell,i}(g) \mid \forall \ell \in \{0, \dots, L-1\}, \forall i \in \{0, \dots, K-1\}, \forall g \in \{0, \dots, 2^{2\ell} - 1\}\} \end{aligned}$$

the *pyramid match kernel* between the two images is computed as

$$K^L(X,Y) = \sum_{\ell=0}^{L-1} \sum_{i=0}^{K-1} \sum_{g=0}^{2^{2\ell}-1} \min\left(H^X_{\ell,i}(g), H^Y_{\ell,i}(g)\right) \tag{4.7}$$

but from a practical point of view, this can be efficiently implemented as a simple histogram intersection kernel.
Finally, it's important to notice that for this last representation the features are extracted from images only by using the grid approach. This has been done both to follow the same approach of the original paper and also to avoid the possibility of having empty cells which might happen for those containing uniform regions of the image[2].

5 Classification

In order to perform the scene recognition task, two main classes of classifiers have been implemented: K-Nearest Neighbors (KNN) and Support Vector Machines (SVM).

5.1 K-Nearest Neighbors (KNN)

A K-Nearest Neighbors classifier taking as input features the normalized histograms of visual words has been implemented. The classifier assigns to each image in the test set the label of the majority class among its k nearest neighbors[3] in the training set.
In the simple case of $k = 1$, the label of the closest histogram in the training set is assigned to the test image. A slightly better result can be achieved by performing a linear search for the hyperparameter k over the range $[1, 50]$ using the average accuracy as assessment metric. For each value of k in the range, the accuracy of the corresponding KNN classifier is computed and stored. At the end, the value of k that maximizes the accuracy is selected and the performance of the resulting classifier is evaluated.

5.2 Support Vector Machines (SVM)

A series of multi-class Support Vector Machine (SVM) classifiers have been implemented following the "*one-vs-all*" strategy. For each possible class, a single binary classifier is trained and the final prediction is obtained by selecting the class with the highest confidence score. Each binary classifier is trained taking as input features one of the presented image representations and modified ground truth labels where the class of interest is labeled as $+1$ and all other classes are labeled as -1.

[2] E.g. cells positioned in the top region of the left side image or in the bottom left corner of the right side image in Figure 3.1.

[3] Notice k is the number of KNN neighbors, not to be confused with the number of clusters K.

Different kernels for these SVM classifiers have been tested and compared. Initially, the default radial basis function (RBF) kernel has been adopted. Its performance has then been compared with the generalized Gaussian kernel (with $\gamma = 1/2$) based on the χ^2 distance and the histogram intersection kernel in equation 5.1, both widely adopted in histogram comparison tasks.

$$k_{\chi^2}(\mathbf{x}, \mathbf{x}') = \exp\left(-\gamma \sum_i \frac{(x_i - x_i')^2}{x_i + x_i'}\right) \qquad k_{\cap}(\mathbf{x}, \mathbf{x}') = \sum_i \min(x_i, x_i') \tag{5.1}$$

For the *SPM* feature representation, the SVM classifiers only used the histogram intersection kernel as anticipated in the previous section and pyramid levels up to $L = 2$.

6 Results

The results obtained with all the classifiers, the two feature extraction methods and the different image representations are summarized in Table 6.1. The average accuracy over all the classes has been used as the main assessment metric for all the models. All the experimental results have been obtained by running the models on a machine equipped with an Intel® Core™ i7–8565U CPU @ 4.60 GHz and 8 GB of RAM.
The dummy classifier, that always predicts the most frequent class (*OpenCountry*) on the test set, has been used as baseline for comparison for the other models and the average accuracy it achieved is 10.39 %.
First of all, it's possible to notice that the results obtained using grid sampling consistently surpass the ones resulting from the SIFT detector. This confirms the idea presented in *Fei-Fei and Perona* [2] and *Lazebnik et al.* [3] for which grid sampling works better for scene classification tasks, as it allows to capture uniform regions such as sky, water, or road surfaces that might be crucial in discriminating between classes.
Looking at the performance of the different classifiers instead, the KNN ones surpassed the baseline, recording average accuracies between 40 % and 50 % with a small gain in performance given by the multiple neighbors approach over the single neighbor classifier.
Undoubtably, much better results have been achieved by the SVM classifiers with accuracies ranging from 50 % to 70 %. In particular, the adoption of specialized kernels significantly boosted the performance of the SVM classifiers, at least in the *HIST* and *TF-IDF* representations. Indeed, the 71.42 % accuracy obtained with the χ^2 kernel and the 70.62 % with the intersection one are in agreement with the findings of *Van Gemert et al.* [7] for the hard assignment case.
However, from the comparison of the different image representation techniques it emerges that, while little to no differences are observed between the *HIST* and *TF-IDF* representations, the implementation of the soft assignment techniques failed in improving the classification performances as reported in *Van Gemert et al.* [7], except for some isolated cases. In contrast with the results of the paper, the accuracies measured for these representations are mostly comparable, if not worse[4], with respect to the ones of the hard assignment cases.
A further level of assessment is also given by the confusion matrices computed for all the classifiers[5] and reported in appendix A3. These, not only confirm the superiority of the SVM classifiers with respect to the KNN ones, but also highlight the most difficult scenes to recognize. In detail, in most of the classifiers a noticeable (but understandable) confusion arises between the *Open Country* and *Coast* classes as well as among the *LivingRoom*, *Bedroom*, *Kitchen* and *Office* classes. Consistently, the hardest scene to classify has been the *Industrial* one, which matches with the results from literature.
By far, the best accuracy has been obtained by the SVM classifiers using the spatial pyramid matching approach. In this case, the combination of sampling features from a regular grid and adopting the *SPM* representation for the images allowed to achieve an accuracy of 75.54 %. This result is significantly better than the other classifiers and is a clear indication that the spatial information is crucial for the scene recognition task. The superiority of this approach is also confirmed by the almost completely diagonal confusion matrix (Figure A.7), in which only few of the previous misclassifications remain, mainly between the *Living Room* and *Bedroom* classes. Although higher than the other classifiers, the lowest accuracy is still measured for the *Industrial* class.

[4] The worst being 18.22 % for SIFT detected features using the χ^2 kernel.

[5] Only for *HIST* representation and grid sampling method as explained in appendix A3.

Classifier	**Image Representation**	**Feature Extraction**	
		SIFT detector	*Grid sampling*
Dummy Classifier	*HIST*	10.39 %	10.39 %
	TF-IDF	10.39 %	10.39 %
	KCB	10.39 %	10.39 %
	UNC	10.39 %	10.39 %
	PLA	10.39 %	10.39 %
1-NN	*HIST*	31.49 %	43.75 %
	TF-IDF	31.83 %	39.30 %
	KCB	34.81 %	42.08 %
	UNC	31.36 %	43.79 %
	PLA	37.62 %	49.75 %
k-NN	*HIST*	37.19 %	43.95 %
	TF-IDF	36.42 %	42.18 %
	KCB	41.84 %	44.29 %
	UNC	36.62 %	48.81 %
	PLA	42.81 %	56.25 %
SVM (RBF)	*HIST*	50.65 %	66.43 %
	TF-IDF	50.89 %	62.65 %
	KCB	52.16 %	58.79 %
	UNC	52.29 %	59.46 %
	PLA	46.47 %	67.00 %
SVM (χ^2)	*HIST*	52.66 %	71.42 %
	TF-IDF	52.03 %	70.22 %
	KCB	44.15 %	55.11 %
	UNC	50.22 %	55.38 %
	PLA	18.22 %	41.21 %
SVM ($\cap$)	*HIST*	50.95 %	70.62 %
	TF-IDF	50.79 %	68.91 %
	KCB	47.20 %	53.20 %
	UNC	46.97 %	56.95 %
	PLA	22.95 %	48.58 %
SVM (*Spatial Pyramid*)	*SPM*	—	75.54 %

Table 6.1: Average accuracies for all the implemented classifiers comparing feature extraction methods and image representations.

7 Conclusions

In conclusion, this study has shown the implementation of a Bag of Visual Words (BoW) classifier for scene recognition from the construction of a visual vocabulary by performing clustering on SIFT descriptors to the classification of images using machine learning techniques such as K-Nearest Neighbors (KNN) and Support Vector Machines (SVM).
From the results, it emerges that the SVMs have clearly outperformed the KNN classifiers in terms of accuracy, but the latter at least surpassed the dummy classifier performance used as baseline. In particular, the usage of specialized kernels for the SVM classifiers has also shown to be beneficial with respect to the default RBF kernel.
Concerning the worse results recorded for the soft assignment approach, these might be explained the different steps taken in this study with respect to the original paper. Specifically, in this work, *k-means* clustering has been used instead of the *radius based* clustering implemented by the original authors. Moreover, *Van Gemert et al.* [7] conducted k-fold cross-validation on the training set to find the optimal value of the σ, while in this study the value has been set to $\sigma = 200$ based on the results of the paper due to computational and time constraints. Additionally, the authors also performed clustering on a random subset of the extracted descriptors, but it's possible that the difference in the subset size and in representative sampled descriptors might have led to different results.
Ultimately, the best accuracy has been achieved by the SVM classifiers using the spatial pyramid approach proposed by *Lazebnik et al.* [3]. Although the final accuracy of 75.54 % doesn't quite match the one obtained by the authors of the original paper, the results are by far better than the other classifiers and this confirms the importance of adding spatial information to the classic BoW approach. Also in this case, the slight differences in the final results might be due to the different parameters used in the clustering process. Nevertheless, the results obtained are still satisfactory and confirm the validity of the BoW approach for the scene recognition task.

References

[1] Chih-Chung Chang and Chih-Jen Lin. Libsvm: A library for support vector machines. *ACM transactions on intelligent systems and technology (TIST)*, 2(3):1–27, 2011.

[2] L. Fei-Fei and P. Perona. A bayesian hierarchical model for learning natural scene categories. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 2, pages 524–531 vol. 2, 2005. `doi:10.1109/CVPR.2005.16`.

[3] S. Lazebnik, C. Schmid, and J. Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 2, pages 2169–2178, 2006. `doi:10.1109/CVPR.2006.68`.

[4] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[5] Peter J. Rousseeuw. Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics*, 20:53–65, 1987. URL: `https://www.sciencedirect.com/science/article/pii/0377042787901257`, `doi:10.1016/0377-0427(87)90125-7`.

[6] Marco Tallone. Bow classifier. URL: `https://github.com/marcotallone/bow-classifier`.

[7] Jan C. van Gemert, Jan-Mark Geusebroek, Cor J. Veenman, and Arnold W. M. Smeulders. Kernel codebooks for scene categorization. In David Forsyth, Philip Torr, and Andrew Zisserman, editors, *Computer Vision – ECCV 2008*, pages 696–709, Berlin, Heidelberg, 2008. Springer Berlin Heidelberg.

A Appendix

A1 Descriptors Distribution

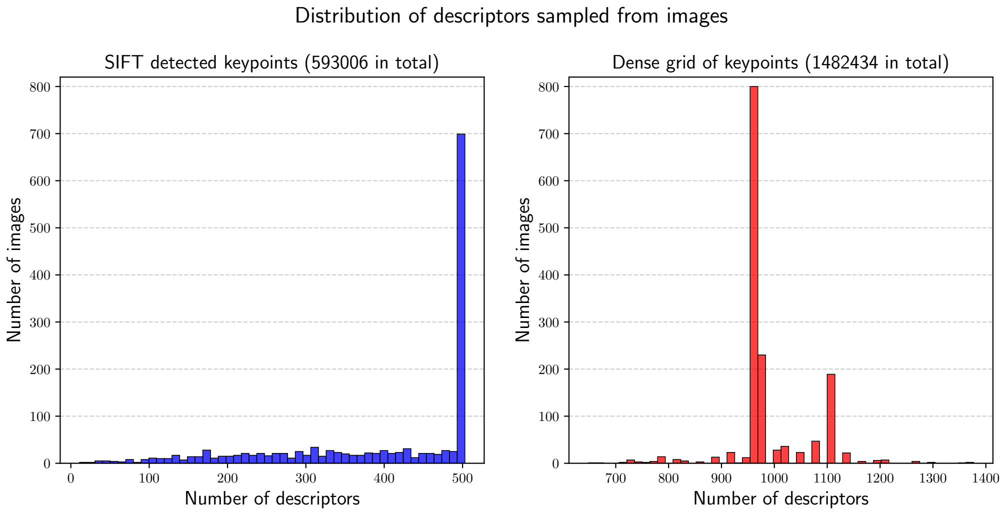

Figure A.1: Distribution of keypoints sampled from the train set using the SIFT detector (*blue*) and the dense grid approach (*red*). For the first method, the desired number of keypoints (500) has been detected in the majority of the images, while for the second method the number of keypoints per image is concentrated around 1000 with small differences due to the different sizes and aspect ratios of the images.

A2 Kernel Density Estimation

The Gaussian density estimator $K_\sigma(x)$ is defined as

$$K_\sigma(x) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{x^2}{2\sigma^2}\right) \tag{A.1}$$

Whereas, given the fact that both visual words and SIFT descriptors are 128-dimensional vectors, the Euclidean distance $D(w_i, x_j)$ between the i-th visual word w_i and the j-th descriptor x_j can be introduced as

$$D(w_i, x_j) = ||w_i - x_j|| = \sqrt{\sum_{k=1}^{128}(w_{ik} - x_{jk})^2} \tag{A.2}$$

Hence the soft assignment approach proposed by *Van Gemert et al.* [7] relies on the density estimator

$$K_\sigma(D(w_i, x_j)) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{D(w_i, x_j)^2}{2\sigma^2}\right) \tag{A.3}$$

A3 Confusion Matrices

Here the confusion matrices for all the classifiers are reported. These matrices refer to the case in which features have been extracted by a SIFT descriptor from keypoints sampled on a dense grid on the images and the input representation is the normalized histograms of visual words (*HIST*). This has been done to maintain the results consistent with the ones obtained using the spatial pyramid matching approach and to be able to compare the performances of the different classifiers. However, similar matrices can be computed in the other cases as well.

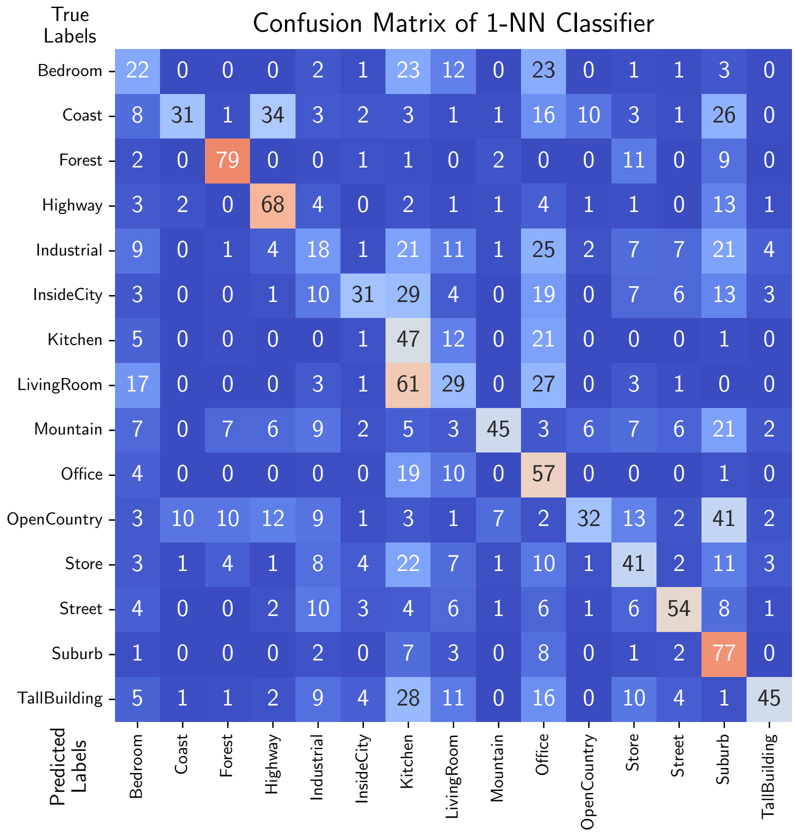

Figure A.2: Confusion matrix for the 1-NN classifier.

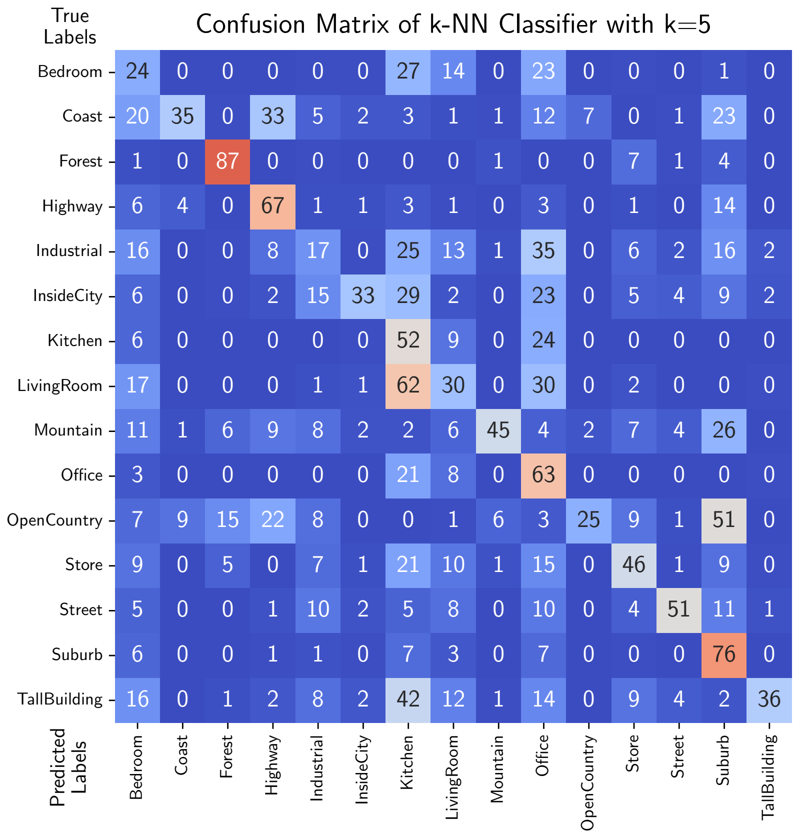

Figure A.3: Confusion matrix for the k-NN classifier with $k = 5$.

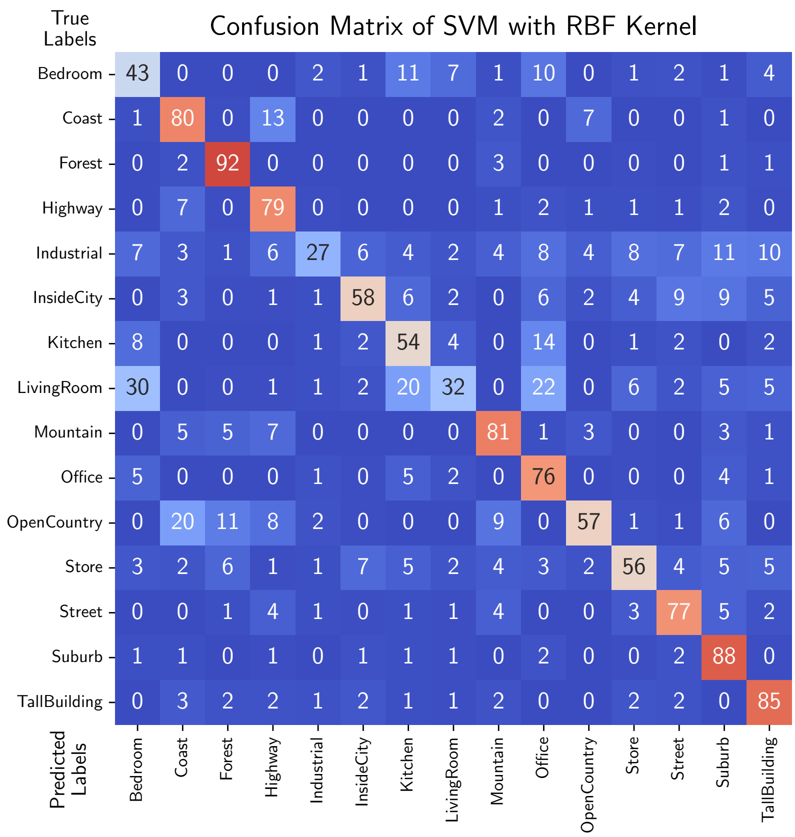

Figure A.4: Confusion matrix for the SVM classifier with RBF kernel.

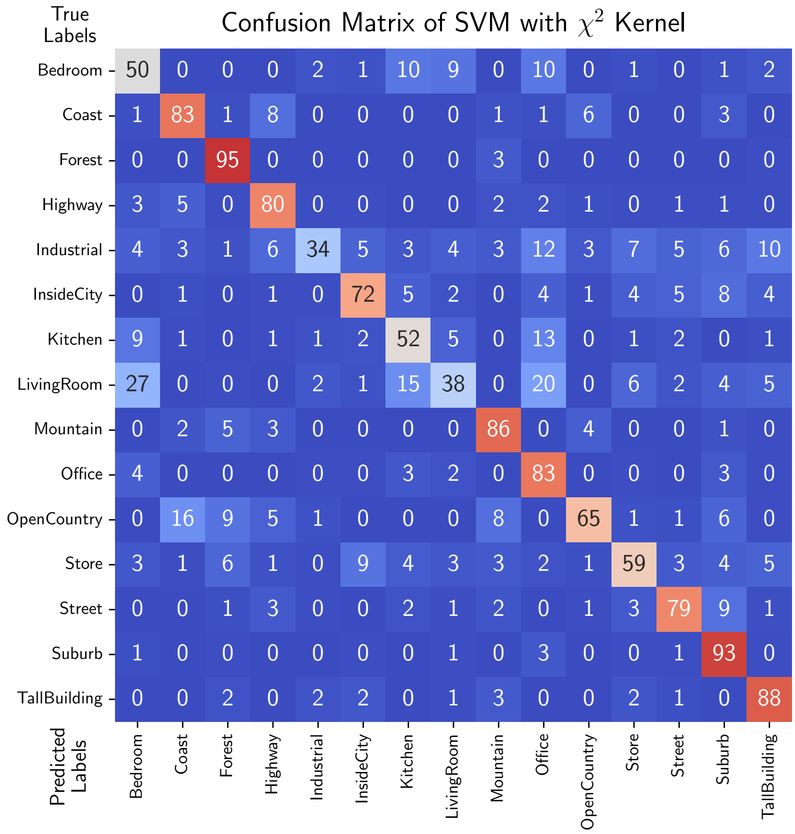

Figure A.5: Confusion matrix for the SVM classifier with χ^2 kernel.

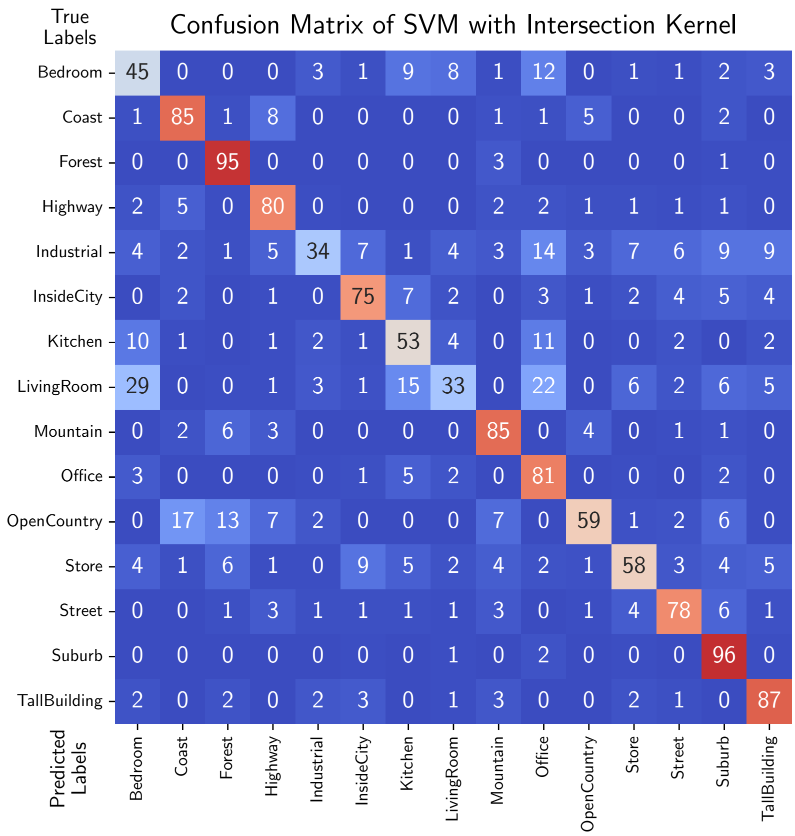

Figure A.6: Confusion matrix for the SVM classifier with histogram intersection kernel.

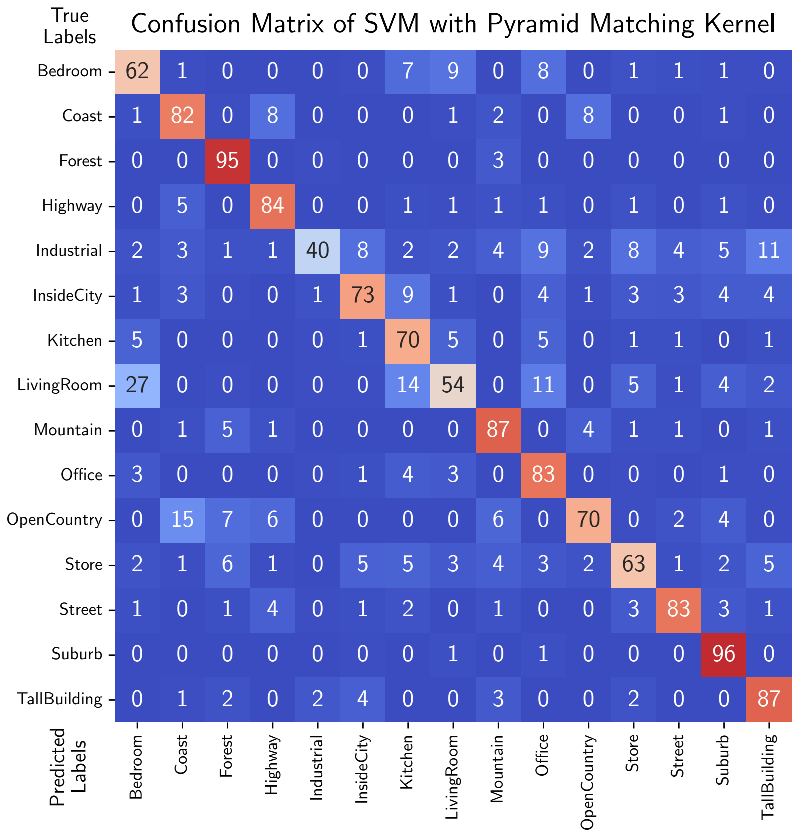

Figure A.7: Confusion matrix for the SVM classifier with spatial pyramid matching approach.