MPhil in Advanced Computer Science

2018-2019
Coursework submission
Cover sheet

Student name……Zhuoni Jie……

College……Wolfson College……

CRSID……zj245……

Submit to Graduate Education
Date of submission

Module name Computer Vision

Module Code No. L248

Assignment No. Exercise 2

Assignment mark/grade

Assessor comments

GE Records……..

CRSID……zj245……

Module No……L248……

Assignment No……Exercise 2……

GE stamp date and initials

Computer Vision: Exercise 2

Zhuoni Jie
zj245@cam.ac.uk

February 16, 2019

1 Dataset

In this project, we use a provided subset of "Places" dataset described by Zhou et al ([1]), which includes a training set, testing set, and class labels.

Places originally contain more than 7 million images from 476 place categories, making it the largest image database of scenes and places so far and the first scene-centric database competitive enough to train algorithms that require huge amounts of data, such as CNNs. The subset we use here contain 4 place categories: bridge, coast, mountain, and rain forest. Each category has 165 training images and 35 testing images, leading to 800 images in total. Images are resized to 300×300 pixels for feature extraction and analysis.

2 Feature Extraction

2.1 Histogram of Oriented Gradients (HOG)

The Histogram of Oriented Gradients (HOG) ([2]) is a popular feature for describing appearance in computer vision and image processing that has been successfully used in many recognition and detection tasks. HOGs, which are the distribution of intensity gradients or edge directions, describe the local object appearance and shape within an image. A HOG descriptor counts the number of oriented gradient occurrences in a dense grid of uniformly spaced cells. These occurrences are represented as a histogram for each cell normalized in a larger block area. Since it operates on local cells, a HOG is invariant to geometric and photometric transformations, except for object orientation. Such changes would only appear in larger spatial regions.

For HOGs, we used 16×16 pixel cells with 8 gradient orientations and a block size of 1×1 cells, leading to a 2592-dimensional HOG descriptor as visualized in Figure 2.

2.2 Scale Invariant Feature Transform (SIFT)

The Scale-Invariant Feature Transform (SIFT) ([3]) is a method for extracting distinctive invariant features from images that can be used to perform reliable matching between different

Figure 1: The original image

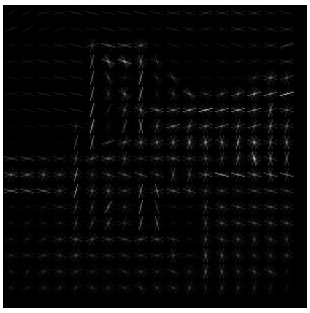

Figure 2: The HOG descriptor

Figure 3: The SIFT descriptors

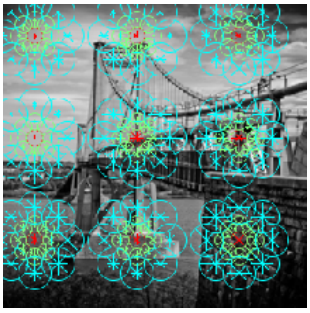

Figure 4: The DAISY descriptors

views of an object or scene. The method transforms an image into a large collection of local feature vectors (local descriptors called SIFT keys), which are invariant to image scale and rotation, and are shown to provide robust matching across a substantial range of affine distortion, change in 3D viewpoint, addition of noise, and change in illumination. The features are highly distinctive.

For the training set, I got 586297 sift descriptors in total, which are clustered into 100 clusters using Bag-of-Visual-Words (BoVW) approach. A sample image with SIFT descriptors is shown as in Figure 3.

2.3 DAISY

DAISY (4) is a local descriptor inspired from earlier local features such as SIFT and GLOH (5), but can be computed much more efficiently for densematching purposes. DAISY's peed increase comes from replacing the weighted sums of gradient norms by convolutions of the gradients in specific directions with several Gaussian filters, which gives the same kind of invariance as the SIFT and GLOH histogram building, but is much faster for dense-matching purposes and allows the computation of the descriptors in all directions with little overhead.

For parameters, I use a step of 32 and a radius of 32, getting 855360 DAISY descriptors in total for the training set. Similarly as SIFT, DAISY descriptors are also clustered into 100 clusters using BoVW. A sample image with DAISY descriptors is shown as in Figure 4. For visualization purposes, this sample uses DAISY with a step of 100 and a radius of 32.

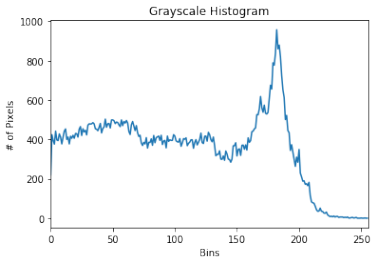

Figure 5: The grayscle histogram

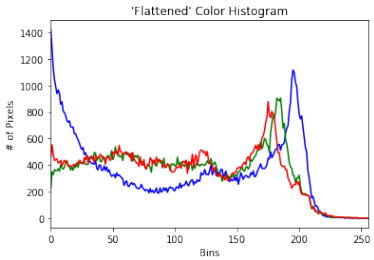

Figure 6: The RGB color histogram

2.4 RGB histogram

The RGB histogram has been widely used to extract global color features for color object recognition. Ballard and Swain ([6]) described objects by their color histograms. The RGB histogram is a combination of three 1D histograms based on the R, G, and B channels of the RGB color space. This histogram originally possesses no invariance properties. Moreover, to obtain invariance with respect to lighting geometry the use of normalized RGB histograms was advocated. This method remained however variant with respect to illuminant changes. To tackle this problem Funt and Finlayson ([7]) proposed an illuminant invariant indexing method, which was however variant with respect to lighting geometry. Finlayson et al. ([8]) combined these two theories and proposed a indexing method which is both invariant to shading and illuminant changes.

Sample images showing grayscale histogram and RGB histogram are presented in Figure 5 and Figure 6.

2.5 Bag-of-Visual-Words (BoVW) Approach

Based on keypoints extracted as salient image patches, an image can be described as a "bag of visual words" and this representation has been used in scene classification ([9]). From the past few years, the bag-of-features approach has been actively used in numerous computer vision applications and has shown a solid performance for image annotation, classification and retrieval. Avni et al. ([10]) proposed a bag of SIFT feature based x-ray image retrieval system and achieved the top performance on IRMA project library ([11]). Caicedo et al. ([12]) presented a study depicting the systematic evaluation of many representations which resulted from the bag of features technique for classification of pathology images.

I make use of the SIFT and DAISY descriptors extracted from all the training images to form a bag-of-visual-words by clustering them using Mini-Batch KMeans, getting 100 clusters. Then each SIFT and DAISY descriptor of the image is assigned to one of the 100 clusters. In this way, I obtain a histogram of length 100, which is normalized with L1 normalization. The obtained BoVW features are then feed to the classifier to get the classification result.

2.6 Concatenation

I fuse local information with global appearance information, by concatenating DAISY descriptors to a HOG descriptor. I also extend feature descriptors with color information, by concatenating a color descriptor to DAISY and HOG. This leads to DAISY + HOG and DAISY + HOG + RGB histogram descriptors.

3 Classifiers

Support Vector Machines (SVM) (13) are supervised learning models that utilize hyperplanes and support vectors to analyze data for classification and regression. Given labeled training data, the SVM algorithm outputs an optimal hyperplane which optimally categorizes new examples. The original idea of support vector network was implied for the situation that training data was separable by a hyperplane without error. Later, Cortes and Vapnik introduced the notion of softmargins such that a minimal subset of error in the training data is permit table, allowing the remaining part of the training data to be separated by constructing an optimal separating hyperplane (14). SVM's remarkably robust performance with respect to sparse and noisy data making is making it systematically used in a variety of applications.

SVM is known for its kernel trick to handle nonlinear input spaces. The main function of the kernel is to transform the given dataset input data into the required form. There are various types of functions such as linear, polynomial, and radial basis function (RBF). Here in this project I use linear kernel and RBF kernel SVMs for classification and comparison.

4 Results

4.1 Feature Performance

I conducted the experiments on a macOS Mojave with 2.6 GHz Intel Core i7 GPU and 16 GB 2400 MHz DDR4 memory. The feature extraction and model training is implemented in Python 3.7 with libraries including OpenCV (15), scikit-learn (16), and scikit-image (17). The total feature extraction time is 02:01. The results using implementing linear SVM ($C = 1$) are summarized in Figure 7.

	HOG			SIFT			DAISY			RGB Histogram			HOG + DAISY			HOG + SIFT + DAISY + RGB Hist		
	precision	recall	f1	precision	recall	f1	precision	recall	f1	precision	recall	f1	precision	recall	f1	precision	recall	f1
micro avg	0.76	0.76	0.76	0.74	0.74	0.74	0.38	0.38	0.38	0.61	0.61	0.61	0.83	0.83	0.83	0.86	0.86	0.86
macro avg	0.76	0.76	0.75	0.73	0.74	0.73	0.44	0.38	0.35	0.61	0.61	0.61	0.84	0.83	0.83	0.86	0.86	0.86
weighted avg	0.76	0.76	0.75	0.73	0.74	0.73	0.44	0.38	0.35	0.61	0.61	0.61	0.84	0.83	0.83	0.86	0.86	0.86
overal accuracy	0.76			0.74			0.38			0.61			0.83			0.86		

Figure 7: Experiment results using different features with linear SVM

From the table we can see for a single feature, HOG achieves the best scene recognition performance here with an overall accuracy of 76%, while SIFT ranks second with an overall accuracy of 74%. The poorest result is generated by DAISY alone, getting only 38% overall accuracy. For hybrid features, using the four features together generates the best result with

	bridge	coast	rainforest	mountain
bridge	16	6	6	7
coast	2	31	0	2
rainforest	2	0	28	5
mountain	1	2	1	31

Figure 8: HOG

	bridge	coast	rainforest	mountain
bridge	20	12	0	3
coast	14	20	0	1
rainforest	15	1	4	15
mountain	15	9	2	9

Figure 9: DAISY

	bridge	coast	rainforest	mountain
bridge	28	3	2	2
coast	5	22	1	7
rainforest	4	0	30	1
mountain	3	6	3	23

Figure 10: SIFT

	bridge	coast	rainforest	mountain
bridge	14	6	5	10
coast	6	24	0	5
rainforest	2	2	28	3
mountain	6	8	1	20

Figure 11: RGB Hist

	bridge	coast	rainforest	mountain
bridge	24	5	2	4
coast	1	32	0	2
rainforest	0	1	30	4
mountain	1	1	3	30

Figure 12: HOGDAISY

	bridge	coast	rainforest	mountain
bridge	28	2	2	3
coast	4	31	0	0
rainforest	0	1	32	2
mountain	3	2	0	30

Figure 13: All features

an accuracy of 86%, showing the promising performance of combining global, local, and color information of images. We should also notice that only combining HOG and DAISY can get good results close to the best one, with and overall accuracy of 83%, significantly improving both features' single performance. This shows the benefits of combining global and local descriptors.

I also calculated the confusion matrix to analyze the accuracy of each classification, as presented below.

From the confusion matrix we can see that different features have varied recognition abilities across the four categories. HOG, SIFT, and RGB Histogram descriptors all perform relatively well on detecting the rainforest category, while DAISY does extremely poor job in recognizing rainforest scenes. RGB Histogram feature has nice performance on recognizing coast, rainforest, and mountain scenes, probably because these scenes have very representative colors that contain much information, while bridge scenes are highly influenced by specific environments and illumination and are thus highly varied in terms of color distribution. HOG + DAISY feature and HOG + DAISY + SIFT + RGB Hist feature perform extremely well in coast, rainforest, and mountain categories, while relatively weaker in bridge recognition. This is probably due to bridges' huge variation in their shapes, textures, colors, illumination, and other environmental factors.

4.2 Misclassified Images

As an example, the bridge image below is misclassified as mountain using a HOG-trained model. This is probably because of the unusual smooth curving shape of the bridge, which resembles that of a mountain's.

5 Discussion

5.1 Generalizability / Extended Evaluation

After obtaining the results using a linear SVM, I changed the kernel to RBF and performed a grid search for discovering the optimal hyper-parameters for the RBF SVM, getting the

Figure 14: A misclassified bridge image, which is classified as mountain when using the HOG-trained model

following results for model using full features as shown in Figure 15. The best parameters selected are $C = 100.0$ and $\gamma = 0.1$ with a 5-fold cross-validation accuracy score of 0.87.

	bridge	coast	rainforest	mountain
bridge	27	2	2	4
coast	3	32	0	0
rainforest	1	0	32	2
mountain	4	1	0	30

Figure 15: The confusion matrix result of RBF kernel SVM model

This model with more complicated kernel generates results similar to using linear kernel, gaining and accuracy score of 86%. This indicates that linear SVM is already good to capture their relationships and generate nice recognition results.

I also use the four new pictures (Figure 16 - 19) to evaluate the trained full feature RBF SVM model's performance, and the trained model correctly recognized the four different scenes.

5.2 Future Work

Future work can be done to possibly improve the recognition performance. First, we can empirically determine the best visual vocabulary size (K value in KMeans clustering) through cross-validation. Second, the hyper-parameters of HOG and DAISY descriptors can be further tunned, and Principle Component Analysis can also be used to reduce the feature dimensions

Figure 16: Bridge

Figure 17: Mountain

Figure 18: Rainforest

Figure 19: Coast

for better results. Third, other color discriptors such as Opponent histogram, Hue histogram, color moments and moment invariants, and color SIFT descriptors can be implemented to gain better properties, as discussed in (18). In addition to the standard BoVW pipeline, other methods like Sparse Coding, max pooling, spatial encoding (Spatial Pyramid Matching) modules can be also used. Apart from SVM, other more complex machine learning models like CNNs can also be implemented for classification.

Appendix

The code written in python can be accessed here:
https://nbviewer.jupyter.org/github/karlinjzn/jiezn/blob/master/L248_Exercise2.ipynb

References

[1] Zhou, Bolei, et al. "Learning deep features for scene recognition using places database." Advances in neural information processing systems. 2014.

[2] Dalal, Navneet, and Bill Triggs. "Histograms of oriented gradients for human detection." international Conference on computer vision & Pattern Recognition (CVPR'05). Vol. 1. IEEE Computer Society, 2005.

[3] Lowe, David G. "Distinctive image features from scale-invariant keypoints." International journal of computer vision 60.2 (2004): 91-110.

[4] Tola, Engin, Vincent Lepetit, and Pascal Fua. "Daisy: An efficient dense descriptor applied to wide-baseline stereo." IEEE transactions on pattern analysis and machine intelligence 32.5 (2010): 815-830.

[5] Mikolajczyk, Krystian, and Cordelia Schmid. "A performance evaluation of local descriptors." IEEE transactions on pattern analysis and machine intelligence 27.10 (2005): 1615-1630.

[6] Ballard, Dana H. "Generalizing the Hough transform to detect arbitrary shapes." Pattern recognition 13.2 (1981): 111-122.

[7] Funt, Brian V., and Graham D. Finlayson. "Color constant color indexing." IEEE transactions on Pattern analysis and Machine Intelligence 17.5 (1995): 522-529.

[8] Finlayson, Graham D., Bernt Schiele, and James L. Crowley. "Comprehensive colour image normalization." European conference on computer vision. Springer, Berlin, Heidelberg, 1998.

[9] Yang, Jun, et al. "Evaluating bag-of-visual-words representations in scene classification." Proceedings of the international workshop on Workshop on multimedia information retrieval. ACM, 2007.

[10] Avni, Uri, et al. "X-ray categorization and retrieval on the organ and pathology level, using patch-based visual words." IEEE Transactions on Medical Imaging 30.3 (2011): 733-746.

[11] Lehmann, Thomas M., et al. "Content-based image retrieval in medical applications." Methods of information in medicine 43.04 (2004): 354-361.

[12] Caicedo, Juan C., Angel Cruz, and Fabio A. Gonzalez. "Histopathology image classification using bag of features and kernel functions." Conference on Artificial Intelligence in Medicine in Europe. Springer, Berlin, Heidelberg, 2009.

[13] Chang, Chih-Chung, and Chih-Jen Lin. "LIBSVM: a library for support vector machines." ACM transactions on intelligent systems and technology (TIST) 2.3 (2011): 27.

[14] Cortes, Corinna, and Vladimir Vapnik. "Support-vector networks." Machine learning 20.3 (1995): 273-297.

[15] OpenCV: https://opencv.org/

[16] Pedregosa, Fabian, et al. "Scikit-learn: Machine learning in Python." Journal of machine learning research 12.Oct (2011): 2825-2830.

[17] Van der Walt, Stefan, et al. "scikit-image: image processing in Python." PeerJ 2 (2014): e453.

[18] Van De Sande, Koen, Theo Gevers, and Cees Snoek. "Evaluating color descriptors for object and scene recognition." IEEE transactions on pattern analysis and machine intelligence 32.9 (2010): 1582-1596.