TRAFFIC ANALYSIS WITHOUT MOTION FEATURES

Zhiming Luo[1,2,3] *Pierre-Marc Jodoin*[3] *Shao-Zi Li*[1,2] *Song-Zhi Su*[1,2*]

[1] School of Information Science and Technology, Xiamen University, Xiamen, 361005

[2] Fujian Key Laboratory of the Brain-like Intelligent Systems, Xiamen, 361005

[3]Université de Sherbrooke, Sherbrooke, Québec, Canada

ABSTRACT

In this paper, we investigate the possibility of monitoring traffic without using any motion features. The goal of our system is to process videos with ultra-low frame rate, *i.e.* videos for which reliable motion features cannot be computed. In this work, we investigate how 2D spatial features combined with a machine learning method can assess traffic conditions such as fluid traffic, dense traffic, and traffic jam. The underlying hypothesis that we ought to validate is that traffic images are heavily characterized by their 2D spatial textures. In that perspective, we tested different 2D texture features and machine learning methods to see how accurate such an approach can be. We also performed a regression on the image descriptor in order to estimate traffic density.

Experimental results obtained on the UCSD traffic dataset reveal that our approach generalizes well to various weather and lighting conditions. It even outperforms state-of-the-art traffic analysis methods relying on spatio-temporal features.

Index Terms— Traffic analysis, SVM, codebook, convolutional neural network.

1. INTRODUCTION

Video surveillance systems are often used for traffic monitoring and to characterize traffic load. Knowing in real time when the traffic is fluid or when it jams is a key information to help authorities re-route traffic and thus reduce congestion.

As far as we know, current traffic monitoring systems all rely on spatio-temporal features. These features are typically used to segment and track vehicles and/or to compute traffic flow [1, 2, 3, 4, 5, 6, 7, 8, 9]. These informations are generally used to estimate traffic speed, count the number of vehicles on the road and recover motion trajectories.

But these traffic monitoring systems share a fundamental limitation: for it to work, they need high frame rate videos with stable environmental conditions. Without these conditions, reliable motion features cannot be extracted thus leading to corrupted results. Although a bit old, the survey by Kastriaki et al. [10] describes well these limitations.

Unfortunately, low frame rate videos are very common as many large-scale camera networks cannot stream and store high-frame rate videos gathered by hundreds (if not thousands) of cameras. This is mainly due to bandwidth and storage limitations. Instead, cameras often send to the server one frame every 2 or 3 second (if not less). It is also the case for IP cameras transmitting over a WIFI network. The limited bandwidth makes it hard to have more than 2 frames per second. One might also think of a non-stationary low-altitude UAV which can only take a limited number of images of a given road when it flies over it. In such case, one can only analyze traffic based on a limited number of unregistered images.

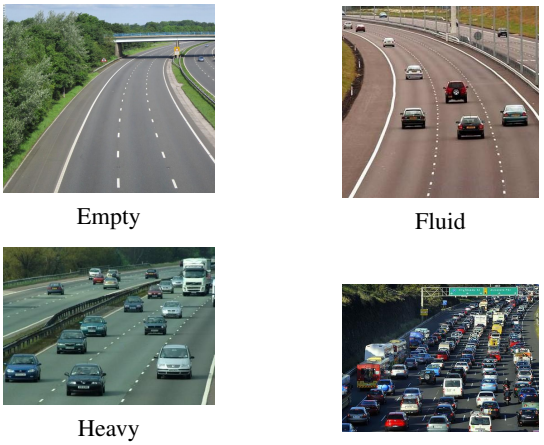

Fig. 1. Images taken from our Motorway Dataset and showing different traffic density.

In this paper, we explore the possibility of using only spatial texture features to classify traffic. Figure 1 shows examples of four different traffic configurations. Since no such work has been done in the past, we tested different texture features with different machine learning methods and show that modern pattern recognition methods can outperform state-of-the-art video-based traffic analysis methods.

2. RELATED WORKS

Traffic classification methods can be roughly divided into two groups: the vehicle-oriented ones and those relying on a holistic approach.

Vehicle-based methods assess traffic activity by tracking every moving vehicle on the road. For these methods, vehicles are first localized with a background subtraction method [1, 11, 12] or with moving feature keypoints [7, 9] and then followed with a tracking method such as mean-shift,

a Kalman filter tracker, or a background-subtraction-based tracking method [3, 9]. Resulting tracks are bundled together in order to identify traffic lanes and/or the average traffic speed and/or the average traffic density [7, 3, 13]. Note that traffic density can also be obtained by simply counting the number of detected cars [7, 8].

Unfortunately, tracking simultaneously a large number of moving objects is still a challenge nowadays. As a solution, other methods focus on understanding the traffic flow from a macroscopic (or holistic) point of view and thus avoid tracking. The global representation of a scene is obtained by accounting for spatio-temporal features other than tracking and background subtraction. For example, Derpanis and Wildes [6] use 3D spatio-temporal filters based on the third derivative of a Gaussian function. Porikli and Li [14] use features from the MPEG compressed domain, *i.e.* the DCT coefficients and the motion vectors used for compression. Lee and Bovik [4] accumulates optical flow over time in order to get a temporally average vector field which is then used to classify traffic flow. Other methods [5, 15] analyze traffic flow as a dynamic texture classification problems. Chan and Vasconcelos[5] defined an autoregressive stochastic process over the spatial and temporal components while Derpanis and Wildes [15] associates different distribution to different traffic flow status in spatial-temporal orientation domain.

All these methods end up classifying the traffic into a certain number of traffic classes (typically low, medium, heavy, and traffic jam). To do so, various machine learning methods have been used such as K-nearest neighbors, SVM, and decision trees to name a few.

From our perspective, the main limitation of these previous methods is their need for high-frame-rate videos to compute reliable motion features. As a consequence, none of these methods can be used to process low-frame rate videos, *i.e.* videos with less than 1 or 2 fps.

The reader should note that although some method uses spatial features to process road scenes, to our knowledge none of them have been used to assess traffic condition. For example, Ess *et al.* [16] process video frames acquired from the front view of a moving vehicle to determined the environment in front of the car (*e.g.* parking, highway, side road, tunnel, etc.). Similarly, Sikiric *et al.*[17] implemented a bag of visual words pipeline to segment road images. As for Ess *et al.* [16], the resulting segmentation map is used to identify in which scene the car is.

3. PROPOSED METHOD

The goal of this paper is to validate the hypothesis that a well-designed and well-trained 2D pattern recognition system can correctly assess traffic condition without using motion features. In that perspective, we tested different SVM methods together with four different codebook visual descriptors as well as two convolutional neural network (CNN) features.

3.1. Codebook Descriptors

Image classification systems always involve a training and testing phase. The training phase is often carried out in three steps: 1) extract local image features of all training images (*e.g.* HOG or SIFT), 2)generate a codebook from these features (*e.g.* k-means, GMM) and 3) encode local image features into visual descriptors and train a classifier (*e.g.* SVM). As for the testing phase, it typically follows three steps : 1) extract local features from the test image, 2) encode a visual descriptor vector with the local features and the codebook, and 3) classification of the visual descriptor vector [18].

In this paper, we tested four different visual descriptors, namely a bag of visual words (BOVW), Vector of Locally Aggregated Descriptors (VLAD) [19], improved Fisher Vector (IFV) [20] and Locality-constrained Linear Coding (LLC) [21]. For all these methods, we used the same set of dense 128-dimension SIFT features $(\vec{x}_1, \vec{x}_2, ..., \vec{x}_N)$.

BOVW: starting from the SIFT features, we first compute a dictionary of K visual words $(\vec{\mu}_1, \vec{\mu}_2, ..., \vec{\mu}_k)$ following a k-means method. Then, a histogram of visual words is computed for each image. The resulting histogram is a K-dimension visual descriptor which contains the number of times a visual word has been seen in an image. The histogram vector is then used for both training and testing.

VLAD: as for BOVW, we first learn a dictionary of K visual words with K-means and assign each SIFT descriptor $\vec{x}_i$ to its nearest visual word $\vec{\mu}_k = NN(\vec{x}_i)$. Then, VLAD accounts for the residual distance to the nearest visual word:

$$v_{jk} = \sum_{\vec{x}_i \text{ such that } \mathrm{NN}(\vec{x}_i)=\vec{\mu}_k}^{K} (x_{ij} - \mu_{kj}). \quad (1)$$

Note that v is a $128 \times K$ matrix where 128 is the SIFT vector size and K the total number of visual words. Matrix v is then concatenated into a 1D $128 * K$ vector and used for training and testing.

LLC: starting from the same dictionary than VLAD and BOVW, LLC computes a visual descriptor by accounting for sparse coding. That is, instead of assigning each SIFT feature $\vec{x}_i$ to its X nearest visual words μ_k, it assigns a *soft* weight to the nearest visual words (in our case, we use X=5). The weights are computed with the following sparse coding scheme :

$$\arg\min_{C} \sum_{i=1}^{N} ||\vec{x}_i - BC_i||^2 + \gamma ||\vec{d}_i \odot C_i||^2 \quad (2)$$

where C is a $N \times K$ weight matrix, B is the K-dimensional codebook, $\odot$ is an element-wise multiplicator and $\vec{d}_i$ is a distance vector to the nearest visual words. Once calculated, the columns of C are summed in order to get a 1D vector which we then use for training and testing.

IFV: here, visual words are expressed with a Gaussian mixture model (GMM), restricted to diagonal variance matrices. Then, for each visual word $\vec{v}_k$ consider the mean and covariance deviation vectors:

$$v_{kj} = \frac{1}{N\sqrt{\pi_k}} \sum_{\vec{x}_i} q_{ik} \frac{x_{ij} - \mu_{kj}}{\sigma_{kj}} \quad (3)$$

$$w_{kj} = \frac{1}{N\sqrt{2\pi_k}} \sum_{\vec{x}_i} q_{ik} \left[\left(\frac{x_{ij} - \mu_{kj}}{\sigma_{kj}} \right)^2 - 1 \right] \quad (4)$$

where π_k is the prior probability of visual word k, q_{ik} the posterior probability that vector $\vec{x}_i$ is an element of the visual word k and σ_{kj} is a standard deviation. The $\vec{v}_k$ and $\vec{w}_k$ vectors are then stacked up into a $2D \times K$ matrix which we concatenate into a 1D descriptor for training and testing.

3.2. Pre-Trained CNN Features

Since traffic scenes may contain a wide variety of structured and unstructured texture from urban to country scenes, we also tested CNN features trained on imageNet, one of the largest image dataset currently available. The architecture of our CNNs starts from an image which is fed to a cascade of convolutional layers, the last layer being a fully connected softmax layer whose output gives a log probability for each class. In our case, we use as features the output of the last layer right before the softmax. In this paper, we tested features from two different pre-trained deep models, the Caffe reference model [22] as well as the VGG model [23]. The Caffe's model consists of five convolutional layers and three fully connected layer while the VGG's model has a similar architecture but with different filter sizes and stride steps. For both models, the resulting feature vector contains 4096 dimensions.

3.3. SVM

Once the image descriptors are computed, a linear SVM as well as 3 different kernel SVM *(Hellinger, Chi2 and HIK)* have been trained for classification. The equation of these kernels are $k_{hellinger}(\vec{v}, \vec{v}') = \sum_{i=1}^{N} \sqrt{v_i v_i'}$, $k_{chi2}(\vec{v}, \vec{v}') = \sum_{i=1}^{N} \frac{2v_i v_i'}{v_i + v_i'}$ and $k_{hik}(\vec{v}, \vec{v}') = \sum_{i=1}^{N} min(v_i, v_i')$ where $\vec{v}$ and $\vec{v}'$ are image descriptors.

4. EXPERIMENT RESULTS

4.1. Experimental Setup

Three datasets have been used to gauge performances. The first one is the UCSD traffic dataset [5] which has been widely used to assess traffic activity. This dataset contains 254 highway video sequences, all filmed by the same camera. These videos contains light, heavy and traffic jams filmed at different periods of the day and under different weather conditions. The UCSD videos have a 320*240 resolution and come with a frame rate of 10 fps.

We also built our own dataset (the *Motorway* dataset, cf. Fig. 1) which contains 400 images all taken from different highway cameras deployed in UK. These images were divided into 4 categories, namely *empty, fluid, heavy* and *jam*.

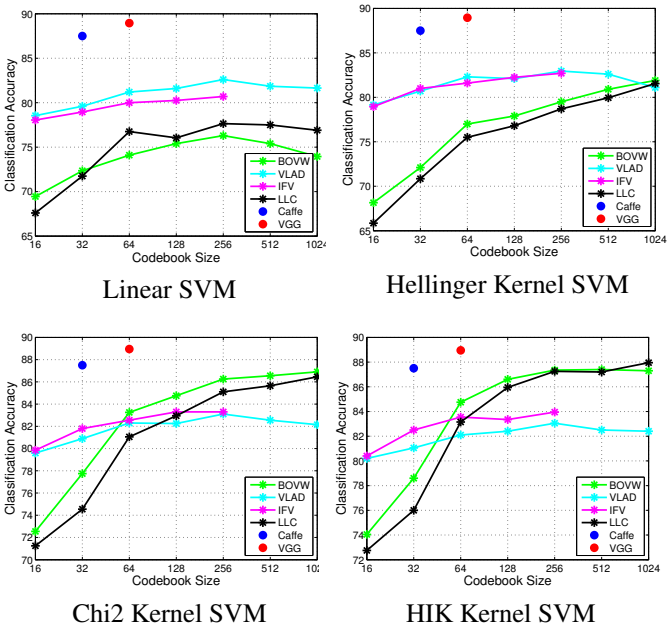

Fig. 2. Performance comparison on the Motorway dataset of all our methods testes on different codebook sizes and the CNN descriptors.

In order to analyze some more traffic videos, we took highway videos from the ChangeDetection.net dataset[24]. The main advantage with these videos is that they come with a pixel-accurate groundtruth which allows to precisely measure traffic density.

We used the VLFeat toolbox[25] to extract SIFT features on a dense grid with a step size of 4 pixels under three different patch sizes 16*16, 24*24 and 32*32. All codebook and CNN features were also *L2* normalized to unit length and the liblinear toolbox was used for training and testing. We used a fix SVM slack parameter of $C = 1$.

4.2. Classification Results

4.2.1. Motorway Dataset

For this first set of experiments, we trained on 200 randomly selected images taken from the *Motorway* dataset and tested on the remaining 200 images. We repeated that experiment 10 times and kept the mean classification accuracy.

Figure 2 shows results of different codebook descriptors, different codebook sizes as well as the two CNN descriptors. As one can see, as the number of visual words increases, the overall accuracy of BOVW and LLC also increases. As for VLAD and IFV, since they account for very high dimension descriptors, the codebook size and the type of kernel have a marginal influence on the accuracy. Results for BOVW and LLC are also much better with kernel SVM than with linear SVM. That being said, the CNN features outperform the codebook descriptors for all SVM implementations.

Table 1. Classification accuracy on the UCSD dataset for 4 different image descriptors and 4 SVM kernels.

	BOVW(64)	BOVW(128)	Caffe	VGG
Linear	94.9/94.9	96.1/**96.9**	95.7/95.3	96.1/96.1
Hellinger	94.1/94.1	94.1/94.9	**96.5**/94.9	95.7/**96.5**
Chi2	**96.5**/94.5	95.7/96.5	**96.5**/95.3	94.9/96.1
HIK	95.7/96.1	**96.5**/95.7	**96.5**/95.3	94.5/96.5

4.2.2. UCSD Dataset

The training and testing methodology used for this dataset is the one provided by UCSD. We thus trained and tested four times our methods, each times with 75% of randomly selected images for training and 25% for testing. Since we focus on low frame rate videos, we only processed 1 frame out of 6 (~ 1.7 *fps*). Once every frame of a video has been processed, the class label for that video is obtained with a majority vote. We also tested our method with and without the region of interest (ROI) provided with the dataset. The ROI is a 180x180 windows centered on the highway.

Table 1 contains the classification accuracy obtained with our methods. Note that due to space limitation, we only report results for BOVW (with codebook size 64 and 128) and the two CNN features, those for which we got the best results on the *Motorway* dataset. The left values are accuracy obtained after processing the original images while the right ones are those obtained after processing the 180x180 ROI. Results with and without ROI are relatively similar although slightly in favor of BOVW(128) + ROI with linear SVM. The confusion matrix for that method is shown in Table 2.

Table 3 shows results obtained by other methods all relying on motion features (results come from the original papers). As can be seen, results obtained with BOVW and the CNN features outperform all previous methods thus showing that motion features are not mandatory for assessing traffic.

Table 2. Confusion matrix on the UCSD dataset (BOVW 128 + ROI with linear SVM)

		Predicted		
		Heavy	Medium	Light
True	Heavy	40	4	0
	Medium	2	42	1
	Light	0	1	164

Table 3. Results of various methods on the UCSD dataset

Method	Accuracy	Method	Accuracy
Chan[5]	94.5%	Asmaa[12]	95.3%
Sobral[11]	94.5%	Caffe	**96.5%**
Derpanis[15]	95.3%	VGG (ROI)	**96.5%**
Riaz[9]	95.3%	BOVW(128 ROI)	**96.9%**

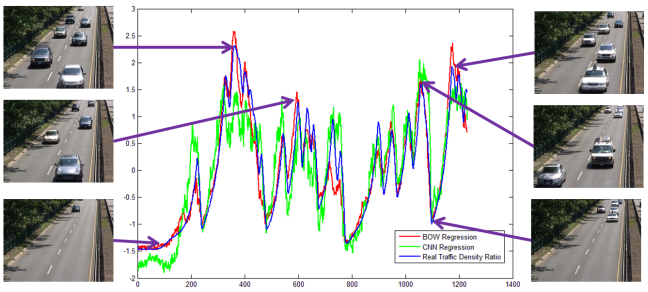

Fig. 3. Traffic Density Estimation.

4.3. Traffic Density Estimation

The goal for this third set of experiments is two fold. First, see how accurate our method can be at estimating traffic density (that is the percentage of the road occupied by a car) and second, see if a model trained on one dataset generalizes well to another dataset.

In that perspective, we trained our models on the Motorway dataset and tested 3 highway videos from ChangeDetection.net. Since these videos provide pixel-accurate groundtruth, the density at each frame was calculated as follows: $\frac{NF}{NR}$ where NF is the number of foreground pixels and NR is the number of background road pixels. In order to train the regression model, we simply set the traffic density ratio as follows : *Empty*=0, *Fluid*=0.33, *Heavy*=0.67, and *Jam*=1. We trained a HIK kernel SVM regression model with the BOVW descriptor with 256 words as well as the linear SVM regression model with the CNN feature from the Caffe's model.

Figure 3 shows the density plots obtained on 1231 annotated frames from the "highway" video. As can be seen, the output of our regression models (the red and green curves) fit surprisingly well on the real traffic density (the blue curve). We evaluated the correlation between these curves with the Pearson correlation coefficient: $P_{X,Y} = \frac{Cov(X,Y)}{\sigma_X \cdot \sigma_Y}$. The resulting correlation coefficient for the BOVW descriptor is 0.96, and the CNN feature is 0.88. We also got correlation results for the "tunnelExit" and the "turnpike" videos. For TunnelExit, BOVW is 0.82 and CNN is 0.76. Due to the perspective issues on the Turnpike video (this video shows traffic on the side which is not the case for most images on the Motorway dataset, c.f.Fig. 1), we got a slighly lower correlation of 0.61 for BOVW and 0.62 for CNN.

5. CONCLUSION

In this paper, we show that highway traffic can be assessed by accounting solely on 2D features. Accurate results have been obtained both on a single-camera dataset (UCSD) and on a multi-camera dataset (Motorway). We also showed that traffic density can also be well estimated, even when the system is trained on one dataset and tested on another one. These results strongly suggest that traffic activities are highly correlated to their 2D texture and that motion features such as tracking and optical flow are not essential to solve these kinds of problems.

6. REFERENCES

[1] Y-K Jung, K-W Lee, and Y-S Ho, "Content-based event retrieval using semantic scene interpretation for automated traffic surveillance," *IEEE Trans on Int. Trans. Sys*, vol. 2, no. 3, pp. 151–163, 2001.

[2] Z. Zhigang, X. Guangyou, Y. Bo, S. Dingji, and L. Xueyin, "Visatram: A real-time vision system for automatic traffic monitoring," *Image and Vision Computing*, vol. 18, no. 10, pp. 781–794, 2000.

[3] K. Shunsuke, M. Yasuyuki, I. Katsushi, and S. Masao, "Traffic monitoring and accident detection at intersections," *IEEE Trans. Intelligent Transportation Systems,*, vol. 1, no. 2, pp. 108–118, 2000.

[4] J. Lee and A. Bovik, "Estimation and analysis of urban traffic flow," in *IEEE ICIP*, 2009, pp. 1157–1160.

[5] A. Chan and N. Vasconcelos, "Classification and retrieval of traffic video using auto-regressive stochastic processes," in *IEEE Intelligent Vehicles Symposium*, 2005, pp. 771–776.

[6] K.Derpanis and R. Wildes, "Classification of traffic video based on a spatiotemporal orientation analysis," in *Workshop on App. of Comp. Vis.*, 2011, pp. 606–613.

[7] Shan Hua, Jiansheng Wua, and Ling Xub, "Real-time traffic congestion detection based on video analysis," *Journal of Information and Computational Science*, vol. 9, no. 10, pp. 2907–2914, 2012.

[8] C. Hua-Tsung, T. Li-Wu, G.Hui-Zhen, L. Suh-Yin, and L. B-SP, "Traffic congestion classification for nighttime surveillance videos," in *Proc. on IEEE ICMEW*, 2012, pp. 169–174.

[9] R. Amina and K. Shoab, "Traffic congestion classification using motion vector statistical features," in *Proc. of ICMV*. ISOP, 2013.

[10] V Kastrinaki, M Zervakis, and K. Kostas, "A survey of video processing techniques for traffic applications," *Image and vision computing*, vol. 21, no. 4, pp. 359–381, 2003.

[11] Andrews Sobral, Luciano Oliveira, Leizer Schnitman, and Felippe Souza, "Highway traffic congestion classification using holistic properties," in *10th IASTED International Conference on Signal Processing, Pattern Recognition and Applications*, 2013.

[12] Ouessai Asmaa, Keche Mokhtar, and Ouamri Abdelaziz, "Road traffic density estimation using microscopic and macroscopic parameters," *Image and Vision Computing*, vol. 31, no. 11, pp. 887–894, 2013.

[13] D. Beymer, P. McLauchlan, B. Coifman, and J. Malik, "A real-time computer vision system for measuring traffic parameters," in *Proc. on IEEE CVPR*, Jun 1997, pp. 495–501.

[14] F. Porikli and Xiaokun Li, "Traffic congestion estimation using hmm models without vehicle tracking," in *IEEE Intelligent Vehicles Symposium*, June 2004, pp. 188–193.

[15] K. Derpanis and R. Wildes, "Classification of traffic video based on a spatiotemporal orientation analysis," in *IEEE Workshop on Applications of Computer Vision (WACV)*, 2011, pp. 606–613.

[16] A. Ess, T. Mueller, H. Grabner, , and L. van Gool, "Segmentation-based urban traffic scene understanding," in *BMVC*, September 2009.

[17] I. Sikiric, K. Brkic, J. Krapac, and S. Segvic, "Image representations on a budget: Traffic scene classification in a restricted bandwidth scenario," in *IEEE Intelligent Vehicles Symposium*, 2014, pp. 845–852.

[18] K. Chatfield, V. Lempitsky, A. Vedaldi, and A. Zisserman, "The devil is in the details: an evaluation of recent feature encoding methods," in *BMVC*, 2011.

[19] R. Arandjelović and A. Zisserman, "All about VLAD," in *in proc of CVPR*, 2013.

[20] F. Perronnin, J. Snchez, and T. Mensink, "Improving the fisher kernel for large-scale image classification," in *in proc. of ECCV*, 2010.

[21] J. Wang, J. Yang, K. Yu, F. Lv, T. Huang, and Y. Gong, "Locality-constrained linear coding for image classification," in *in proc of CVPR*, 2010, pp. 3360–3367.

[22] Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell, "Caffe: Convolutional architecture for fast feature embedding," *arXiv preprint arXiv:1408.5093*, 2014.

[23] K. Chatfield, K. Simonyan, A. Vedaldi, and A. Zisserman, "Return of the devil in the details: Delving deep into convolutional nets," in *British Machine Vision Conference*, 2014.

[24] N. Goyette, P. Jodoin, F. Porikli, J. Konrad, and P. Ishwar, "Changedetection.net: A new change detection benchmark dataset," in *IEEE CVPR Change Detection Workshop*, June 2012, pp. 1–8.

[25] A. Vedaldi and B. Fulkerson, "VLFeat: An open and portable library of computer vision algorithms," 2008.