CONSTRUCTING BINARY DESCRIPTORS WITH A STOCHASTIC HILL CLIMBING SEARCH

Nenad Markuš*, Igor S. Pandžić

University of Zagreb
Faculty of Electrical Engineering and Computing
Unska 3, 10000 Zagreb, Croatia

Jörgen Ahlberg

Linköping University
Department of Electrical Engineering
SE-581 83 Linköping, Sweden

ABSTRACT

Binary descriptors of image patches provide processing speed advantages and require less storage than methods that encode the patch appearance with a vector of real numbers. We provide evidence that, despite its simplicity, a stochastic hill climbing descriptor construction process defeats recently proposed alternatives on a standard discriminative power benchmark. The method is easy to implement and understand, has no free parameters that need fine tuning, and runs fast. We use our findings to construct a new keypoint descriptor which provides certain advantages over competing approaches.

1. INTRODUCTION

Local image patch descriptors proved to be a useful tool in computer vision. Some examples include object/scene recognition [1], image retrieval [2], face verification [3], face alignment [4] and image stitching [5]. The importance of these applications lead to a large number of publications that introduced different descriptors. Out of various approaches, binary keypoint descriptors [6, 7, 8, 9, 10, 11, 12, 13] recently provoked considerable interest as they require far less storage and provide much faster matching times compared to alternative methods that encode the patch appearance as a vector of real numbers [14, 15, 16]. Fan et al. [13] showed that their performance is competitive to that of SIFT [14] in a number of interesting scenarios, such as object retrieval.

In this paper, we experimentally investigate certain binary descriptor construction procedures. Our contributions are:

- We show the effectiveness of the stochastic hill climbing search in binary descriptor construction.
- We propose a novel procedure for constructing decision ferns (simplified decision trees, see [17]) from a set of labeled image patch pairs.
- We introduce a new binary keypoint descriptor which achieves similar accuracy to SIFT [14] and SURF [15] while being significantly faster.

2. SELECTING DISCRIMINATIVE BITS

A binary descriptor consists of $b$ classifiers. Each classifier, denoted as a bit in further text, outputs a 0 or a 1 for a given image patch. Thus, a descriptor maps image patches into binary vectors which are used as signatures for search engines. The idea is to select individual bits in such a way that matching patches are "close" in the Hamming space and non-matching patches are "far". Let $\text{AUC}(d)$ denote the area under the receiver operating characteristics (ROC) curve for a descriptor $d$, measured on a set of matching and non-matching image patch pairs. The true positive rates (TPRs) and false positive rates (FPRs) are computed by thresholding the Hamming distance between signatures. Brown et al. [18] proposed to select the parameters of real-valued descriptors (such as pooling region locations) by optimizing the AUC criterion. We apply this reasoning to select individual bits (i.e., dimensions) of a binary descriptor. Since the AUC criterion is not continuous, we apply stochastic hill climbing to achieve our goal. This also relates our paper to a large body of research in feature selection (for example, see [19, 20]). The whole procedure can be summarized by the following steps:

1. Generate a pool $P$ of $B$ bits.
2. Select $b$ bits from $P$ to obtain a descriptor $d^*$.
3. Iterate $N$ times:
   
   (a) Swap a random bit from $d^*$ with a random bit from $P$ to obtain $d$.
   
   (b) If $\text{AUC}(d^*) < \text{AUC}(d)$, set $d^* = d$.

The number of iterations, $N$, is set to $4 \cdot B$ as this led to good results in all our experiments. Note that the described procedure does not specify the exact method how to generate individual bits to obtain a pool $P$. The method could be taken, for example, from a state-of-the-art paper, such as [13]. We perform extensive experimental validation of the proposed approach in the next subsection.
2.1. Experimental validation

We compare the proposed procedure to the recent boosting- and correlation-based bit selection methods. The boosting-based methods [11, 12] are based on the principle of reweighting the training data during learning, inspired by AdaBoost [21]. The correlation-based method [13] sequentially selects accurate bits that have low correlation with other already selected bits. Each method has a continuous parameter that significantly influences performance: boosting shrinkage coefficient and the correlation threshold. In our experiments, we select these parameters to maximize the accuracy on the test set. Note that the proposed descriptor construction procedure does not have free parameters that need to be fine-tuned, which is an obvious advantage.

We use the dataset introduced by Brown et al. [18] to provide experimental evidence that the proposed bit selection procedure has practical value. We report the results in terms of ROC curves and 95% error rates. The dataset consists of three subsets: Notre Dame, Liberty and Yosemite. Each contains a large number of $64 \times 64$ rotation- and scale-normalized patches extracted around DoG keypoints [14]. The ground truth for each subset consists of 100k, 200k and 500k pairs of patches, 50% correspond to matching pairs and 50% to non-matching pairs. We use a simplified notation for the training and testing subsets in our experiments. For example, L/ND will denote the scenario in which the Liberty subset patch pairs were used for descriptor learning and the Notre Dame subset patch pairs for testing.

We compare the mentioned bit selection procedures on the task of improving the BRIEF descriptor [6]. The basic idea of BRIEF is to construct a 256 bit descriptor by performing 256 pixel intensity comparison binary tests (“$I_{x_1, y_1} < I_{x_2, y_2}$?”) on an incoming image patch. The pixel sampling locations are fixed in runtime. It is intuitive that the distribution of these locations matters for accuracy. The authors of the original paper [6] propose to pick them at random. Here we show that the accuracy of the descriptor improves if we carefully select 256 tests out of 1024 random ones. This is performed by running a boosting-, correlation- or stochastic hill climbing-based bit selection procedure on a large training set from [18]. The accuracy is measured on a test set. Figure 1 shows the ROC curves on the 100k subsets for experiments from [18]. Table 1 shows the 95% error rates. We can see that the proposed method leads to lower error rates in the majority of scenarios. Also, its behavior is consistent, i.e., it obtains good results for all train/test pairs, which is not the case for the competing approaches. Bit selection processing times can be seen in Table 2. The parameters of all methods were fixed. Thus, this is not an entirely fair comparison since both boosting- and correlation-based approaches typically require several reruns for parameter fine-tuning.

In the next section we apply the proposed framework to construct a novel keypoint descriptor from a large pool of randomized bits based on decision fern ensembles.

3. LEARNING AN ENSEMBLE OF FERNS FROM LABELED IMAGE PAIRS

We use decision fern ensembles [17, 22] as data structures for predicting the label of an incoming image patch. Each fern consists of a sequence of $n$ binary tests and a lookup table of $2^n$ scalar values. In runtime, the tests are applied to an image patch and the resulting sequence of zeroes and ones is interpreted as an index into a lookup table. The read value is defined as the output of the fern. The binary label of an image patch is obtained by thresholding the summed outputs of all ferns: if the sum is greater than zero, the label is set to “1”, and “0” otherwise. Thus, ferns followed by thresholding form a classifier which can be used in descriptors. The proposed framework is similar to the tree-based hashing procedure by Lin et al. [23]. They decompose the tree learning into two steps: hash code inference based on GraphCut [24] and ensemble learning with AdaBoost [21]. Our procedure constructs an ensemble by directly optimizing an appropriate loss function. The details are given further in the text.

We use ferns with pixel intensity comparison binary tests (same ones from BRIEF; also see [25, 17]). Other options are, for example, tests based on gradient orientations [11, 13] or some other more complex features [16]. However, these approaches are computationally expensive and, thus, we avoid them. The pixel sampling locations and the values in the lookup table, i.e., the parameters which specify an ensemble, are learned from the set of labeled image patch pairs: \{$(x_t, y_t, l_t) : t = 1, 2, \ldots, T$\} (the label $l_t = +1$ indicates that patches $x_t$ and $y_t$ are a matching pair, and $l_t = -1$ indicates a non-matching pair). To do this, we attempt to maximize the correlation between the ground truth labels and the ones generated with the ensemble:

$$Q = \sum_{t=1}^{T} l_t \cdot l(x_t) \cdot l(y_t) \quad (1)$$

For $K$ ferns in the ensemble, the labels are computed as

$$l(x) = \begin{cases} +1, & \sum_{k=1}^{K} t_k^T i_k(x) > 0 \\ 0, & \sum_{k=1}^{K} t_k^T i_k(x) = 0 \\ -1, & \text{otherwise} \end{cases} \quad (2)$$

where the inner product $t_k^T i_k(x)$ represents the output of the $k$th fern: $t_k$ is the “vectorized” lookup table and $i_k(x)$ consists of $2^n - 1$ zeroes and a single one. The vector $i_k(x)$

| Selection type | Correlation [13] | Boosting [11, 12] | Proposed |
|---------------|-----------------|-----------------|----------|
| Time [s]      | 3281            | 927             | 175      |

Table 2: Processing times for selecting 256 out of 1024 bits for a dataset of 100k patch pairs.
Fig. 1: The ROC curves of the improved BRIEF descriptors on the testing subsets from [18].
is constructed by applying \( n \) binary tests to the patch \( x \) and indicates the position at which to sample the lookup table. This representation was inspired by the work done by Ren et al. [26]. Optimizing \( Q \) (equations 1 and 2) is difficult and we resort to a heuristic method. Each fern is added sequentially to the ensemble. A given tuple of \( n \) binary tests specifies the vectors \( i_k(x_i) \) and \( i_k(y_i) \) for the \( k \)th fern. The lookup table is created with a simple greedy coordinate ascent algorithm: for each coordinate of \( t_k \) pick an integer from \( \{0, \pm 1, \pm 2, \ldots, \pm 8\} \) that maximizes \( Q \) (if there are multiple candidates, keep the integer with the smallest absolute value). We also tried alternatives based on smooth approximations of \( Q \), local discriminative embedding [27] and metric learning [28]. However, the obtained results were not as good. In our experiments, the \( n \) binary tests associated with each fern are generated at random (this approach leads to good results).

We learn 8192 ensembles consisting of 4 ferns with \( n = 4 \) binary tests. Each ensemble was learned on randomly picked 10% samples from the training set. Next, we use the stochastic hill climbing search to obtain a 256 bit descriptor. Figure 2 shows how the proposed descriptor compares to some other approaches [14, 15, 11, 6] on the L/ND experiment (the results for other experiments from [18] are similar). Table 3 compares the required computational resources. These results show a typical accuracy/processing speed trade-off. The BinBoost and SIFT descriptors are based on gradient orientation histograms which are known for their discriminative power and high computational cost (for example, in the field of object detection [29]). Thus, these two descriptors are not suitable for real-time applications, especially on mobile devices. On the other hand, the BRIEF descriptor is extremely fast but its accuracy might not be sufficient in some cases. Our descriptor offers accuracy similar to SIFT and SURF at significantly better runtime processing speed. Another advantage is its small footprint (256 bits), which leads to high matching speed and low memory requirements. The results can be reproduced with 

http://hotlab.fer.hr/_download/repository/next.zip.

4. CONCLUSION

We have shown that a simple stochastic hill climbing bit selection procedure outperforms recent alternatives [11, 12, 13] on a large dataset [18]. Also, we introduced a novel binary keypoint descriptor which achieves similar accuracy to SIFT [14] and SURF [15], BRIEF [6] and our method on the L/ND experiment.

5. ACKNOWLEDGEMENTS

This research is partially supported by Visage Technologies AB, Linköping, Sweden, and by the Ministry of Science, Education and Sports of the Republic of Croatia.
6. REFERENCES

[1] G. Csurka, C. Dance, L. Fan, J. Willamowski, and C. Bray, “Visual categorization with bags of keypoints,” in *International Workshop on Statistical Learning in Computer Vision, ECCV*, 2004.

[2] F. Perronnin, J. Sanchez, and T. Mensink, “Improving the Fisher Kernel for Large-Scale Image Classification,” in *ECCV*, 2010.

[3] K. Simonyan, O. M. Parkhi, A. Vedaldi, and A. Zisserman, “Fisher Vector Faces in the Wild,” in *BMVC*, 2013.

[4] X. Xiong and F. De la Torre, “Supervised Descent Method and its Applications to Face Alignment,” in *CVPR*, 2013.

[5] M. Brown and D. G. Lowe, “Recognising panoramas,” in *ICCV*, 2003.

[6] M. Calonder, V. Lepetit, C. Strecha, and P. Fua, “BRIEF: Binary Robust Independent Elementary Features,” in *ECCV*, 2010.

[7] E. Rublee, V. Rabaud, K. Konolige, and G. R. Bradski, “ORB: An efficient alternative to SIFT or SURF,” in *ICCV*, 2011.

[8] S. Leutenegger, M. Chli, and R. Y. Siegwart, “BRISK: Binary Robust Invariant Scalable Keypoints,” in *ICCV*, 2011.

[9] T. Trzcinski and V. Lepetit, “Efficient Discriminative Projections for Compact Binary Descriptors,” in *ECCV*, 2012.

[10] A. Alahi, R. Ortiz, and P. Vandergeyts, “FREAK: Fast Retina Keypoint,” in *CVPR*, 2012.

[11] T. Trzcinski, M. Christoudias, V. Lepetit, and P. Fua, “Boosting Binary Keypoint Descriptors,” in *CVPR*, 2013.

[12] X. Yang and K.-T. Cheng, “Local Difference Binary for Ultra-fast and Distinctive Feature Description,” *PAMI*, 2014.

[13] B. Fan, Q. Kong, T. Trzcinski, Z. Wang, C. Pan, and P. Fua, “Receptive Fields Selection for Binary Feature Description,” *IEEE Transaction on Image Processing*, 2014.

[14] D. G. Lowe, “Object recognition from local scale-invariant features,” in *ICCV*, 1999.

[15] H. Bay, T. Tuytelaars, and L. Van Gool, “SURF: Speeded Up Robust Features,” in *ECCV*, 2006.

[16] K. Simonyan, A. Vedaldi, and A. Zisserman, “Learning Local Feature Descriptors Using Convex Optimisation,” in *ECCV*, 2012.

[17] M. Ozuysal, M. Calonder, V. Lepetit, and P. Fua, “Fast Keypoint Recognition using Random Ferns,” *PAMI*, 2010.

[18] M. Brown, G. Hua, and S. Winder, “Discriminative Learning of Local Image Descriptors,” *PAMI*, 2011.

[19] H. Peng, F. Long, and C. Ding, “Feature Selection Based on Mutual Information: Criteria of Max-Dependency, Max-Relevance, and Min-Redundancy,” *PAMI*, 2005.

[20] I. Guyon and A. Elisseeff, “An Introduction to Variable and Feature Selection,” *JMLR*, 2003.

[21] J. Friedman, T. Hastie, and R. Tibshirani, “Additive Logistic Regression: a Statistical View of Boosting,” *Annals of Statistics*, 1998.

[22] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, Springer, 2009.

[23] G. Lin, C. Shen, Q. Shi, A. van den Hengel, and D. Suter, “Fast Supervised Hashing with Decision Trees for High-Dimensional Dat,” in *CVPR*, 2014.

[24] Y. Boykov, O. Veksler, and R. Zabih, “Fast Approximate Energy Minimization via Graph Cuts,” *PAMI*, 2001.

[25] Y. Amit and D. Geman, “Shape quantization and recognition with randomized trees,” *Neural Computation*, 1997.

[26] S. Ren, X. Cao, Y. Wei, and J. Sun, “Face Alignment at 3000 FPS via Regressing Local Binary Features,” in *CVPR*, 2014.

[27] H.-T. Chen, H.-W. Chang, and T.-L. Liu, “Local Discriminant Embedding and Its Variants,” in *CVPR*, 2005.

[28] A. Bellet, A. Habrard, and M. Sebban, “A Survey on Metric Learning for Feature Vectors and Structured Data,” http://arxiv.org/abs/1306.6709, 2013.

[29] N. Dalal and B. Triggs, “Histograms of oriented gradients for human detection,” in *CVPR*, 2005.

[30] H. Jegou, M. Douze, and C. Schmid, “Product Quantization for Nearest Neighbor Search,” *PAMI*, 2011.

[31] M. Norouzi and D. Fleet, “Minimal Loss Hashing for Compact Binary Codes,” in *ICML*, 2011.