Learning Local Descriptors by Optimizing the Keypoint-Correspondence Criterion

Nenad Markuš[†], Igor S. Pandžić[†], and Jörgen Ahlberg[‡]

[†] University of Zagreb, Faculty of Electrical Engineering and Computing, Unska 3, 10000 Zagreb, Croatia
[‡] Computer Vision Laboratory, Dept. of Electrical Engineering, Linköping University, SE-581 83 Linköping, Sweden

***Abstract*—Current best local descriptors are learned on a large dataset of matching and non-matching keypoint pairs. However, data of this kind is not always available since detailed keypoint correspondences can be hard to establish. On the other hand, we can often obtain labels for pairs of keypoint bags. For example, keypoint bags extracted from two images of the same object under different views form a matching pair, and keypoint bags extracted from images of different objects form a non-matching pair. On average, matching pairs should contain more corresponding keypoints than non-matching pairs. We describe an end-to-end differentiable architecture that enables the learning of local keypoint descriptors from such weakly-labeled data.**

I. Introduction

Local descriptors are a widely used tool in computer vision and pattern recognition. Some example applications include object/scene recognition and retrieval [1], [2], [3], face verification [4], [5], face alignment [6], image stitching [7], 3D shape estimation [8] and 3D model retrieval/matching [9], [10]. However, despite years of research, there is still room for improvement, as confirmed by recent results based on convolutional neural networks [11], [12], [13], [14]. Also, we view the research in local descriptors complementary to keypoint detection research, which is still an active area of computer vision (see, for example [15]).

A promising way of obtaining discriminative local descriptors is to learn them from annotated keypoint correspondences. This can be used to form a set of matching and non-matching keypoint pairs:

$$\mathcal{D}_{KP} = \{(k_{i1}, k_{i2}, l_i)\}_{i=1}^{N}. \quad (1)$$

The label $l_i \in \{+1, -1\}$ indicates whether keypoints k_{i1} and k_{i2} form a matching or a non-matching pair. See [16], [17], [18], [11], [12], [13] for some recent examples of descriptor learning methods that use data in this form. Another possibility is to form a set of keypoint triplets:

$$\mathcal{D}_{KT} = \{(k_i, k_i^+, k_i^-)\}_{i=1}^{N}, \quad (2)$$

where k_i and k_i^+ match and k_i and k_i^- do not. Balntas et al. [14] use data in this form in their method. The standard dataset for learning and benchmarking various image keypoint descriptors was introduced by Brown et al. [16]. It contains around 1.5M patches cropped around difference of Gaussians keypoints [19] obtained from multiple views of three different scenes: the Notre Dame Cathedral, the Statue of Liberty and the Yosemite Half Dome. High-quality keypoint labels were obtained with a multi-view stereo algorithm [20]. This makes the dataset reliable both for learning local image-patch descriptors from "handcrafted" features [21], [17], [18] and large models based on convolutional neural networks [22], [11], [12], [23], [13], [14] (the siamese-network framework of Hadsell et al. [24]). However, in the general case, this kind of data is relatively hard to obtain, even more so for non-image data (e.g., 3D models, depth maps, voxel data, video signals, etc.).

Instead of having a dataset with individual keypoint correspondences (which lead to dataset types (1) and (2)) for learning local descriptors as in most prior work, we assume a set of labeled *bags of keypoints* (here we intentionally use the terminology from multiple instance learning [25] as our ideas are closely related with the field). We denote this weakly-labeled dataset as

$$\mathcal{D}_{BT} = \{(K_i, K_i^+, K_i^-)\}_{i=1}^{N}, \quad (3)$$

where bags K_i and K_i^+ form a matching pair, bags K_i and K_i^- form a non-matching pair, and each bag is a set of n keypoints, $K = \{k_1, k_2, \ldots, k_n\}$. Data of this kind is relatively easy to generate. For example, keypoint bags extracted from two images of the same object under different views form a matching pair. These bags can be used together with a keypoint bag extracted from an image of some unrelated object to form a triplet from Equation (3). See Figure 1 for an illustration. This paper describes a method for learning local descriptors from such weakly-labeled data.

II. Related work

We already mentioned a large body of work in local image descriptors and we will not repeat these standard approaches in this section.

We would like to explicitly mention the work of Paulin et al. [27] since they are also motivated to obtain discriminative local descriptors by means that do not require strongly-labeled data (equations (1) and (2)). To achieve their goal, they adapt the convolutional kernel network approach, which is an unsupervised framework for learning convolutional architectures [28].

The learning procedure we propose in the next section is related to the one by Arandjelović et al. [29], as they also propose to learn descriptors from weakly-labeled data. Unlike us, they do not focus on local descriptors and learn whole image representations instead. Also, they derive their learning procedure from a different perspective: we are concerned with local image correspondences and how to find them, and they focus on learning a global descriptor for image retrieval. It

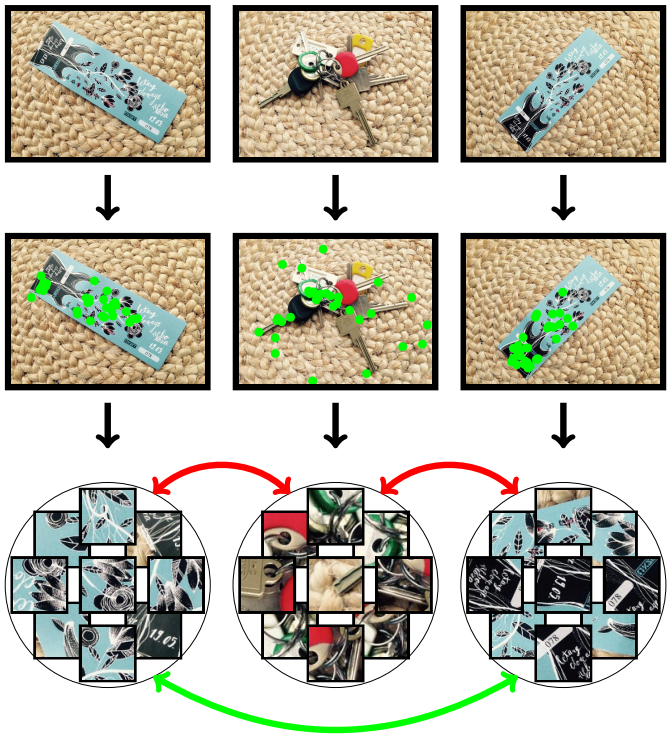

Fig. 1: Each image in the dataset (first row) is processed with a keypoint detector (second row) and transformed into a bag of visual words (third row). Some bags form matching pairs (green arrow, $\longleftrightarrow$) and some form non-matching pairs (red arrows, $\longleftrightarrow$). On average, matching pairs should contain more corresponding local visual words than non-matching pairs. We propose to *learn local descriptors* by optimizing the mentioned local correspondence criterion on a given dataset. Note that most prior work assumes local correspondences are known in advance (e.g., [26], [11], [12], [13], [14]).

is not clear how well would their system work in finding local correspondences between two images. Also, we learn our descriptors *directly* for comparisons with L_2 distance.

III. Method

We study how to the learn parameters of a descriptor extraction process that transforms a local neighborhood of a keypoint (e.g, a patch extracted around a distinctive corner within an image) into a short vector in such a way that similar keypoints are "close" and dissimilar keypoints are "far". Two attractive properties of such representations are low memory requirements and fast matching times. Unlike most prior work, our learning method exploits the information in weakly-labeled data to achieve mentioned goals.

In this paper, we denote the descriptor extraction process as e (this is basically a number of predefined computational steps). For example, in our experiments, e is a convolutional neural network (see Table I for its architecture) that maps a 32×32 local image patch into a 64-dimensional vector. We denote the parameters of e as θ_e. Here we describe an effective procedure for learning θ_e from the training data given by Equation (3). First, we define that two keypoints match if the L_2 distance between their signatures (extracted by e) is less than or equal to some threshold $\tau \in \mathbb{R}$. This threshold is a parameter of the learning process and we specify some recommended values later in the text. Next, we define a *matching score* between two bags of keypoints (both of size n), K_1 and K_2, as

$$S_{e,\tau}(K_1, K_2) = \frac{m_{e,\tau}(K_1, K_2)}{n}, \tag{4}$$

where $m_{e,\tau}(K_1, K_2)$ is the number of keypoints from K_1 that have a matching keypoint in K_2 for the descriptor extractor e and threshold τ. Optimal matching could be computed with the Hungarian algorithm in $O(n^3)$ time. However, this is too slow in our case and we use the following $O(n^2)$ approximation (inspired by the "sum-max" match kernel from [30]):

$$m_{e,\tau}(K_1, K_2) = \sum_{i=1}^{n} \left[\min_{j=1}^{n} d_{ij}^2 \leq \tau \right],$$

where $[\,\cdot\,]$ represents the indicator function[1] and d_{ij} is the Euclidean distance between descriptors of $k_i \in K_1$ and $k_j \in K_2$, i.e.,

$$d_{ij} = ||e(k_i) - e(k_j)||_2.$$

We want high $S_{e,\tau}$ for matching bags and low $S_{e,\tau}$ for non-matching bags. Thus, a suitable loss for parameter learning is

$$L = \sum \frac{S_{e,\tau}(K, K^-)}{S_{e,\tau}(K, K^+) + \epsilon}, \tag{5}$$

where the summation goes over $(K, K^+, K^-) \in \mathcal{D}_{BT}$ (Equation (3)) and ϵ is a small constant included for numerical stability. However, since $S_{e,\tau}$ is not continuous, we cannot apply the standard gradient-based learning techniques. Thus, we resort to the following approximation of the function $[x \leq \tau]$ for $x \in \mathbb{R}$:

$$[x \leq \tau] \approx \frac{1}{1 + \exp(\beta(x - \tau))},$$

where the parameter β regulates the "strength" of the approximation. Since the loss function L is now differentiable, the parameters θ_e can be tuned with standard backpropagation-based methods. However, as we cannot optimize the proposed criterion (i.e., find its global minimum), we approximate the solution with a local minimum to which the learning converges and experimentally show that this leads to good results.

To simplify the implementation, we require that the extractor outputs descriptors of unit length: $||e(k_i)||_2 = ||e(k_j)||_2 = 1$. Notice that in this scenario

$$d_{ij}^2 = ||e(k_i) - e(k_j)||_2^2 = 2 - 2e(k_i)^T e(k_j)$$

and the matching score function $S_{e,\tau}$ (Equation (4)) depends only on the matrix $\mathbf{S} \in \mathbb{R}^{n \times n}$ computed as

$$\mathbf{S} = \mathbf{E}_1 \mathbf{E}_2^T,$$

where the rows of matrices $\mathbf{E}_1$ and $\mathbf{E}_2$ contain descriptors extracted with the extractor e from keypoints in K_1 and

[1] $[p] = 1$ if the proposition p is true and $[p] = 0$ otherwise.

Conv. layer	1	2	3	4
Filter size	3×3	4×4	3×3	1×1
Stride	1	2	1	1
Output channels	32	64	128	32
Activation function	ReLU	ReLU	None	None
Max pooling?	No	No	Yes, 2×2	No

TABLE I: Our descriptor extractor is a fairly simple convolutional neural network that maps a 32×32 RGB patch into a 64-dimensional vector. It consists of four convolutional layers (given in table above), a fully connected layer that maps the output of the last convolutional layer to 64 neurons and a final L_2 normalization module (i.e., the output vector has unit length). The network has around 180k parameters.

K_2. The backpropagation expressions are quite elegant in this setting:

$$\frac{\partial S_{e,\tau}(K_1,K_2)}{\partial \mathbf{E}_1} = \frac{\partial S_{e,\tau}(K_1,K_2)}{\partial \mathbf{S}} \cdot \mathbf{E}_2$$
$$\frac{\partial S_{e,\tau}(K_1,K_2)}{\partial \mathbf{E}_2} = \left(\frac{\partial S_{e,\tau}(K_1,K_2)}{\partial \mathbf{S}}\right)^T \cdot \mathbf{E}_1$$

where $\partial S_{e,\tau}(K_1,K_2)/\partial \mathbf{S}$ is straightforward to compute because $S_{e,\tau}$ contains only the standard components usually used in neural networks (see the definition, Equation (4)). The proposed computational steps can be implemented very efficiently in just a few hundred lines of Torch7 code. Another advantage of unit-length descriptors is that this greatly simplifies the selection of the threshold τ because the Euclidean distance between two descriptors falls in the $[0, 2]$ interval.

The next section provides experiments which show that the proposed learning procedure leads to good results.

IV. Experiments

We perform experiments with image patch descriptors extracted around local detected keypoints. We use the following datasets to provide numerical evidence for our observations:

- UKB [31] (2500 objects, 4 views each);
- ZuBuD [32] (200 buildings, 5 images each);
- INRIA Holidays [33] (approximately 1500 images of 500 different scenes).

See Figure 2 for some examples.

In our experiments, each image is downsampled by a factor of four to reduce noise, filter out irrelevant content and increase keypoint stability. Next, image patches are extracted around detected keypoints. These steps set up a basis for a fair comparison between different descriptors since we always use the same keypoints (location, size[2] and orientation).

A. Learning convolutional features with our method

The architecture of our neural network-based descriptor extractor e can be seen in Table I. To generate the training data for our method, we partition the keypoint bags extracted from the UKB dataset into two subsets. The larger subset contains 2200 objects and is used to sample keypoint bag triplets. This subset is used for learning and the rest of the UKB dataset (300 objects) is used for validation and testing. We train each of our networks for 128 rounds. Each round starts by sampling approximately 5000 random (K, K^+, K^-) triplets from the training dataset and loading them onto the GPU. These triplets are then used in a standard mini-batch learning for 512 iterations of `rmsprop`: we approximate the loss function L (Equation (5)) with a subset of 32 randomly selected triplets. After each round has finished, we test the performance of the network on the validation dataset. If the validation loss does not decrease for several rounds, the learning rate is halved. The initial learning rate of 0.001 works good for mini-batches of size 32. Also, we set $\epsilon = 10^{-6}$, $\beta = 20$ and $\tau = 0.8$ (however, it seems that the method is not particularly sensitive to any of these values). The total learning time in this setting is approximately one day on a GeForce GTX 970 GPU with cuDNN. See Figure 3 for a typical behavior of average loss (L from Equation (5)) divided by the number of triplets) throughout the learning process.

We learn two models with the same architectures and learning parameters, as described in the previous paragraph. The difference between these models were the keypoints. The first model is learned on patches extracted with the FAST keypoint detector (75 per image). The second model combines both FAST and DoG keypoints ($75 + 75$ per image).

In the next section we present a comparison of some recently introduced local descriptors.

B. Matching-based retrieval

Following [18], we implement a simple visual search engine to compare the discriminative power of different descriptors. The retrieval is based on the number of matching keypoints between the query and database images ($S_{d,\tau}(K_1,K_2)$, see Equation (4) for the definition). The threshold τ is tuned separately for each descriptor to produce the best possible results (we simply try all reasonable values; note that this is fair since all descriptors get the same treatment [18]). We assume that improvements in feature detection and aggregation would benefit all descriptors equally, without changing the relative performance differences. This is a common assumption needed to keep the experimentation time reasonable; see, for example, [27]. We benchmark the retrieval performance with the nearest neighbor (NN), first tier (FT) and second tier (ST) scores. The idea is to check the ratio of retrieved objects in the query's class that also appear within the top k matches. Specifically, for a class with C members, $k = 1$ for NN, $k = C - 1$ for FT and $k = 2(C - 1)$ for ST. The final score is an average over all the objects in the database.

For competing neural network-based descriptors, we download the models provided by the authors and apply PCA when necessary (e.g., to reduce decriptor size to a more manageable dimensionality). Also, note that the binary descriptors are compared with the Hamming distance.

Tables IIa and IIb contain the retrieval results for various competing approaches. The descriptor learned with our method (first two rows of both tables) obtain best results on average.

[2] A patch of a fixed size around the keypoint is resampled to 32×32 or 64×64 pixels, depending on the requirements of the descriptor-extraction process.

Fig. 2: Samples from databases used in our experiments: UKB [31] (first row), ZuBuD [32] (second row), INRIA Holidays [33] (third row).

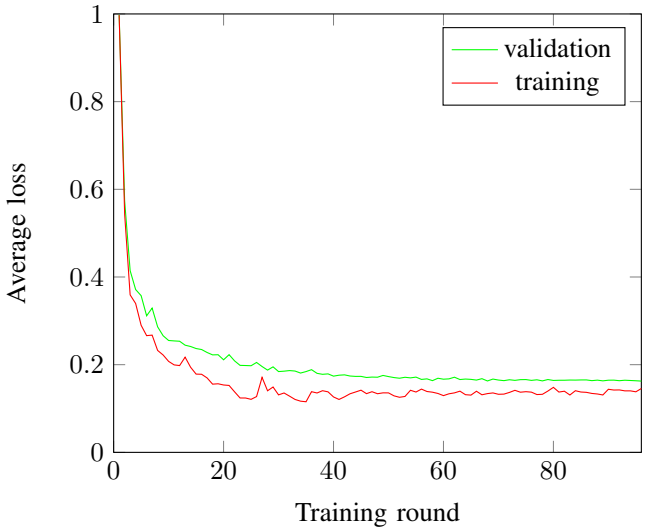

Fig. 3: Typical behavior of the learning process on the UKB training and validation subsets.

The difference is particularly significant when compared to "handcrafted" features. When comparing the retrieval scores of our two models, it seems that tuning the performance for a particular patch-appearance distribution (i.e., the keypoint detector) leads to a non-negligible difference. The model learned on FAST keypoints performs slightly better on the FAST benchmark (Table IIa) than the model learned on both DoG and FAST keypoints. A possible explanation is that a part of the representational power of the network is expended for representing blob-like DoG features. It is the other way arround on the DoG benchmark (Table IIb). Note that the experiments presented so far measure how well do the descriptors recognize individual keypoints (a useful property for, e.g., object pose recognition). Our descriptors are *tuned* for this kind of matching-based recognition task and it would be interesting to see how they perform on some other task.

C. VLAD-based retrieval

In this subsection we experiment with image retrieval based on local feature aggregation. Note that none of the descriptors were tuned specifically for this task. For each image, we transform approximately 1000 SURF [35] keypoints into descriptors and encode them with VLAD [3] (a simplified Fisher kernel representation [2]). The centroids were generated with k-means on a subset of images. The similarity between two images is measured by an inner product between their VLADs. Figure 4 shows the NN, FT and ST retrieval scores on three datasets for different local descriptors (from our models, we show only the one trained on FAST keypoints because both achieve approximately the same scores in these experiments). We see that our approach leads to results as good as or superior to the so far best published methods and significantly outperforms SIFT (which serves as a baseline).

V. Conclusion

We point out that the current best methods for learning local descriptors require a large number of matching and non-matching keypoint pairs. Data of this kind is not always available and, thus, these methods are not always applicable. To address this issue, we introduce a novel algorithm for learning local descriptors from weakly-labeled datasets. Experimental results show that our descriptors compare well to the best available ones. The code and the learned models can be obtained from https://github.com/nenadmarkus/wlrn.

Acknowledgements

This research was partially supported by Visage Technologies AB (Linköping, Sweden), and by the Croatian Science Foundation (project 8065).

References

[1] G. Csurka, C. Dance, L. Fan, J. Willamowski, and C. Bray, "Visual categorization with bags of keypoints," in *ECCV*, 2004. 1

[2] F. Perronnin, J. Sanchez, and T. Mensink, "Improving the Fisher Kernel for Large-Scale Image Classification," in *ECCV*, 2010. 1, 4

Descriptor	Size	UKB-test			ZuBuD			INRIA Holidays		
		NN	FT	ST	NN	FT	ST	NN	FT	ST
FAST, $32 \times 32 \rightarrow 64$	64f	95.8	**85.6**	**90.6**	**96.5**	**81.1**	**86.5**	**56.1**	**40.9**	**47.9**
DoG+FAST, $32 \times 32 \rightarrow 64$	64f	**96.0**	84.7	89.4	95.9	80.3	85.0	56.0	40.4	46.3
DeepCompare[11]+$PCA_{512\rightarrow64}$	64f	93.5	78.8	85.6	96.8	77.7	82.8	52.7	34.6	39.8
MatchNet[12]+$PCA_{4096\rightarrow64}$	64f	80.2	60.4	69.3	94.0	72.9	78.5	46.3	32.3	38.3
DeepDesc [13]	128f	93.8	79.7	86.3	96.0	76.1	81.1	54.7	38.1	43.7
PN-Net [14]	128f	91.7	74.4	81.9	94.7	75.5	81.2	52.0	35.9	41.0
SIFT+$PCA_{128\rightarrow64}$	64f	74.2	52.1	58.0	94.4	71.8	76.7	43.9	26.8	30.0
RFD-R [18]	320b	74.1	53.3	62.3	93.9	72.2	77.8	43.3	26.0	30.2
RFD-G [18]	448b	80.3	58.8	67.0	94.9	73.9	79.0	45.1	28.0	32.6
LDB [34]	256b	61.4	41.2	46.7	83.0	54.8	59.2	42.0	24.8	28.1

(a) Each image was represented with 75 FAST keypoints.

Descriptor	Size	UKB-test			ZuBuD			INRIA Holidays		
		NN	FT	ST	NN	FT	ST	NN	FT	ST
FAST, $32 \times 32 \rightarrow 64$	64f	75.1	58.6	67.2	92.7	71.7	79.2	46.9	28.5	34.5
DoG+FAST, $32 \times 32 \rightarrow 64$	64f	**83.4**	**68.4**	**78.7**	**94.3**	**75.6**	**82.4**	**51.1**	34.3	40.2
DeepCompare[11]+$PCA_{512\rightarrow64}$	64f	72.3	54.7	63.3	92.0	68.4	75.0	50.6	32.4	36.9
MatchNet[12]+$PCA_{4096\rightarrow64}$	64f	58.9	40.9	49.7	90.2	64.9	70.4	41.7	25.4	28.9
DeepDesc [13]	128f	77.2	61.0	70.4	93.2	70.3	77.2	50.6	**35.5**	**42.4**
PN-Net [14]	128f	71.5	53.1	62.0	92.0	66.6	72.3	50.1	31.6	35.3
SIFT+$PCA_{128\rightarrow64}$	64f	40.9	23.2	26.4	78.1	49.8	54.8	29.8	15.3	17.5
RFD-R [18]	320b	44.8	31.2	36.6	79.5	53.3	59.6	43.1	25.2	28.4
RFD-G [18]	448b	53.1	36.4	42.0	85.7	58.0	63.9	45.3	28.8	32.9
LDB [34]	256b	35.8	21.9	26.3	65.1	38.0	42.7	27.3	15.0	18.5

(b) Each image was represented with 75 DoG keypoints.

TABLE II: Retrieval results for different descriptor extraction techniques. First two rows of both tables are for our descriptors.

[3] R. Arandjelovic and A. Zisserman, "All About VLAD," in *CVPR*, 2013. 1, 4

[4] K. Simonyan, O. M. Parkhi, A. Vedaldi, and A. Zisserman, "Fisher Vector Faces in the Wild," in *BMVC*, 2013. 1

[5] Z. Li, D. Gong, X. Li, and D. Tao, "Learning Compact Feature Descriptor and Adaptive Matching Framework for Face Recognition," *IEEE Transaction on Image Processing*, 2015. 1

[6] X. Xiong and F. D. la Torre, "Supervised descent method and its applications to face alignment," in *CVPR*, 2013. 1

[7] M. Brown and D. G. Lowe, "Recognising panoramas," in *ICCV*, 2003. 1

[8] X. Zhou, S. Leonardos, X. Hu, and K. Daniilidis, "3D Shape Estimation from 2D Landmarks: A Convex Relaxation Approach," in *CVPR*, 2015. 1

[9] P. Papadakis, I. Pratikakis, T. Theoharis, G. Passalis, and S. J. Perantonis, "3D object retrieval using an efficient and compact hybrid shape descriptor," in *Eurographics Workshop on 3D Object Retrieval*, 2008. 1

[10] H. Tabia, H. Laga, D. Picard, and P. H. Gosselin, "Covariance descriptors for 3D shape matching and retrieval," in *CVPR*, 2014. 1

[11] S. Zagoruyko and N. Komodakis, "Learning to Compare Image Patches via Convolutional Neural Networks," in *CVPR*, 2015. 1, 2, 5, 6

[12] X. Han, T. Leung, Y. Jia, R. Sukthankar, and A. C. Berg, "MatchNet: Unifying Feature and Metric Learning for Patch-Based Matching," in *CVPR*, 2015. 1, 2, 5

[13] E. Simo-Serra, E. Trulls, L. Ferraz, I. Kokkinos, P. Fua, and F. Moreno-Noguer, "Discriminative Learning of Deep Convolutional Feature Point Descriptors," in *ICCV*, 2015. 1, 2, 5, 6

[14] V. Balntas, E. Johns, L. Tang, and K. Mikolajczyk, "PN-Net: Conjoined Triple Deep Network for Learning Local Image Descriptors," http://arxiv.org/abs/1601.05030. 1, 2, 5, 6

[15] Y. Verdie, K. M. Yi, P. Fua, and V. Lepetit, "TILDE: A Temporally Invariant Learned DEtector," in *CVPR*, 2015. 1

[16] M. Brown, G. Hua, and S. Winder, "Discriminative Learning of Local Image Descriptors," *PAMI*, 2011. 1

[17] T. Trzcinski, M. Christoudias, V. Lepetit, and P. Fua, "Boosting Binary Keypoint Descriptors," in *CVPR*, 2013. 1

[18] 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. 1, 3, 5

[19] D. G. Lowe, "Object recognition from local scale-invariant features," in *ICCV*, 1999. 1, 6

[20] M. Goesele, N. Snavely, B. Curless, H. Hoppe, and S. M. Seitz, "Multi-View Stereo for Community Photo Collections," in *ICCV*, 2007. 1

[21] K. Simonyan, A. Vedaldi, and A. Zisserman, "Learning Local Feature Descriptors Using Convex Optimisation," in *ECCV*, 2012. 1

[22] C. Osendorfer, J. Bayer, S. Urban, and P. van der Smagt, "Convolutional Neural Networks Learn Compact Local Image Descriptors," in *ICONIP (2)*, 2013. 1

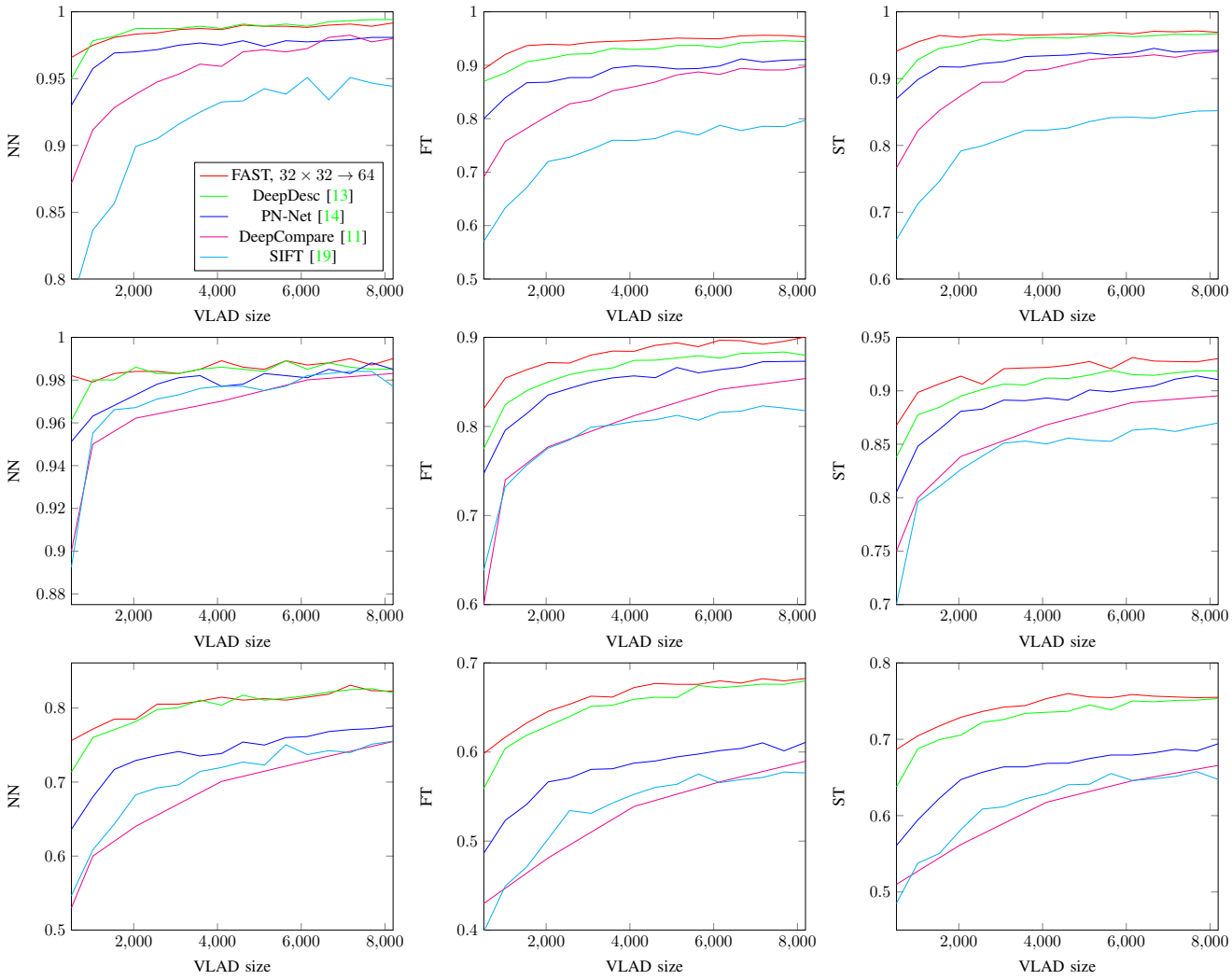

Fig. 4: VLAD-based retrieval results on the UKB-test (first row), ZuBuD (second row) and INRIA Holidays (third row) datasets for varying number of centroids generated with k-means. The legend for all graphs is plotted in the top-left one. The VLAD size is the product of the local descriptor size and the number of centroids.

[23] E. Simo-Serra, E. Trulls, L. Ferraz, I. Kokkinos, and F. Moreno-Noguer, "Fracking Deep Convolutional Image Descriptors," http://arxiv.org/abs/1412.6537, 2015. 1

[24] R. Hadsell, S. Chopra, and Y. LeCun, "Dimensionality Reduction by Learning an Invariant Mapping," in *CVPR*, 2006. 1

[25] T. G. Dietterich, R. H. Lathrop, and T. Lozano-Perez, "Solving the multiple instance problem with axis-parallel rectangles ," *Artificial Intelligence*, 1997. 1

[26] S. Winder and M. Brown, "Learning local image descriptors," in *CVPR*, 2007. 2

[27] M. Paulin, M. Douze, Z. Harchaoui, J. Mairal, F. Perronnin, and C. Schmid, "Local Convolutional Features with Unsupervised Training for Image Retrieval," in *ICCV*, 2015. 1, 3

[28] J. Mairal, P. Koniusz, Z. Harchaoui, and C. Schmid, "Convolutional kernel networks," in *NIPS*, 2014. 1

[29] R. Arandjelović, P. Gronat, A. Torii, T. Pajdla, and J. Sivic, "NetVLAD: CNN architecture for weakly supervised place recognition," in *CVPR*, 2016. 1

[30] C. Wallraven, B. Caputo, and A. Graf, "Recognition with Local Features: the Kernel Recipe," in *ICCV*, 2003. 2

[31] D. Nistér and H. Stewénius, "Scalable Recognition with a Vocabulary Tree," in *CVPR*, 2006. 3, 4

[32] H. Shao, T. Svoboda, and L. V. Gool, "ZuBuD—Zürich building database for image based recognition," ETH Zürich, Tech. Rep., 2003. 3, 4

[33] H. Jegou, M. Douze, and C. Schmid, "Hamming embedding and weak geometric consistency for large scale image search," in *ECCV*, 2008. 3, 4

[34] X. Yang and K.-T. Cheng, "Local Difference Binary for Ultra-fast and Distinctive Feature Description," *PAMI*, 2014. 5

[35] H. Bay, T. Tuytelaars, and L. V. Gool, "SURF: Speeded Up Robust Features," in *ECCV*, 2006. 4