Learning Local Descriptors by Optimizing the Keypoint-Correspondence Criterion: Applications to Face Matching, Learning From Unlabeled Videos and 3D-Shape Retrieval

Nenad Markus, Igor S. Pandzic and Jörgen Ahlberg

N.B.: When citing this work, cite the original publication.
Markus, N., Pandzic, I. S., Ahlberg, J., (2019), Learning Local Descriptors by Optimizing the Keypoint-Correspondence Criterion: Applications to Face Matching, Learning From Unlabeled Videos and 3D-Shape Retrieval, *IEEE Transactions on Image Processing*, 28(1), 279-290.
https://doi.org/10.1109/TIP.2018.2867270

Original publication available at:
https://doi.org/10.1109/TIP.2018.2867270

Learning Local Descriptors by Optimizing the Keypoint-Correspondence Criterion: Applications to Face Matching, Learning from Unlabeled Videos and 3D-Shape Retrieval

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. Additionally, we discuss how to improve the method by incorporating the procedure of mining hard negatives. We also show how can our approach be used to learn convolutional features from unlabeled video signals and 3D models.**

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], 3D model retrieval/matching [9], [10] and visual SLAM [11]. However, despite years of research, there is still room for improvement, as confirmed by recent results based on convolutional neural networks [12], [13], [14], [15], [16], [17], [18]. 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 [19]). There is also research that aims to improve descriptor-matching techniques [20].

Discriminative local descriptors can be learned from annotated keypoint correspondences. This can be used to form a set of matching and non-matching keypoint pairs:

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

The label $l_i \in \{+1, -1\}$ indicates whether keypoints k_{i1} and k_{i2} form a matching or a non-matching pair. See [21], [22], [23], [12], [13], [14], [16], [17], [18] for some recent examples of descriptor-learning methods that use data in this form. Another possibility is to form a set of keypoint triplets:

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

where k_i and k_i^+ match and k_i and k_i^- do not. For example, Balntas et al. [15] 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. [21]. It contains around 1.5M patches cropped around difference of Gaussians (DoG) keypoints [24] 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 [25]. This makes the dataset reliable both for learning local image-patch descriptors from ”handcrafted” features [26], [22], [23] and large models based on convolutional neural networks [27], [12], [13], [28], [14], [15], [17], [18] (the Siamese-network framework of Hadsell et al. [29]). 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 [30] as our ideas are closely related with the field). We denote this weakly-labeled dataset as

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

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: 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.

nenad.markus@fer.hr

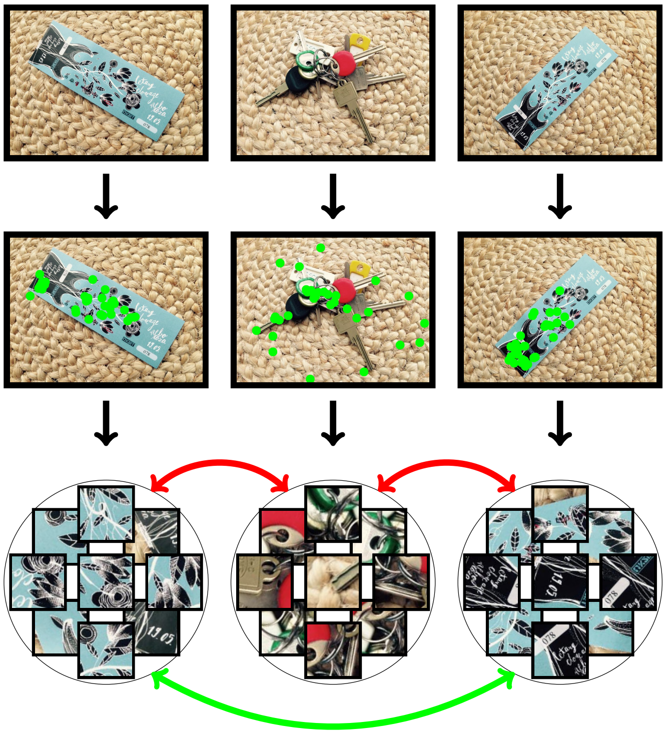

Fig. 1: Each image in the dataset (first row) is processed with a keypoint detector (second row) and transformed into a bag of visual words (third row). Some bags form matching pairs (green arrow, ⟷) and some form non-matching pairs (red arrows, ⟷). On average, matching pairs should contain more corresponding 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., [31], [12], [13], [14], [15], [16]).

In this paper we expand our previous work [32]. There we introduced a method for learning local descriptors from weakly-labeled data and provided preliminary experimental verification of its usefulness. Here we develop the obtained results further and show how to improve the method by incorporating hard-negative mining. We provide strong evidence that it is important to tune the descriptor for the visual appearance of the dataset. This makes our method particularly useful since it enables learning from weaker annotations than traditional descriptor-learning techniques (this is potentially cheaper and more time-efficient). Furthermore, we show how the method can be used in biometric systems and introduce two novel methods for learning local descriptors from unlabeled videos and 3D shapes. We also compare descriptors learned with our method to competing ones on an independent benchmark [33].

II. Related work

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

We would like to mention the work of Paulin et al. [34] 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 [35].

The learning procedure we propose in the next section is related to the one by Arandjelović et al. [36], 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 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.

There are also approaches that learn large, convolutional architectures to directly find correspondences between images [37], [38] or estimate the optical flow [39]. These approaches require a large dataset of annotated correspondences during training. This separates their work from ours since we aim to learn descriptors from weaker annotations.

III. Method

We study how to learn the 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 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 [40]):

$$m_{e,\tau}(K_1, K_2) = \sum_{i=1}^{n} \left[\min_{j=1}^{n} d_{ij}^2 \leq \tau \right], \tag{5}$$

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.,

$$d_{ij} = ||e(k_i) - e(k_j)||_2. \tag{6}$$

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

$$L = \sum \frac{S_{e,\tau}(K, K^-) + \frac{1}{n}}{S_{e,\tau}(K, K^+) + \frac{1}{n}}, \tag{7}$$

where the summation goes over $(K, K^+, K^-) \in \mathcal{D}_{BT}$ (Equation (3)) and $+\frac{1}{n}$ is 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 $x \in \mathbb{R}$:

$$[x \leq \tau] \approx \frac{1}{1 + \exp(\beta(x - \tau))}, \tag{8}$$

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: 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

$$d_{ij}^2 = ||e(k_i) - e(k_j)||_2^2 = 2 - 2e(k_i)^T e(k_j) \tag{9}$$

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

$$\mathbf{S} = \mathbf{E}_1 \mathbf{E}_2^T, \tag{10}$$

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

$$\begin{aligned} \frac{\partial S_{e,\tau}(K_1, K_2)}{\partial \mathbf{E}_1} &= \frac{\partial S_{e,\tau}(K_1, K_2)}{\partial \mathbf{S}} \cdot \mathbf{E}_2 \\ \frac{\partial S_{e,\tau}(K_1, K_2)}{\partial \mathbf{E}_2} &= \left(\frac{\partial S_{e,\tau}(K_1, K_2)}{\partial \mathbf{S}} \right)^T \cdot \mathbf{E}_1 \end{aligned} \tag{11}$$

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 a few hundred lines of Torch code. Another advantage of unit-length descriptors is that this simplifies the selection of the threshold τ: the Euclidean distance between two descriptors falls in the $[0, 2]$ interval (Equation (9)).

We refer to the combination of equations (4)–(8) as **Smoothed Keypoint-mAtching Ratio** (SKAR). We abbreviate the descriptors learned by propagating the gradient through this loss as SKAR descriptors. This notation is used in tables and graphs throughout the experimental part of the paper.

Hard-negative mining. Some descriptor-learning methods (e.g., [14], [17], [18]) incorporate a mechanism of finding the so-called hard negatives: non-matching patches that look sufficiently similar that the descriptor tends to confuse them for matching patches. The hypothesis is that the discriminative power of the descriptor increases when hard negatives are included in the learning process. Our method mines hard negatives when computing the similarity between K and K^- (the $\min$ operator from Equation (5) takes care of this). However, K^- contains only the keypoints extracted from a single non-matching image. The nature of annotated data is most of the time such that it is possible to generate a large number of negative bags for each K. This follows from the same reasoning that is used when mining hard negatives for learning descriptors in a strongly supervised manner. We propose to merge several negative bags into an *augmented negative bag*:

$$K_*^- = \bigcup_j K_j^-, \tag{12}$$

where each K_j^- is one of the non-matching bags to K. In practice, the union in Equation (12) goes only over a random subset of all possible negative bags due to computational and storage reasons. We conjecture that using (K, K^+, K_*^-) in the learning process described in this section can improve the matching performance of the descriptor: since K_*^- is larger than K^-, the $\min$ operator from Equation (5) can extract "harder" non-matching keypoints. This hypothesis is investigated in sections IV and VI.

The following sections describe experiments which show that the proposed learning procedure leads to good results with various diverse keypoint extractors (SIFT/DoG [24], SURF [41], ORB [42], [43]) and in several applications (image retrieval/matching, face verification, learning from unlabeled video signals, 3D-shape recognition).

IV. Learning from weakly-labeled data

The experiments in this section complement the previously presented ones [32]: we repeat the training and validation on a significantly larger number of patches and compare to recent state-of-the-art descriptors.

We use the following datasets for our initial experiments:

- UKB [44] (2500 objects, 4 views each);
- ZuBuD [45] (200 buildings, 5 images each);
- INRIA Holidays [46] (approximately 1500 images of 500 different scenes).

Each image is transformed into a bag of patches by running a keypoint detector over it. This sets up a basis for experimental comparison between different descriptors since we always use

[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 convolutional network that maps a 32×32 RGB patch into a vector of fixed size. It consists of four convolutional layers (given in table above), a fully connected layer that maps the output of the last convolutional layer to 128 neurons and a final L_2 normalization module (i.e., the output vector has unit length). The network has around 250k parameters.

SKAR	O-1	O-6	O-12	S-1	S-6	S-12	S+O-12
O	500	500	500	0	0	0	500
S	0	0	0	500	500	500	500
K_*^- size	1	6	12	1	6	12	12

TABLE II: The nomenclature used for our SKAR descriptors learned on the UKB [44] training partition (first row). Second and third row show the average number of ORB [42] and SIFT (DoG) [24] keypoints extracted per image during training, respectively. The last row shows how many negative bags were used to form K_*^- (Equation (12)).

the same keypoints (location, size[2] and orientation). We extract approximately 5 times more keypoints per image than in our previous paper [32].

The rest of this section is partitioned into four subsections. The SKAR learning process and parameters are described in the next subsection. The last three subsections describe retrieval-based experiments that compare SKAR descriptors to the state of the art.

A. Learning convolutional features with our method

We use a similar descriptor extractor e as in our previous paper [32]. The architecture is specified in Table I. Note that other differentiable architectures could be used as well.

To generate the training data for our method, we partition the UKB dataset into two subsets. The larger subset contains 2200 objects and is used to sample keypoint bag triplets (3). This subset is used for learning and the rest of the UKB dataset (300 objects) is used for validation and testing.

We learn 7 SKAR descriptors on the allotted UKB partition. Each is trained on a different combination of keypoint types and number of negative bags. See Table II for details. We set $\beta = 20$ and $\tau = 0.8$ (see Section III for their meaning), i.e., to same values as in our previous paper [32]. During each training iteration, the loss L (Equation (7)) is approximated by a minibatch of 32 triplets (K, K^+, K_*^-) and the parameters are slowly tuned with `rmsprop` (its learning rate is fixed to 10^{-4}). The whole training process consists of approximately 10 000 such minibatches. This takes around one day for K_*^- of size 12 on a modern machine with 4 GPUs.

B. Matching-based retrieval

As in our previous work [32], we implement a simple visual search engine. The retrieval is based on the number of matching descriptors between the query image and each of the other images from the dataset: the image with more matches is assigned a better rank. Each image of the dataset is used as a query once. We use the ratio criterion, proposed by Lowe [24], to determine whether two descriptors match. This consists of the following steps for each descriptor d from the query bag:

1) in the database bag find two L_2 closest descriptors (denote them as $d_{NN(1)}$ and $d_{NN(2)}$);
2) compute the ratio r: $r = \frac{||d-d_{NN(1)}||_2}{||d-d_{NN(2)}||_2}$;
3) if $r < \tau$, the descriptors d and $d_{NN(1)}$ match.

The threshold τ is selected separately from the set $\{0.7, 0.75, 0.8, 0.85, 0.9\}$ for each descriptor and each dataset to produce the best retrieval results. This approximates the raw discriminative power the descriptor is able to obtain in the ideal case under our tests. Note that this is fair since all descriptors get the same treatment [23].

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.

Tables IIIa and IIIb show the retrieval results for our descriptors from Table II, our descriptor learned on the HPatches dataset [33] (included for completeness, see Section VI for details), recent state-of-the-art ones [18], [14], [15] and three ”handcrafted” baselines (SIFT [24], SURF [41] and intensity order features (IOF) from [47], of which LIOP performed best and is included). Important conclusions:

1) using augmented negative bags (12) significantly improves SKAR descriptors;
2) tuning the SKAR descriptor to the dataset properties might matter quite a lot (e.g., to the used keypoint type).

We can see from Table IIIa that our descriptors trained on ORB keypoints, SKAR O-1, O-6 and O-12, obtain excellent results for ORB keypoints. The SKAR descriptors learned on SIFT (DoG) keypoints, S-1, S-6 and S-12, do better than two baselines, SIFT and SURF, but are outperformed by HardNet [18]. The dataset based on SIFT keypoints (results in Table IIIb) is more difficult than the one based on ORB keypoints, as evidenced by worse performance of all descriptors[3]. We conjecture this is due to the specific methodology we used to crop

[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.

[3] We double-checked our evaluation pipeline for corectness: some descriptors really do obtain such poor results in this setup. This may be due to the scaling parameters we used when cropping patches around SIFT keypoints.

Descriptor	UKB-test			ZuBuD			INRIA Holidays		
	NN	FT	ST	NN	FT	ST	NN	FT	ST
SKAR O-1	90.3	81.1	89.3	94.5	79.2	86.2	49.1	35.9	42.4
SKAR O-6	95.9	88.0	93.1	96.9	84.7	90.4	54.6	40.1	47.1
SKAR O-12	97.8	**89.7**	**93.5**	96.7	85.8	91.1	60.8	45.0	**51.1**
SKAR S-1	92.7	79.9	88.1	91.2	75.1	82.0	53.8	37.4	42.8
SKAR S-6	93.5	81.2	87.9	93.6	75.9	82.9	58.2	41.0	47.0
SKAR S-12	94.9	83.0	89.5	94.7	77.8	83.9	59.0	41.0	46.6
SKAR S+O-12	**98.0**	89.1	93.2	96.9	85.5	90.8	**63.0**	**45.3**	50.9
SKAR-HPatches*	95.9	88.3	93.2	**98.2**	**88.5**	**92.4**	62.7	43.2	47.7
DeepDesc [14]	72.2	57.6	68.7	90.1	70.8	77.4	41.2	26.8	31.8
HardNet [18]	95.4	84.2	90.1	97.4	85.0	89.5	61.0	44.1	49.8
PN-Net [15]	68.6	52.5	62.4	88.5	66.7	73.8	25.9	20.6	27.0
SIFT [24]	74.2	51.2	58.3	93.7	70.6	75.0	21.7	17.3	23.4
SURF [41]	66.0	47.1	56.9	90.9	68.7	73.7	24.6	17.8	23.3
IOF [47]	51.6	35.3	42.5	87.7	66.4	71.6	20.1	12.5	15.3

(a) Each image was represented with $\sim$ 500 ORB keypoints.

Descriptor	UKB-test			ZuBuD			INRIA Holidays		
	NN	FT	ST	NN	FT	ST	NN	FT	ST
SKAR O-1	38.7	31.0	43.0	86.1	60.8	70.0	12.0	8.9	12.5
SKAR O-6	35.9	32.6	46.2	86.3	63.3	72.3	14.1	10.9	16.0
SKAR O-12	60.5	47.9	59.5	92.2	70.2	77.6	17.6	13.9	19.1
SKAR S-1	90.6	78.2	86.1	96.4	80.0	85.9	36.2	25.6	31.9
SKAR S-6	94.8	84.6	89.5	**97.1**	81.3	86.6	43.8	30.5	37.5
SKAR S-12	94.2	84.5	89.4	97.0	80.6	86.3	43.0	30.4	36.9
SKAR S+O-12	**95.1**	**84.8**	**90.0**	96.9	**81.8**	**86.9**	**46.8**	**32.1**	**37.9**
SKAR-HPatches*	64.2	51.4	61.3	93.7	73.7	77.8	14.5	11.4	15.7
DeepDesc [14]	38.5	30.5	40.9	91.0	66.0	71.1	5.5	4.4	7.1
HardNet [18]	65.0	47.8	54.9	92.1	69.2	73.6	9.0	6.9	9.7
PN-Net [15]	5.6	10.4	18.6	80.6	51.2	56.8	1.1	1.2	2.1
SIFT [24]	80.6	67.1	75.8	95.6	76.6	80.6	23.8	21.9	29.9
SURF [41]	7.6	10.1	18.5	73.6	44.7	50.7	0.3	0.4	0.7
IOF [47]	29.4	22.7	30.6	84.9	59.1	64.7	7.6	6.6	10.4

(b) Each image was represented with $\sim$ 500 SIFT (DoG) keypoints.

TABLE III: Retrieval results [%] for different methods. First eight rows of both tables are for our descriptors.

the patches around the detected keypoints (this methodology was initially chosen for ORB and left unchanged for SIFT). However, note that this is not important for our conclusions since all descriptors are tested on the same patches and we are interested in the relative ordering of retrieval scores (not their absolute values). On the experiments with SIFT keypoints (Table IIIb) we can see that SKAR descriptors learned on ORB keypoints exhibit poor retrieval results. However, SKAR descriptors learned on SIFT keypoints obtain best results[4]. It is interesting to note that the SIFT descriptor outperforms other deep-learning approaches [18], [14], [15] for SIFT (DoG) keypoints. We can also see that using augmented negative bags (12) during training improves the discriminative power of SKAR descriptors by a large margin, especially for keypoint types the descriptor was not trained on. Our descriptor learned on the HPatches dataset [33], SKAR-HPatches*, obtains good results for ORB keypoints and is similar to HardNet [18] for SIFT keypoints. Our best descriptor, SKAR S+O-12, obtains excellent results in both setups. This indicates that the model from Table I has sufficient capacity to perform well for both ORB and SIFT (DoG) keypoints. However, it has too be tuned for the appearance of typical patches in the dataset (this is influenced, among other things, with the keypoint detector and its parameters). We suspect that all tested descriptors suffer from this issue[5]. If true, the SKAR learning procedure is a good candidate to mitigate this since it enables learning from much weaker annotations than other approaches that require keypoint correspondences.

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 the extracted 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 2 shows the NN, FT and ST VLAD retrieval scores on three datasets for different local descriptors when using 500 ORB [42] keypoints per image on average. We included only our best descriptor, SKAR S+O-12, to reduce clutter. We see that this descriptor obtains excellent retrieval results. This indicates that the SKAR learning method is valuable even for non-matching tasks.

Figure 3 shows retrieval performance on SURF keypoints. Note that SKAR S+O-12 was not trained for their appearance. The results are similar to those presented in Figure 2: HardNet [18] and our descriptor obtain similar performance on ZuBuD

[4] These descriptors (S-1, S-6 and S-12) also obtain solid results for ORB keypoints (Table IIIa) even though it can be seen as cross-dataset testing. We conjecture that this is due to the fact that our dataset of SIFT patches is a better training set due to its difficulty (results in better generalization).

[5] Rigorously validating this hypothesis is out of the scope of this paper.

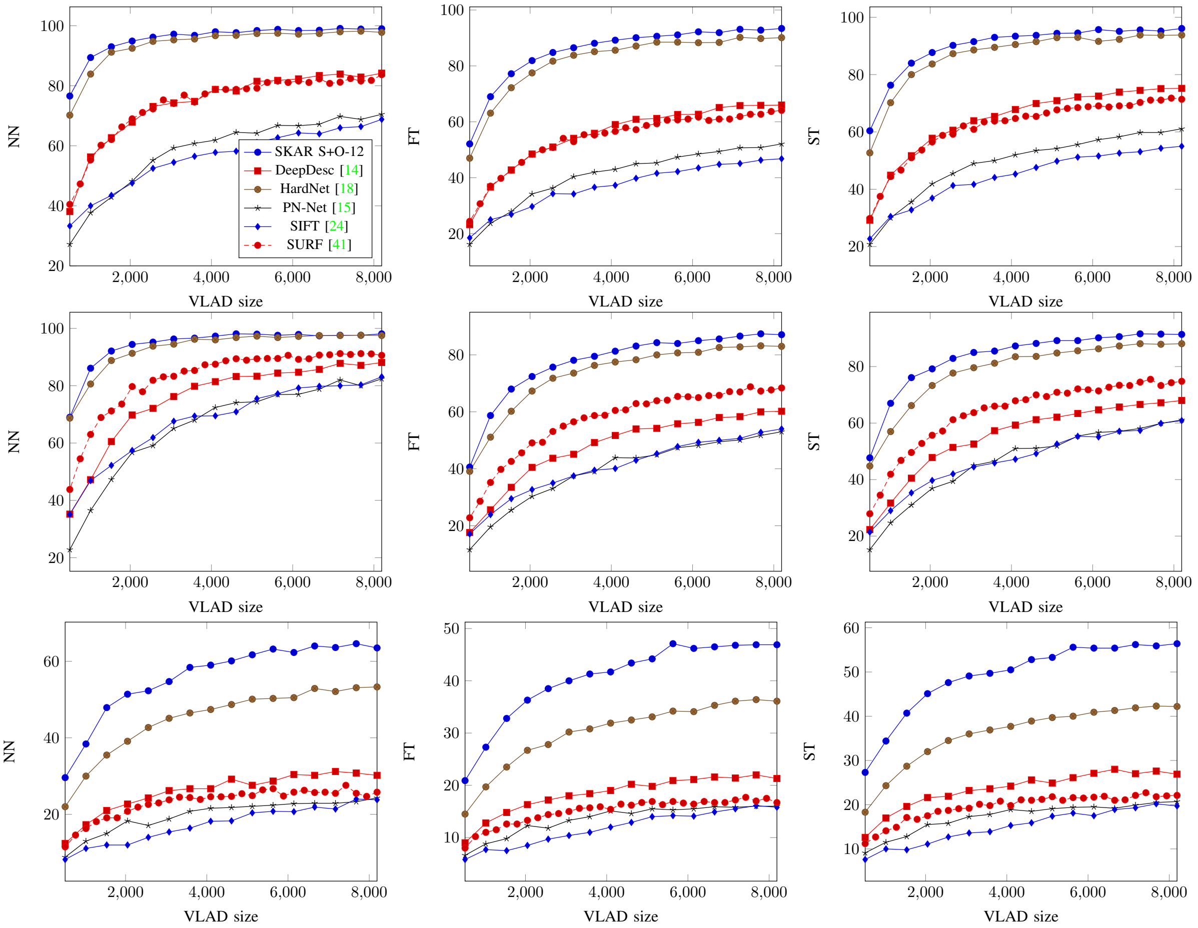

Fig. 2: VLAD-based retrieval results for ~ 500 ORB [42] keypoints per image on the UKB-test (first row), ZuBuD (second row) and INRIA Holidays (third row) datasets for varying number of centroids generated with k-means. The legend for all graphs is plotted in the top-left one. The VLAD size is the product of the local descriptor size and the number of centroids.

[45] and UKB [44] datasets, and our descriptor is the clear winner on INRIA Holidays [46]. Other descriptors show weaker performance across all datasets.

The main motivation for using our method is that it requires simpler training-set annotations. However, one might ask whether this really matters since annotated images abound nowadays and there is substantial experimental evidence that convolutional features are transferable (e.g., [48]). We have already shown that descriptors learned on the dataset of Brown et al. [21] do not perform as good as our descriptors learned on the UKB dataset [44]. We do not attribute this effect to the inherent superiority of our method or to the descriptor-extractor architecture we used in our experiments. We attribute this effect to the features of the training data: the patches extracted from the UKB images resemble testing data more than the patches obtained from the dataset of Brown et al. [21]. We claim that it is important to tune the parameters of the method to the task at hand (i.e., there are no completely transferable features). Note that this is a common opinion in the machine-learning community. Our experiments presented so far agree with this view. We provide further evidence in the next subsection with experiments in face verification.

D. *Training descriptors for a specific task*

Here we show that it is crucial to tune the descriptor for a specific task if high accuracy is desired. The tuning can be achieved with our method which requires simpler data annotation (potentially saves both time and money as opposed to standard descriptor-learning approaches). We show this through an experiment in face recognition/matching.

We use the framework described by Li et al. [49]. The basic idea is that the fine details on the skin of the face[6] are unique for each individual and that this can be used for biometric applications. Given two face images, we determine whether or not they belong to the same person by matching these fine facial features. If we obtain a large number of matches, we can confidently claim that the images belong to the same person. Note that this approach is only applicable to high-resolution face images and that it differs from standard face-recognition methods (e.g., [50]).

We use the Bosphorus database [51] for our experiments. The database contains high-resolution face textures belonging to 105 subjects. Multiple poses, expressions and occlusion

[6] These include face pores, fine wrinkles, hair, moles and small scars.

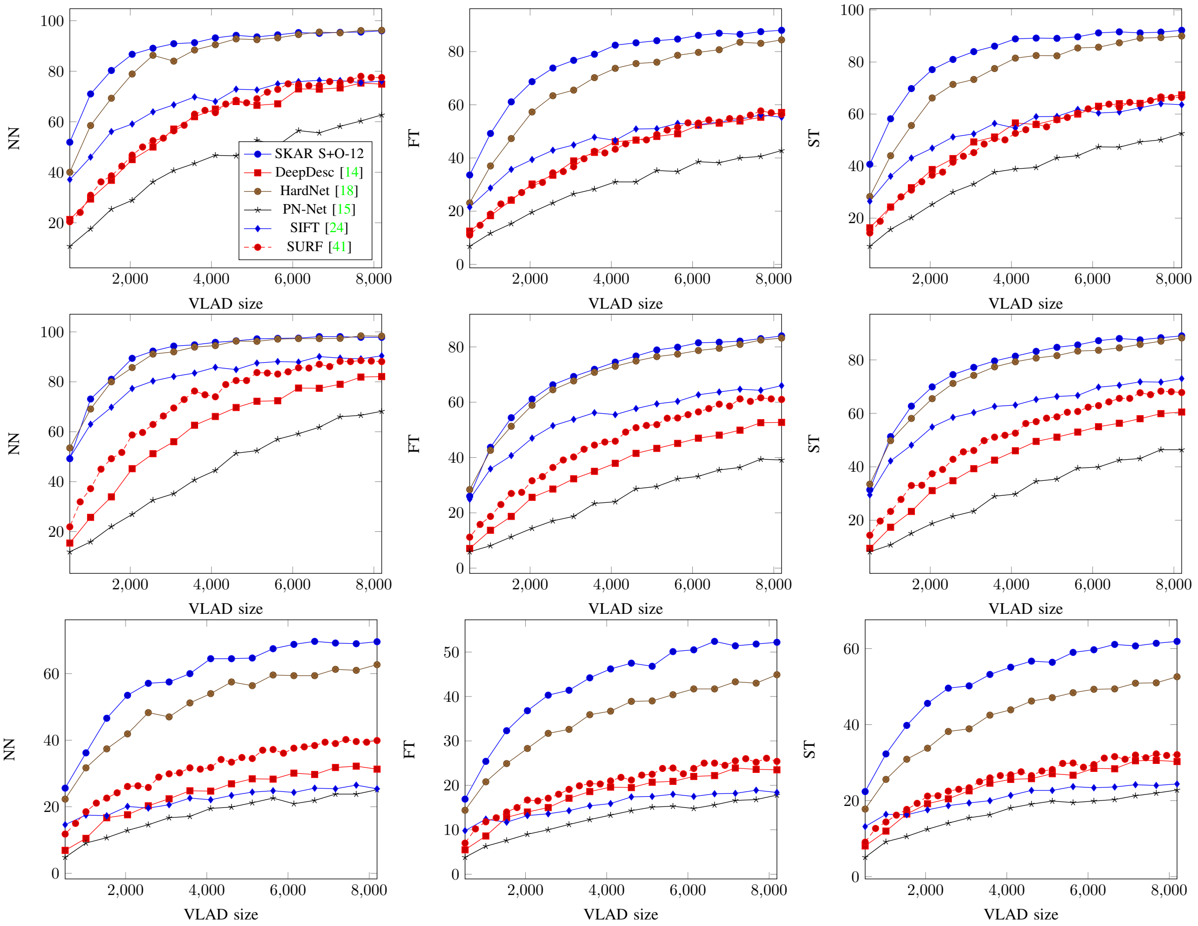

Fig. 3: VLAD-based retrieval results for ~ 1000 SURF [41] keypoints per image on the UKB-test (first row), ZuBuD (second row) and INRIA Holidays (third row) datasets. The graphs are of the same structure as those in Figure 2.

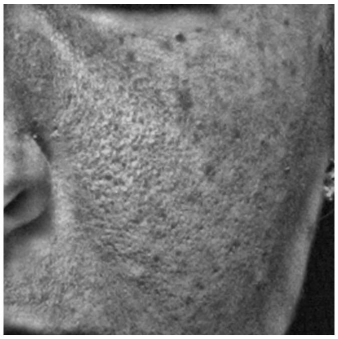

Fig. 4: For each face in the database [51], we crop a region under the left eye and use it in our experiments. Fine facial details are clearly visible after a contrast-normalization step.

conditions are present for each subject. In our experiment, we use only the frontal and near-frontal faces. We use the left cheek, i.e., the region of the face under the left eye, as a visual representation of the face. The region-cropping scheme can be seen in Figure 4. We use the SURF [41] keypoint detector to find 512 keypoints within each region and extract 32×32 patches around each of these keypoints. This pipeline transforms the cheek region into a bag of visual words.

The obtained dataset is partitioned into a training, validation and testing subset. The training and validation subsets are used to learn a descriptor extractor with the same architecture as the one from our previous experiments (see Table I). We experimentally compare these two extractors on the generated testing subset. Note that both extractors have the same architectures. Figure 5 shows the VLAD-based recognition accuracy. We can see that learning the parameters for a particular task leads to large improvements.

The next section demonstrates how the proposed method can be used to learn from unlabeled data. The experimental verification is done through learning convolutional features from unlabeled videos and retrieval of deformable 3D shapes.

V. Learning from unlabeled data

Unsupervised learning is a task of uncovering hidden structure from unlabeled data. The hope is that through this process one obtains useful information or features transferable to other tasks. Modern approaches that aim to learn convolutional features from unlabeled videos and images usually rely on a *simple trick* that exploits the structure within these signals. Wang and Gupta [52] use object tracking in videos: two patches connected by a track should have similar visual

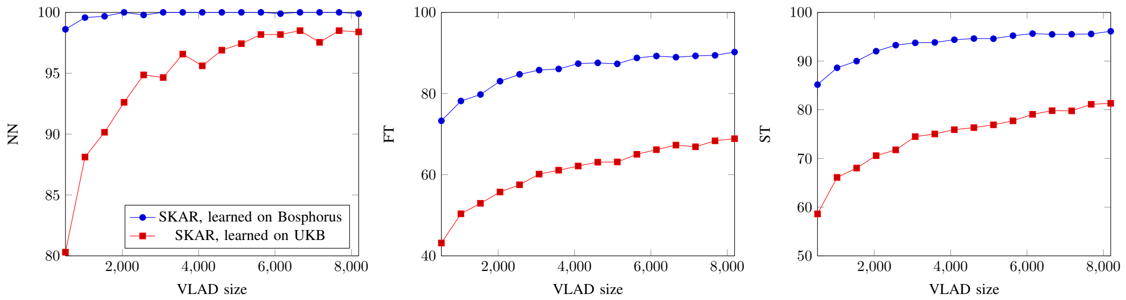

Fig. 5: The VLAD-based recognition accuracy for the testing subset of the Bosphorus dataset [51]. The graphs are of the same structure as those in Figure 2. The superior descriptor was learned on the training subset of the Bosphorus dataset.

representations since they probably belong to same object or object part. Noroozi and Favaro [53] partition an image into a 3×3 matrix of blocks, then randomly permute these blocks and learn a network to reassemble the original image. Misra et al. [54] use a similar idea: they shuffle several frames from a video and then learn a network to put them in a correct temporal order. Methods that learn by predicting the next sample in a sequence also exist [55], [56].

Here we describe two such simple tricks that enable learning of convolutional features from unlabeled data. We experimentally show that these features perform well on relevant tasks. The details are given in the following text.

A. *Learning from unlabeled videos*

Two frames that belong to the same video and are temporally close should have many matching keypoints. Also, two frames from unrelated videos should not have many matching keypoints. These trivial observations let us learn a discriminative local descriptor by using the method proposed in this paper.

We perform experiments on the HUJI EgoSeg dataset [57]. The dataset contains approximately one hundred videos of people doing various activities: biking, running, cooking, sailing, driving, etc. We do not use any labels/annotations associated with the dataset. Also note that the public domain (YouTube) is full of such unlabeled videos.

The training-set generation procedure proceeds as follows. For each video in the dataset, we extract several groups of frames separated apart by approximately 1 minute (obviously, the number of groups depends on the length of the video). Each of these groups consists of 5 frames spaced 250 miliseconds apart. Each frame is transformed into a bag of 512 keypoints with the SURF detector. Two bags that come from the same group form a positive pair (i.e., we assume that they have many matching keypoints). These form a triplet (Equation 3) with any bag of keypoints that comes from some other group of frames. The described procedure enables us to generate a large training dataset for our method. We use this dataset to learn the model specified in Table I. We compare this model to the one learned on the UKB training images [32] (same architecture, also trained on 512 SURF keypoints per bag).

Figure 6 shows VLAD-based retrieval results. We can see that both models perform approximately the same. This is a confirmation that our approach enables learning of useful convolutional features from unlabeled videos. Next, we apply similar ideas to the retrieval of 3D shapes.

B. *Learning 3D-shape retrieval from unlabeled data*

Our plan is to learn a shape-retrieval system from labels (annotations) generated in an automatic way. To achieve this in our framework, we transform the problem into an image-retrieval task by rendering each 3D shape from multiple views [58] and transforming the resulting images into bags of keypoints. Note that some bag pairs are expected to have many matching keypoints between them. Specifically, those that are extracted from related views of the same 3D shape. On the other hand, bag pairs that come from two unrelated shapes should not have many matching keypoints. These observations enable us to generate a dataset for our method (Equation (3)) without any labels associated with each 3D shape.

We use the McGill [59] and PSB [60] datasets for our experiments. The McGill dataset contains 255 shapes with significant part articulations grouped into 10 classes (ants, spiders, crabs, humans, etc.). The PSB dataset is larger: it contains 1814 shapes grouped into 90 classes. Examples from both datasets can bee seen in Figure 7.

We generate the training data from the PSB dataset. We achieve this by rendering 4 groups of 32 randomly chosen views for each shape. Each of these groups is transformed into a bag of keypoints by keeping 512 most salient ORB keypoints [42] extracted from the 32 views belonging to the group. This procedure results in 4×1814 keypoint bags. Two bags form a matching pair if they were generated from the same shape. Together with some unrelated bag, they form a triplet from Equation (3). We use this data to learn a model with architecture specified in Table I.

To test the method, we use the McGill dataset. Each shape is transformed into a keypoint bag with the similar procedure as the one used to generate the training data. The difference is that we render just one group of 32 views per shape (instead of 4). Consequently, there is just one bag of keypoints for each shape. We use this approach to adhere to the standard testing protocol on the McGill dataset. Table IV compares

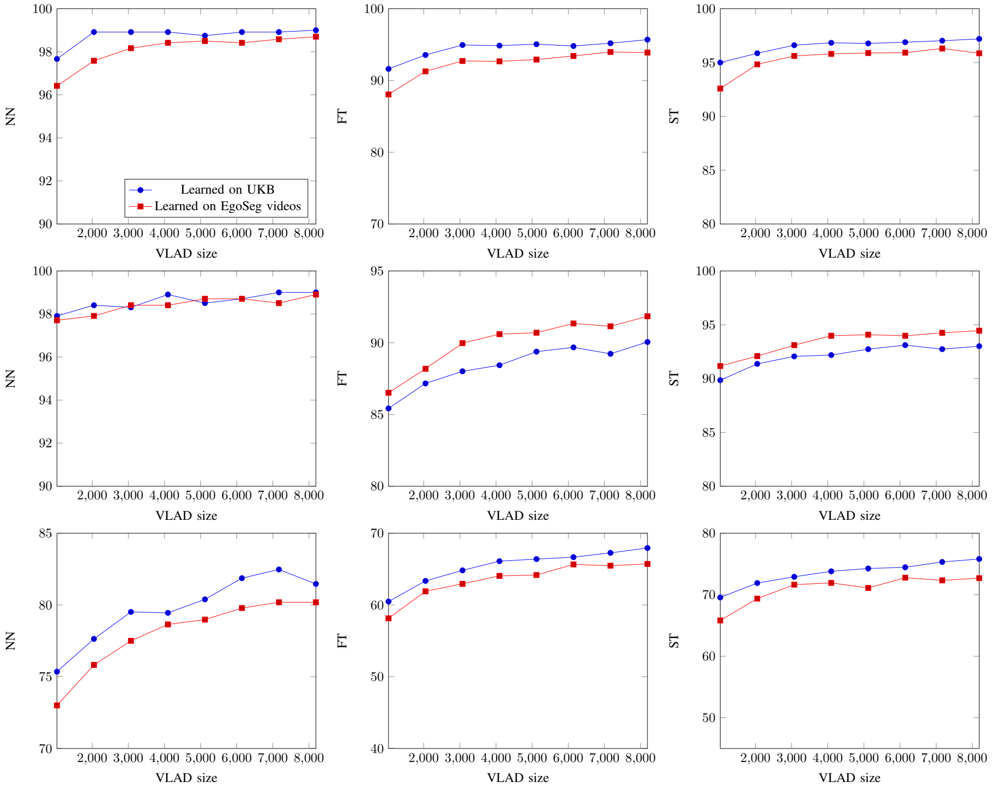

Fig. 6: VLAD-based retrieval results on the UKB-test (first row), ZuBuD (second row) and INRIA Holidays (third row). The graphs are of the same structure as those in Figure 2. We can see that both models perform approximately the same.

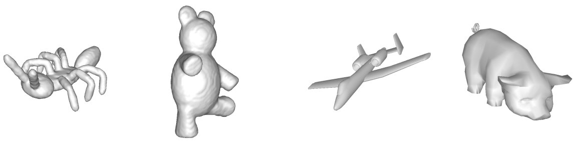

Fig. 7: Samples from the McGill [59] (first two shapes) and PSB [60] (second two shapes) 3D-shape datasets.

Method	NN	FT	ST
SKAR (unlabeled)	96.8	73.9	90.4
GIFT [61]	98.4	90.5	97.3
DeepShape [62]	98.8	78.2	83.4
Covariance descriptors [10]	97.7	73.2	81.8
Graph-based [63]	97.6	74.1	91.1
3D SIFT [64]	97.2	65.8	78.4
VLAT [65]	96.9	65.8	78.1
Hybrid BOW [9]	95.7	63.5	79.0
Hybrid 2D/3D [66]	92.5	55.7	69.8

TABLE IV: Retrieval performance [%] on the McGill dataset.

VLAD-based retrieval scores (64 centroids) achieved by our approach to some other methods from the literature. We can see that our approach obtains comparable scores to all methods except to the ones based on CNNs learned in a supervised way directly for shape retrieval [62], [61]. However, one should note that the DeepShape method [62] used a part of the McGill dataset ($\approx$ 40%) for learning and reported retrieval scores on the rest: not all subsets are of the same difficulty so their results are ambiguous. The GIFT [61] system involves a re-ranking component [67] among other augmentations. Our retrieval pipeline would also benefit from these. However, our main goal was to show that useful convolutional features can be learned from unlabeled 3D shapes: the features perform as well or better than other handcrafted features.

VI. Experiments on the HPatches benchmark

The experiments presented so far have all been designed by us. To have a more fair and independent comparison, here we present experiments on the HPatches benchmark [33]. This benchmark enables an objective comparison of local image descriptors on a large dataset. The images in the dataset contain significant illumination and viewpoint changes. The matches between local keypoints in the corresponding images are provided as ground truth. To simulate the noise that the keypoint detectors introduce in practice, the precomputed keypoints are perturbed in their position, scale and orientation by three increasing noise levels: easy, hard and tough.

The evaluation is done through a strict protocol consisting of three tasks: matching, retrieval and verification. The performance on each of the tasks provides an insight into the descriptor's potential for a certain application (the tasks were designed to imitate typical use cases). The matching task measures how many keypoints does a descriptor match correctly between a target and a reference image. The retrieval task measures how well a descriptor retrieves similar patches from a large collection. The verification task measures how well a descriptor separates positive from negative pairs of patches. The performance on each of the tasks is measured by precision/recall and their variations. For more details, see the paper that introduced the benchmark [33].

For our experiments, we use the same training/testing data partition as the one used in the ECCV2016 workshop ”Local Features: State of the art, open problems and performance evaluation” (this is split ”a” in the HPatches benchmark).

We report the results for four descriptors learned with our method, all having the same architecture. The descriptors SKAR-EgoSeg and SKAR-EgoSeg* were learned on the EgoSeg dataset (see Section V-A for details) without and with mining hard negatives, respectively. The descriptors SKAR-HPatches and SKAR-HPatches* were learned on the training partition of the HPatches dataset[7] without and with mining hard negatives, respectively. The augmented negative bags K_*^- (Equation (12)) were obtained by merging 6 negative bags. Our descriptors are compared to SIFT [24] and four convolutional descriptors [68], [12], [14], [18] learned on the dataset of Borwn et al. [21]. See tables Va, Vb, VI and VII for the results on each of the tasks.

The descriptors learned with our method on the HPatches training dataset, SKAR-HPatches and SKAR-HPatches*, obtain very good results when compared to competing approaches, especially on the image-matching task for which they were explicitly tuned for. By comparison, the SKAR-EgoSeg descriptor obtains significantly worse results on all three tasks, especially in the ”hard” and ”tough” noise-level settings. This is despite having the same architecture as SKAR-HPatches. However, note that SKAR-EgoSeg was learned on SURF keypoints extracted from a dataset unrelated to HPatches testing data which consists of DoG and Hessian keypoints. This provides more evidence that it is crucial to tune the descriptor for the visual appearance and properties of the testing images. Hard-negative mining helps significantly: both SKAR-EgoSeg* and SKAR-Hpatches* outperform their counterparts learned with our original method [32]. Bearing in mind that hard negatives can be easily obtained most of the time, i.e., without any additional data-labeling efforts, it is useful to include the proposed mining procedure in the learning loop since it boosts performance.

As for other convolutional descriptors, TFeat [68], DeepCompare [12] and DeepDesc [14] obtain similar results to SKAR-EgoSeg and outperform SIFT [24] (although not by a large margin). The HardNet descriptor [18], which uses the L2-Net architecture [17] but improves on its learning procedure, obtains strong results, especially on the verification task. We conjecture that high performance on this task is due to the learning procedure that uses data in the form of keypoint triplets with hard-negative mining. However, our strongest descriptor, SKAR-HPatches*, clearly outperforms HardNet on the matching and retrieval tasks. This is despite the fact that the neural network used by our descriptors has 5 times less parameters and being 4 times faster when processing a patch than L2-Net [17] used by HardNet. This demonstrates that it is more important to learn the network for a specific appearance of the dataset than increasing its size for performance gains. Our method makes this task easier to achieve since it requires smaller dataset-annotation efforts than traditional descriptor-learning approaches.

VII. 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 and analyze an algorithm for learning local descriptors from weakly-labeled datasets and discuss the improvements that could be obtained through the process of hard-negative mining. The experiments show that our descriptors compare well to the best available ones and significantly outperform SIFT. We also show how to learn useful convolutional features from unlabeled videos and 3D shapes. Note that these properties of our method enable the learning of local descriptors from datasets with much simpler annotations (or none at all). This potentially saves time and reduces cost when building a computer-vision system that relies on highly discriminative local descriptors. The code is available online as a Git repository (use commit `ab4392d4eb87c25c349d2d6ffb514bee2860f8f0`).

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, 5

[3] R. Arandjelovic and A. Zisserman, “All About VLAD,” in *CVPR*, 2013. 1, 5

[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, 9

[10] H. Tabia, H. Laga, D. Picard, and P. H. Gosselin, “Covariance descriptors for 3D shape matching and retrieval,” in *CVPR*, 2014. 1, 9

[7] We used only image-level annotations when constructing keypoint bags. Also, note that the training and testing sequences are disjoint, so the keypoints on which the evaluation is performed are not included in the training set.

Noise level	Easy	Hard	Tough
SIFT [24]	0.849	0.657	0.512
TFeat [68]	0.912	0.848	0.752
DeepCompare [12]	0.845	0.724	0.609
DeepDesc [14]	0.904	0.830	0.733
HardNet [18]	0.949	0.901	**0.811**
SKAR-EgoSeg	0.896	0.747	0.609
SKAR-EgoSeg*	0.903	0.770	0.637
SKAR-HPatches	0.949	0.900	0.809
SKAR-HPatches*	**0.950**	**0.902**	0.806

(a) Negative patch pairs sampled from different sequences.

Noise level	Easy	Hard	Tough
SIFT [24]	0.783	0.570	0.429
TFeat [68]	0.868	0.781	0.671
DeepCompare [12]	0.785	0.644	0.524
DeepDesc [14]	0.851	0.751	0.640
HardNet [18]	0.923	0.858	**0.752**
SKAR-EgoSeg	0.852	0.678	0.532
SKAR-EgoSeg*	0.866	0.713	0.572
SKAR-HPatches	0.919	0.848	0.734
SKAR-HPatches*	**0.927**	**0.863**	0.751

(b) Negative patch pairs sampled from the same sequence.

TABLE V: Average-precision scores on the verification task of the HPatches benchmark. Each experiment consists of 2×10^5 positive pairs and 10^6 negative pairs.

Noise level	Easy	Hard	Tough
SIFT [24]	0.453	0.193	0.086
TFeat [68]	0.467	0.259	0.131
DeepCompare [12]	0.399	0.199	0.095
DeepDesc [14]	0.430	0.239	0.125
HardNet [18]	0.658	0.479	0.303
SKAR-EgoSeg	0.495	0.226	0.102
SKAR-EgoSeg*	0.507	0.251	0.123
SKAR-HPatches	0.677	0.494	0.313
SKAR-HPatches*	**0.695**	**0.525**	**0.342**

TABLE VI: Mean average precision on the image-matching task of the HPatches benchmark, measured over multiple predefined image pairs.

Noise level	Easy	Hard	Tough
SIFT [24]	0.545	0.264	0.134
TFeat [68]	0.551	0.344	0.189
DeepCompare [12]	0.527	0.309	0.170
DeepDesc [14]	0.561	0.374	0.228
HardNet [18]	0.725	0.572	0.385
SKAR-EgoSeg	0.595	0.319	0.169
SKAR-EgoSeg*	0.605	0.344	0.189
SKAR-HPatches	0.756	0.621	0.440
SKAR-HPatches*	**0.770**	**0.645**	**0.464**

TABLE VII: Mean average precision on the retrieval task of the Hpatches benchmark (10^4 experiemnts of retrieving 5 patches corresponding to a query among 2×10^4 distractors).

[11] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardós, "ORB-SLAM: a versatile and accurate monocular SLAM system," *IEEE Transactions on Robotics*, 2015. 1

[12] S. Zagoruyko and N. Komodakis, "Learning to Compare Image Patches via Convolutional Neural Networks," in *CVPR*, 2015. 1, 2, 10, 11

[13] 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

[14] 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, 3, 4, 5, 6, 7, 10, 11

[15] V. Balntas, E. Johns, L. Tang, and K. Mikolajczyk, "PN-Net: Conjoined Triple Deep Network for Learning Local Image Descriptors," arXiv:1601.05030, 2016. 1, 2, 4, 5, 6, 7

[16] K. M. Yi, E. Trulls, V. Lepetit, and P. Fua, "LIFT: Learned Invariant Feature Transform," in *ECCV*, 2016. 1, 2

[17] Y. Tian, B. Fan, and F. Wu, "L2-net: Deep learning of discriminative patch descriptor in euclidean space," in *CVPR*, 2017. 1, 3, 10

[18] A. Mishchuk, D. Mishkin, F. Radenović, and J. Matas, "Working hard to know your neighbor's margins: Local descriptor learning loss," in *NIPS*, 2017. 1, 3, 4, 5, 6, 7, 10, 11

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

[20] J. W. Bian, W.-Y. Lin, Y. Matsushita, S.-K. Yeung, T.-D. Nguyen, and M.-M. Cheng, "GMS: Grid-based Motion Statistics for Fast, Ultra-robust Feature Correspondence," in *CVPR*, 2017. 1

[21] M. Brown, G. Hua, and S. Winder, "Discriminative Learning of Local Image Descriptors," *PAMI*, 2011. 1, 6, 10

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

[23] 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, 4

[24] D. G. Lowe, "Object recognition from local scale-invariant features," in *ICCV*, 1999. 1, 3, 4, 5, 6, 7, 10, 11

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

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

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

[28] E. Simo-Serra, E. Trulls, L. Ferraz, I. Kokkinos, and F. Moreno-Noguer, "Fracking Deep Convolutional Image Descriptors," arXiv:1412.6537, 2015. 1

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

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

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

[32] N. Markuš, I. S. Pandžić, and J. Ahlberg, "Learning Local Descriptors by Optimizing the Keypoint-Correspondence Criterion," in *ICPR*, 2016. 2, 3, 4, 8, 10

[33] V. Balntas, K. Lenc, A. Vedaldi, and K. Mikolajczyk, "HPatches: A benchmark and evaluation of handcrafted and learned local descriptors," in *CVPR*, 2017. 2, 4, 5, 9, 10

[34] 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. 2

[35] J. Mairal, P. Koniusz, Z. Harchaoui, and C. Schmid, "Convolutional kernel networks," in *NIPS*, 2014. 2

[36] R. Arandjelović, P. Gronat, A. Torii, T. Pajdla, and J. Sivic, "NetVLAD: CNN architecture for weakly supervised place recognition," in *CVPR*, 2016. 2

[37] A. Kanazawa, D. W. Jacobs, and M. Chandraker, "WarpNet: Weakly Supervised Matching for Single-view Reconstruction," in *CVPR*, 2016. 2

[38] C. B. Choy, J. Y. Gwak, and S. Savarese, "Universal Correspondence Network," in *NIPS*, 2016. 2

[39] A. Dosovitskiy, P. Fischer, E. Ilg, P. Häusser, C. Hazirbas, and . Golkov, "FlowNet: Learning Optical Flow with Convolutional Networks," in *ICCV*, 2015. 2

[40] C. Wallraven, B. Caputo, and A. Graf, "Recognition with Local Features: the Kernel Recipe," in *ICCV*, 2003. 3

[41] H. Bay, T. Tuytelaars, and L. V. Gool, "SURF: Speeded Up Robust Features," in *ECCV*, 2006. 3, 4, 5, 6, 7

[42] E. Rublee, V. Rabaud, K. Konolige, and G. R. Bradski, "ORB: An efficient alternative to SIFT or SURF," in *ICCV*, 2011. 3, 4, 5, 6, 8

[43] E. Rosten and T. Drummond, "Machine Learning for High-speed Corner Detection," in *ECCV*, 2006. 3

[44] D. Nistér and H. Stewénius, "Scalable Recognition with a Vocabulary Tree," in *CVPR*, 2006. 3, 4, 6

[45] H. Shao, T. Svoboda, and L. V. Gool, "ZuBuD—Zürich building database for image based recognition," ETH Zürich, Tech. Rep., 2003. 3, 6

[46] H. Jegou, M. Douze, and C. Schmid, "Hamming embedding and weak geometric consistency for large scale image search," in *ECCV*, 2008. 3, 6

[47] Z. Wang, B. Fan, G. Wang, and F. Wu, "Exploring Local and Overall Ordinal Information for Robust Feature Description," *PAMI*, 2016. 4, 5

[48] A. S. Razavian, H. Azizpour, J. Sullivan, and S. Carlsson, "CNN Features Off-the-Shelf: An Astounding Baseline for Recognition," in *CVPR*, 2014. 6

[49] D. Li, H. Zhou, and K.-M. Lam, "High-Resolution Face Verification Using Pore-Scale Facial Features," *IEEE Transaction on Image Processing*, 2015. 6

[50] F. Schroff, D. Kalenichenko, and J. Philbin, "FaceNet: A Unified Embedding for Face Recognition and Clustering," in *CVPR*, 2015. 6

[51] A. Savran, H. D. N. Alyüz, O. Celiktutan, B. Gökberk, B. Sankur, and L. Akarun, "Bosphorus Database for 3D Face Analysis," in *Workshop on Biometrics and Identity Management (BIOID 2008)*, 2008. 6, 7, 8

[52] X. Wang and A. Gupta, "Unsupervised Learning of Visual Representations Using Videos," in *CVPR*, 2015. 7

[53] M. Noroozi and P. Favaro, "Unsupervised learning of visual representations by solving jigsaw puzzles," in *ECCV*, 2016. 8

[54] I. Misra, C. L. Zitnick, and M. Hebert, "Shuffle and Learn: Unsupervised Learning Using Temporal Order Verification," in *ECCV*, 2016. 8

[55] A. van den Oord, N. Kalchbrenner, and K. Kavukcuoglu, "Pixel Recurrent Neural Networks," arXiv:1601.06759, 2016. 8

[56] A. van den Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu, "Wavenet: A Generative Model for Raw Audio," arXiv:1609.03499, 2016. 8

[57] Y. Poleg, C. Arora, and S. Peleg, "Temporal segmentation of egocentric videos," in *CVPR*, 2014. 8

[58] D.-Y. Chen, X.-P. Tian, Y.-T. Shen, and M. Ouhyoung, "On Visual Similarity Based 3D Model Retrieval," *Computer Graphics Forum (EUROGRAPHICS'03)*, 2003. 8

[59] K. Siddiqi, J. Zhang, D. Macrini, A. Shokoufandeh, S. Bouix, and S. Dickinson, "Retrieving articulated 3-D models using medial surfaces," *Machine Vision and Applications*, 2008. 8, 9

[60] P. Shilane, P. Min, M. Kazhdan, and T. Funkhouser, "The Princeton Shape Benchmark," in *Shape Modeling International*, 2004. 8, 9

[61] S. Bai, X. Bai, Z. Zhou, Z. Zhang, and L. J. Latecki, "GIFT: A Real-time and Scalable 3D Shape Search Engine," in *CVPR*, 2016. 9

[62] J. Xie, Y. Fang, F. Zhu, and E. Wong, "DeepShape: Deep Learned Shape Descriptor for 3D Shape Matching and Retrieval," in *CVPR*, 2015. 9

[63] A. Agathos, I. Pratikakis, P. Papadakis, S. J. Perantonis, P. N. Azariadis, , and N. S. Sapidis, "Retrieval of 3D articulated objects using a graph-based representation," in *Eurographics Workshop on 3D Object Retrieval*, 2009. 9

[64] A. Godil and A. I. Wagan, "Salient Local 3D Features for 3D Shape Retrieval," http://arxiv.org/abs/1105.2796, 2011. 9

[65] H. Tabia, D. Picard, H. Laga, and P. H. Gosselin, "Compact vectors of locally aggregated tensors for 3D shape retrieval," in *Eurographics Workshop on 3D Object Retrieval*, 2013. 9

[66] G. Lavoué, "Combination of bag-of-words descriptors for robust partial shape retrieval," *The Visual Computer*, 2012. 9

[67] S. Zhang, M. Yang, T. Cour, K. Yu, and D. N. Metaxas, "Query specific rank fusion for image retrieval," *PAMI*, 2015. 9

[68] V. Balntas, E. Riba, D. Ponsa, and K. Mikolajczyk, "Learning local feature descriptors with triplets and shallow convolutional neural networks," in *BMVC*, 2016. 10, 11

Nenad Markuš received his B.Sc., M.Sc. and Ph.D. degrees from the University of Zagreb in 2009, 2011 and 2017, respectively. His scientific interests include problems in computer vision, pattern recognition and machine learning. Besides various research and teaching activities, he also took part in the development of several commercial computer-vision systems.

Igor S. Pandžić is a Professor at the Faculty of Electrical Engineering and Computing, University of Zagreb, Croatia. He teaches undergraduate and postgraduate courses in the fields of virtual environments and communications. His main research interests are in the field of computer graphics and, more recently, computer vision, with particular interest in face analysis and animation and strong focus on applications of these technologies. Igor also worked on networked collaborative virtual environments, computer generated film production and parallel computing. He published five books and around 100 papers on these topics.

Jörgen Ahlberg received his M.Sc. degree in Computer Science and Engineering in 1996 and his Ph.D. in Electrical Engineering in 2002, both from Linköping University, Sweden. He then held positions as scientist and research leader at FOI, the Swedish Defence Research Agency for nine years. He is currently an Adjunct Senior Lecturer at Linköping University and runs R&D projects at Visage Technologies, Termisk Systemteknik, and Glana Sensors, companies specializing in visual, thermal and hyperspectral computer vision respectively. Research interests, in the general area of image analysis and vision, includes tracking and analysis of facial images as well as automatic detection, recognition, and tracking in thermal and hyperspectral systems. He has published more than 40 scientific papers, of which four award-winning.