ELF: Embedded Localisation of Features in pre-trained CNN

Assia Benbihi[1,2], Matthieu Geist[3], and Cédric Pradalier[1,4]

[1]UMI2958 GeorgiaTech-CNRS
[2]Centrale Supélec, Université Paris Saclay, Metz
[3]Google Research, Brain Team
[4]GeorgiaTech Lorraine

Abstract

This paper introduces a novel feature detector based only on information embedded inside a CNN trained on standard tasks (e.g. classification). While previous works already show that the features of a trained CNN are suitable descriptors, we show here how to extract the feature locations from the network to build a detector. This information is computed from the gradient of the feature map with respect to the input image. This provides a saliency map with local maxima on relevant keypoint locations. Contrary to recent CNN-based detectors, this method requires neither supervised training nor finetuning. We evaluate how repeatable and how 'matchable' the detected keypoints are with the repeatability and matching scores. Matchability is measured with a simple descriptor introduced for the sake of the evaluation. This novel detector reaches similar performances on the standard evaluation HPatches dataset, as well as comparable robustness against illumination and viewpoint changes on Webcam and photo-tourism images. These results show that a CNN trained on a standard task embeds feature location information that is as relevant as when the CNN is specifically trained for feature detection.

1 Introduction

Feature extraction, description and matching is a recurrent problem in vision tasks such as Structure from Motion (SfM), visual SLAM, scene recognition and image retrieval. The extraction consists in detecting image keypoints, then the matching pairs the nearest keypoints based on their descriptor distance. Even though hand-crafted solutions, such as SIFT [21], prove to be successful, recent breakthroughs on local feature detection and description rely on supervised deep-learning methods [12, 28, 44]. They detect keypoints on saliency maps learned by a Convolutional Neural Network (CNN), then compute descriptors using another CNN or a separate branch of it. They all require strong supervision and
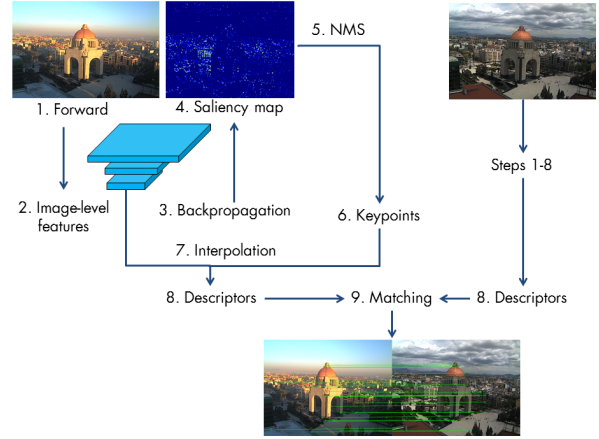

Figure 1: (1-6) Embedded Detector: Given a CNN trained on a standard vision task (classification), we backpropagate the feature map back to the image space to compute a saliency map. It is thresholded to keep only the most informative signal and keypoints are the local maxima. (7-8): simple-descriptor.

complex training procedures: [44] requires ground-truth matching keypoints to initiate the training, [28] needs the ground-truth camera pose and depth maps of the images, [12] circumvents the need for ground-truth data by using synthetic one but requires a heavy domain adaptation to transfer the training to realistic images. All these methods require a significant learning effort. In this paper, we show that a trained network already embeds enough information to build State-of-the-Art (SoA) detector and descriptor.

The proposed method for local feature detection needs only a CNN trained on standard task, such as ImageNet [11] classification, and no further training. The detector, dubbed ELF, relies on the features learned by such a CNN and extract their locations from the feature map gradients. Previous work already highlights that trained CNN features are relevant descriptors [13] and recent works [6, 15, 34] specifically train CNN to produce features suitable for keypoint description. However, no existing ap-

proach uses a pre-trained CNN for feature detection.

ELF computes the gradient of a trained CNN feature map with respect to *w.r.t* the image: this outputs a saliency map with local maxima on keypoint positions. Trained detectors learn this saliency map with a CNN whereas we extract it with gradient computations. This approach is inspired by [35] which observes that the gradient of classification scores *w.r.t* the image is similar to the image saliency map. ELF differs in that it takes the gradient of feature maps and not the classification score contrary to existing work exploiting CNN gradients [33, 37, 38, 40]. These previous works aim at visualising the learning signal for classification specifically whereas ELF extracts the feature locations. The extracted saliency map is then thresholded to keep only the most relevant locations and standard Non-Maxima Suppression (NMS) extracts the final keypoints (Figure 2).
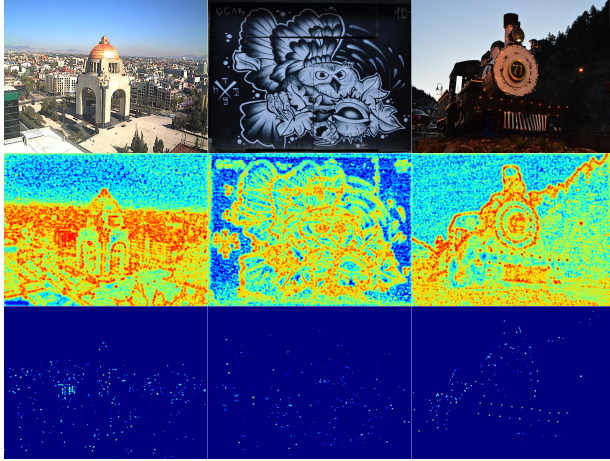

Figure 2: Saliency maps thresholding to keep only the most informative location. Top: original image. (Left-Right: Webcam [43], HPatches [5], COCO[20]) Middle: blurred saliency maps. Bottom: saliency map after threshold. (Better seen on a computer.)

ELF relies only on six parameters: $2{\times}2$ Gaussian blur parameters for the automatic threshold level estimation and for the saliency map denoising; and two parameters for the (NMS) window and the border to ignore. Detection only requires one forward and one backward passes and takes $\sim$0.2s per image on a simple Quadro M2200, which makes it suitable for real-time applications.

ELF is compared to individual detectors with standard *repeatability* [26] but results show that this metric is not discriminative enough. Most of the existing detectors can extract keypoints repeated across images with similar repeatability scores. Also, this metric does not express how 'useful' the detected keypoints are: if we sample all pixels as keypoints, we reach 100% of *rep.* but the matching may not be perfect if many areas look alike. Therefore,

the detected keypoints are also evaluated on how 'matchable' they are with the *matching score* [26]. This metric requires to describe the keypoints so we define a simple descriptor: it is based on the interpolation of a CNN feature map on the detected keypoints, as in [12]. This avoids biasing the performance by choosing an existing competitive descriptor. Experiments show that even this simple descriptor reaches competitive results which comforts the observation of [13], on the relevance of CNN features as descriptors. More details are provided section 4.1.

ELF is tested on five architectures: three classification networks trained on ImageNet classification: AlexNet, VGG and Xception [9, 18, 36], as well as SuperPoint [12] and LF-Net [28] descriptor networks. Although outside the scope of this paper, this comparison provides preliminary results of the influence of the network architecture, task and training data on ELF's performance. Metrics are computed on HPatches [5] for generic performances. We derive two auxiliary datasets from HPatches to study scale and rotation robustness. Light and 3D viewpoint robustness analysis are run on the Strecha, Webcam and datasets [39, 43]. These extensive experiments show that ELF is on par with other sparse detectors, which suggests that the feature representation and location information learnt by a CNN to complete a vision task is as relevant as when the CNN is specifically trained for feature detection. We additionally test ELF's robustness on 3D reconstruction from images in the context of the CVPR 2019 Image Matching challenge [1]. Once again, ELF is on par with other sparse methods even though denser methods, e.g. [12], are more appropriate for such a task. Our contributions are the following:

- We show that a CNN trained on a standard vision task embeds feature location in the feature gradients. This information is as relevant for feature detection as when a CNN is specifically trained for it.

- We define a systematic method for local feature detection. Extensive experiments show that ELF is on par with other SoA deep trained detectors. They also update the previous result from [13]: self-taught CNN features provide SoA descriptors in spite of recent improvements in CNN descriptors [10].

- We release the python-based evaluation code to ease future comparison together with ELF code[1]. The introduced robustness datasets are also made public [2].

2   Related work

Early methods rely on hand-crafted detection and description : SIFT [21] detects 3D spatial-scale keypoints on dif-

---

[1]ELF code:`https://github.com/ELF-det/elf`
[2]Rotation and scale dataset: `https://bit.ly/31RAh1S`

ference of gaussians and describes them with a 3D Histogram Of Gradients (HOG). SURF [7] uses image integral to speed up the previous detection and uses a sum of Haar wavelet responses for description. KAZE [4] extends the previous multi-scale approach by detecting features in non-linear scale spaces instead of the classic Gaussian ones. ORB [30] combines the FAST [29] detection, the BRIEF [8] description, and improves them to make the pipeline scale and rotation invariant. MSER-based detector hand-crafts desired invariance properties for keypoints, and designs a fast algorithm to detect them [23]. Even though these hand-crafted methods have proven to be successful and to reach state-of-the-art performance for some applications, recent research focus on learning-based methods.

One of the first learned detector is TILDE [43], trained under drastic changes of light and weather on the Webcam dataset. They use supervision to learn saliency maps which maxima are keypoint locations. Ground-truth saliency maps are generated with 'good keypoints': they use SIFT and filter out keypoints that are not repeated in more than 100 images. One drawback of this method is the need for supervision that relies on another detector. However, there is no universal explicit definition of what a good keypoint is. This lack of specification inspires Quad-Networks [31] to adopt an unsupervised approach: they train a neural network to rank keypoints according to their robustness to random hand-crafted transformations. They keep the top/bottom quantile of the ranking as keypoints. ELF is similar in that it does not requires supervision but differs in that it does not need to further train the CNN.

Other learned detectors are trained within full detection/description pipelines such as LIFT [44], SuperPoint [12] and LF-Net [28]. LIFT contribution lies in their original training method of three CNNs. The detector CNN learns a saliency map where the most salient points are keypoints. They then crop patches around these keypoints, compute their orientations and descriptors with two other CNNs. They first train the descriptor with patches around ground-truth matching points with contrastive loss, then the orientation CNN together with the descriptor and finally with the detector. One drawback of this method is the need for ground-truth matching keypoints to initiate the training. In [12], the problem is avoided by pre-training the detector on a synthetic geometric dataset made of polygons on which they detect mostly corners. The detector is then finetuned during the descriptor training on image pairs from COCO [20] with synthetic homographies and the correspondence contrastive loss introduced in [10]. LF-Net relies on another type of supervision: it uses ground-truth camera poses and image depth maps that are easier to compute with laser or standard SfM than ground-truth matching keypoints. Its training pipeline builds over LIFT and employs the pro-

jective camera model to project detected keypoints from one image to the other. These keypoint pairs form the ground-truth matching points to train the network. ELF differs in that the CNN model is already trained on a standard task. It then extracts the relevant information embedded inside the network for local feature detection, which requires no training nor supervision.

The detection method of this paper is mainly inspired from the initial observation in [35]: given a CNN trained for classification, the gradient of a class score *w.r.t* the image is the saliency map of the class object in the input image. A line of works aims at visualizing the CNN representation by inverting it into the image space through optimization [14, 22]. Our work differs in that we backpropagate the feature map itself and not a feature loss. Following works use these saliency maps to better understand the CNN training process and justify the CNN outputs. Efforts mostly focus on the gradient definitions [37, 38, 40, 46]. They differ in the way they handle the backpropagation of the non-linear units such as Relu. Grad-CAM [33] introduces a variant where they fuse several gradients of the classification score *w.r.t* feature maps and not the image space. Instead, ELF computes the gradient of the feature map, and not a classification score, *w.r.t* the image. Also we run simple backpropagation which differs in the non-linearity handling: all the signal is backpropagated no matter whether the feature maps or the gradients are positive or not. Finally, as far as we know, this is the first work to exploit the localisation information present in these gradients for feature detection.

The simple descriptor introduced for the sake of the matchability evaluation is taken from UCN [10]. Given a feature map and the keypoints to describe, it interpolates the feature map on the keypoints location. Using a trained CNN for feature description is one of the early applications of CNN [13]. Later, research has taken on specifically training the CNN to generate features suitable for keypoint matching either with patch-based approaches, among which [15, 24, 34, 45], or image-based approaches [10, 41]. We choose the description method from UCN [10], also used by SuperPoint, for its complexity is only $O(1)$ compared to patch-based approaches that are $O(N)$ with N the number of keypoints. We favor UCN to InLoc [41] as it is simpler to compute. The motivation here is only to get a simple descriptor easy to integrate with all detectors for fair comparison of the *detector* matching performances. So we overlook the description performance.

3 Method

This section defines ELF, a detection method valid for any trained CNN. Keypoints are local maxima of a saliency

map computed as the feature gradient *w.r.t* the image. We use the data adaptive Kapur method [17] to automatically threshold the saliency map and keep only the most salient locations, then run NMS for local maxima detection.
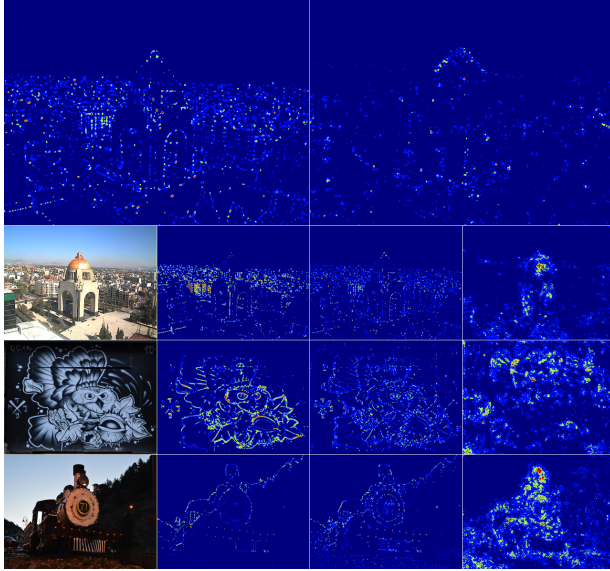

Figure 3: (Bigger version Figure 15.) Saliency maps computed from the feature map gradient $\left| {}^T F^l(x) \cdot \frac{\partial F^l}{\partial \mathbf{I}} \right|$. Enhanced image contrast for better visualisation. Top row: gradients of VGG $pool_2$ and $pool_3$ show a loss of resolution from $pool_2$ to $pool_3$. Bottom: $(pool_i)_{i \in [1,2,5]}$ of VGG on Webcam, HPatches and Coco images. Low level saliency maps activate accurately whereas higher saliency maps are blurred.

3.1 Feature Specific Saliency

We generate a saliency map that activates on the most informative image region for a specific CNN feature level l. Let $\mathbf{I}$ be a vector image of dimension $D_I = H_I \cdot W_I \cdot C_I$. Let F^l be a vectorized feature map of dimension $D_F = H_l \cdot W_l \cdot C_l$. The saliency map S^l, of dimension D_I, is $S^l(\mathbf{I}) = \left| {}^t F^l(\mathbf{I}) \cdot \nabla_I F^l \right|$, with $\nabla_I F^l$ a $D_F \times D_I$ matrix.

The saliency activates on the image regions that contribute the most to the feature representation $F^l(\mathbf{I})$. The term $\nabla_I F^l$ explicits the correlation between the feature space of F^l and the image space in general. The multiplication by $F^l(\mathbf{I})$ applies the correlation to the features $F^l(\mathbf{I})$ specifically and generate a visualisation in image space $S^l(\mathbf{I})$. From a geometrical point of view, this operation can be seen as the projection $\nabla_I F^l$ of a feature signal $F^l(\mathbf{I})$ into the image space. From a signal processing approach, $F^l(\mathbf{I})$ is an input signal filtered through $\nabla_I F^l$ into the image space. If $C_I > 1$, S^l is converted into a grayscale image by averaging it across channels.

3.2 Feature Map Selection

We provide visual guidelines to choose the feature level l so that F^l still holds high resolution localisation information while providing a useful high-level representation.

CNN operations such as convolution and pooling increase the receptive field of feature maps while reducing their spatial dimensions. This means that F^l has less spatial resolution than F^{l-1} and the backpropagated signal S^l ends up more spread than S^{l-1}. This is similar to when an image is too enlarged and it can be observed in Figure 3, which shows the gradients of the VGG feature maps. On the top row, $pool_2$'s gradient (left) better captures the location details of the dome whereas $pool_3$'s gradient (right) is more spread. On the bottom rows, the images lose their resolution as we go higher in the network. Another consequence of this resolution loss is that small features are not embedded in F^l if l is too high. This would reduce the space of potential keypoint to only large features which would hinder the method. This observation motivates us to favor low-level feature maps for feature detection. We chose the final F^l by taking the highest l which provides accurate localisation. This is visually observable by sparse high intensity signal contrary to the blurry aspect of higher layers.

3.3 Automatic Data-Adaptive Thresholding

The threshold is automatic and adapts to the saliency map distribution to keep only the most informative regions. Figure 2 shows saliency maps before and after thresholding using Kapur's method [17], which we briefly recall below. It chooses the threshold to maximize the information between the image background and foreground *i.e.* the pixel distribution below and above the threshold. This method is especially relevant in this case as it aims at maintaining as much information on the distribution above the threshold as possible. This distribution describes the set of local maxima among which we choose our keypoints. More formally, for an image $\mathbf{I}$ of N pixels with n sorted gray levels and $(f_i)_{i \in n}$ the corresponding histogram, $p_i = \frac{f_i}{N}$ is the empirical probability of a pixel to hold the value f_i. Let $s \in n$ be a threshold level and A, B the empirical background and foreground distributions. The level s is chosen to maximize the information between A and B and the threshold value is set to f_s: $A = \left(\frac{p_i}{\sum_{i<s} p_i} \right)_{i<s}$ and $B = \left(\frac{p_i}{\sum_{i>=s} p_i} \right)_{i>s}$. For better results, we blur the image with a Gaussian of parameters $(\mu_{thr}, \sigma_{thr})$ before computing the threshold level.

Once the threshold is set, we denoise the image with a second Gaussian blur of parameters $(\mu_{noise}, \sigma_{noise})$ and run standard NMS (the same as for SuperPoint) where we iteratively select decreasing global maxima while ensuring that their nearest neighbor distance is higher than the

window $w_{\text{NMS}} \in \mathbb{N}$. Also we ignore the $b_{\text{NMS}} \in \mathbb{N}$ pixels around the image border.

3.4 Simple descriptor

As mentioned in the introduction, the repeatability score does not discriminate among detectors anymore. So they are also evaluated on how 'matchable' their detected keypoints are with the matching score. To do so, the ELF detector is completed with a simple descriptor inspired by SuperPoint's descriptor. The use of this simple descriptor over existing competitive ones avoids unfairly boosting ELF's perfomance. Inspired by SuperPoint, we interpolate a CNN feature map on the detected keypoints. Although simple, experiments show that this simple descriptor completes ELF into a competitive feature detection/description method.

The feature map used for description may be different from the one for detection. High-level feature maps have wider receptive field hence take higher context into account for the description of a pixel location. This leads to more informative descriptors which motivates us to favor higher level maps. However we are also constrained by the loss of resolution previously described: if the feature map level is too high, the interpolation of the descriptors generate vector too similar to each other. For example, the VGG $pool_4$ layer produces more discriminative descriptors than $pool_5$ even though $pool_5$ embeds information more relevant for classification. Empirically we observe that there exists a layer level l' above which the description performance stops increasing before decreasing. This is measured through the matching score metric introduced in [26]. The final choice of the feature map is done by testing some layers $l' > l$ and select the lowest feature map before the descriptor performance stagnates.

The compared detectors are evaluated with both their original descriptor and this simple one. We detail the motivation behind this choice: detectors may be biased to sample keypoints that their respective descriptor can describe 'well' [44]. So it is fair to compute the matching score with the original detector/descriptor pairs. However, a detector can sample 'useless points' (e.g. sky pixels for 3d reconstructions) that its descriptor can characterise 'well'. In this case, the descriptor 'hides' the detector default. This motivates the integration of a common independent descriptor with all detectors to evaluate them. Both approaches are run since each is as fair as the other.

4 Experiments

This section describes the evaluation metrics and datasets as well as the method's tuning. Our method is compared to detectors with available public code: the fully hand-crafted SIFT [21], SURF [7], ORB [30], KAZE [4], the learning-based LIFT [44], SuperPoint [12], LF-Net [28], the individual detectors TILDE [43], MSER [23].

4.1 Metrics

We follow the standard validation guidelines [26] that evaluates the detection performance with *repeatability (rep)*. It measures the percentage of keypoints common to both images. We also compute the *matching score (ms)* as an additional *detector* metric. It captures the percentage of keypoint pairs that are nearest neighbours in both image space and descriptor space i.e. the ratio of keypoints correctly matched. For fair completeness, the mathematical definitions of the metrics are provided in Appendix and their implementation in the soon-to-be released code.

A way to reach perfect *rep* is to sample all the pixels or sample them with a frequency higher than the distance threshold ϵ_{kp} of the metric. One way to prevent the first flaw is to limit the number of keypoints but it does not counter the second. Since detectors are always used together with descriptors, another way to think the detector evaluation is: *'a good keypoint is one that can be discriminatively described and matched'*. One could think that such a metric can be corrupted by the descriptor. But we ensure that a detector flaw cannot be hidden by a very performing descriptor with two guidelines. One experiment must evaluate all detector with one fixed descriptor (the simple one defined in 3.4). Second, *ms* can never be higher than *rep* so a detector with a poor *rep* leads to a poor *ms*.

Here the number of detected keypoints is limited to 500 for all methods. As done in [12, 28], we replace the overlap score in [26] to compute correspondences with the 5-pixel distance threshold. Following [44], we also modify the matching score definition of [26] to run a greedy bipartite-graph matching on all descriptors and not just the descriptor pairs for which the distance is below an arbitrary threshold. We do so to be able to compare all state-of-the-art methods even when their descriptor dimension and range vary significantly. (More details in Appendix.)

4.2 Datasets

All images are resized to the 480×640 pixels and the image pair transformations are rectified accordingly.

General performances. The HPatches dataset [5] gathers a subset of standard evaluation images such as DTU and OxfordAffine [2, 25]: it provides a total of 696 images, 6 images for 116 scenes and the corresponding homographies between the images of a same scene. For 57 of these scenes, the main changes are photogrammetric and the remaining 59 show significant geometric deformations due to viewpoint changes on planar scenes.

Figure 4: Left-Right: HPatches: planar viewpoint. Webcam: light. HPatches: rotation. HPatches: scale. Strecha: 3D viewpoint.

Illumination Robustness. The Webcam dataset [43] gathers static outdoor scenes with drastic natural light changes contrary to HPatches which mostly holds artificial light changes in indoor scenes.

Rotation and Scale Robustness. We derive two datasets from HPatches. For each of the 116 scenes, we keep the first image and rotate it with angles from $0°$ to $210°$ with an interval of $40°$. Four zoomed-in version of the image are generated with scales $[1.25, 1.5, 1.75, 2]$. We release these two datasets together with their ground truth homographies for future comparisons.

3D Viewpoint Robustness. We use three Strecha scenes [39] with increasing viewpoint changes: *Fountain, Castle entry, Herzjesu-P8*. The viewpoint changes proposed by HPatches are limited to planar scenes which does not reflect the complexity of 3D structures. Since the ground-truth depths are not available anymore, we use COLMAP [32] 3D reconstruction to obtain ground-truth scaleless depth. We release the obtained depth maps and camera poses together with the evaluation code. ELF robustness is additionally tested in the CVPR19 Image Matching Challenge [1] (see results sections).

4.3 Baselines

We describe the rationale behind the evaluation. The tests run on a QuadroM2200 with Tensorflow 1.4, Cuda8, Cudnn6 and Opencv3.4. We use the OpenCV implementation of SIFT, SURF, ORB, KAZE, MSER with the default parameters and the author's code for TILDE, LIFT, SuperPoint, LF-Net with the provided models and parameters. When comparing detectors in the feature matching pipeline, we measure their matching score with both their original descriptor and ELF simple descriptor. For MSER and TILDE, we use the VGG simple descriptor.

Architecture influence. ELF is tested on five networks: three classification ones trained on ImageNet (AlexNet, VGG, Xception [9, 18, 36]) as well as the trained SuperPoint's and LF-Net's descriptor ones. We call each variant with the network's names prefixed with

ELF as in saliency. The paper compares the influence of i) architecture for a fixed task (ELF-AlexNet [18] *vs.* ELF-VGG [36] *v.s.* ELF-Xception [9]), ii) the task (ELF-VGG *vs.* ELF-SuperPoint (SP) descriptor), iii) the training dataset (ELF-LFNet on phototourism *vs.* ELF-SP on MS-COCO). This study is being refined with more independent comparisons of tasks, datasets and architectures soon available in a journal extension.

We use the author's code and pre-trained models which we convert to Tensorflow [3] except for LF-Net. We search the blurring parameters $(\mu_{thr}, \sigma_{thr})$, $(\mu_{noise}, \sigma_{noise})$ in the range $[\![3, 21]\!]^2$ and the NMS parameters (w_{NMS}, b_{NMS}) in $[\![4, 13]\!]^2$.

Individual components comparison. Individual detectors are compared with the matchability of their detection and the description of the simple VGG-pool3 descriptor. This way, the *m.s.* only depends on the detection performance since the description is fixed for all detectors. The comparison between ELF and recent deep methods raises the question of whether triplet-like losses are relevant to train CNN descriptors. Indeed, these losses constrain the CNN features directly so that matching keypoints are near each other in descriptor space. Simpler loss, such as cross-entropy for classification, only the constrain the CNN output on the task while leaving the representation up to the CNN.

ELF-VGG detector is also integrated with existing descriptors. This evaluates how useful the CNN self-learned feature localisation compares with the hand-crafted and the learned ones.

Gradient Baseline. Visually, the feature gradient map is reminiscent of the image gradients computed with the Sobel or Laplacian operators. We run two variants of our pipeline where we replace the feature gradient with them. This aims at showing whether CNN feature gradients embed more information than image intensity gradients.

5 Results

Experiments show that ELF compares with the state-of-the-art on HPatches and demonstrates similar robustness properties with recent learned methods. It generates saliency maps visually akin to a Laplacian on very structured images (HPatches) but proves to be more robust on outdoor scenes with natural conditions (Webcam). When integrated with existing feature descriptors, ELF boosts the matching score. Even integrating ELF simple descriptor improves it with the exception of SuperPoint for which results are equivalent. This sheds new light on the representations learnt by CNNs and suggests that deep description methods may underexploit the information embedded in their trained networks. Another suggestion may be that the current metrics are not relevant anymore for

deep learning methods. Indeed, all can detect repeatable keypoints with more or less the same performances. Even though the matchability of the points (*m.s*) is a bit more discriminative, neither express how 'useful' the *kp* are for the end-goal task. One way to do so is to evaluate an end-goal task (*e.g.* Structure-from-Motion). However, for the evaluation to be rigorous all the other steps should be fixed for all papers. Recently, the Image Matching CVPR19 workshop proposed such an evaluation but is not fully automatic yet. These results also challenge whether current descriptor-training loss are a strong enough signal to constrain CNN features better than a simple cross-entropy.
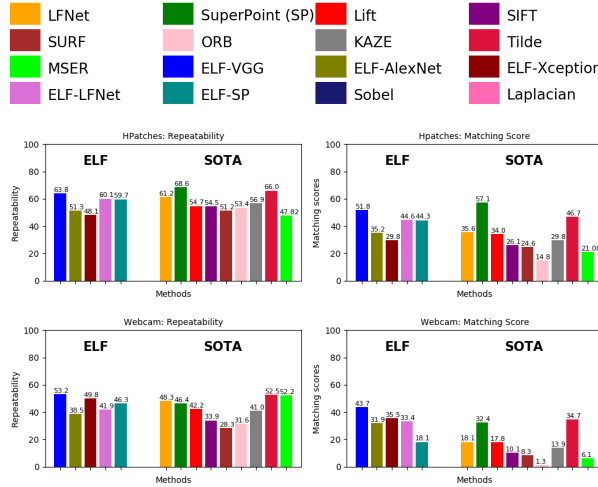

Figure 5: Top-Down: HPatches-Webcam. Left-Right: repeatability, matching score.

The tabular version of the following results is provided in Appendix. The graph results are better seen with color on a computer screen. Unless mentioned otherwise, we compute repeatability for each detector, and the matching score of detectors with their respective descriptors, when they have one. We use ELF-VGG-$pool_4$ descriptor for TILDE, MSER, ELF-VGG, ELF-SuperPoint, and ELF-LFNet. We use AlexNet and Xception feature maps to build their respective simple descriptors. The meta-parameters for each variants are provided in Appendix.

General performances. Figure 5 (top) shows that the *rep* variance is low across detectors whereas *ms* is more discriminative, hence the validation method (Section 4.1). On HPatches, SuperPoint (SP) reaches the best *rep-ms* [68.6, 57.1] closely followed by ELF (e.g. ELF-VGG: [63.8, 51.8]) and TILDE [66.0, 46.7]. In general, we observe that learning-based methods all outperform hand-crafted ones. Still, LF-Net and LIFT curiously underperform on HPatches: one reason may be that the data they are trained on differs too much from this one. LIFT is trained on outdoor images only and LF-Net on either indoor or outdoor datasets, whereas HPatches is made of a mix of them. We compute metrics for both LF-Net models and report the highest one (indoor). Even though LF-Net and LIFT fall behind the top learned methods, they still outperform hand-crafted ones which suggests that their framework learn feature specific information that hand-crafted methods can not capture. This supports the recent direction towards trained detectors and descriptors.

Light Robustness Again, *ms* is a better discriminant on Webcam than *rep* (Figure 5 bottom). ELF-VGG reaches top *rep-ms* [53.2, 43.7] closely followed by TILDE [52.5, 34.7] which was the state-of-the-art detector.

Overall, there is a performance degradation (∼20%) from HPatches to Webcam. HPatches holds images with standard features such as corners that state-of-the-art methods are made to recognise either by definition or by supervision. There are less such features in the Webcam dataset because of the natural lighting that blurs them. Also there are strong intensity variations that these models do not handle well. One reason may be that the learning-based methods never saw such lighting variations in their training set. But this assumption is rejected as we observe that even SuperPoint, which is trained on Coco images, outperforms LIFT and LF-Net, which are trained on outdoor images. Another justification can be that what matters the most is the pixel distribution the network is trained on, rather than the image content. The top methods are classifier-based ELF and SuperPoint: the first ones are trained on the huge Imagenet dataset and benefit from heavy data augmentation. SuperPoint also employs a considerable data strategy to train their network. Thus these networks may cover a much wider pixel distribution which would explain their robustness to pixel distribution changes such as light modifications.

Architecture influence ELF is tested on three classification networks as well as the descriptor networks of SuperPoint and LF-Net (Figure 5, bars under 'ELF').

For a fixed training task (classification) on a fixed dataset (ImageNet), VGG, AlexNet and Xception are compared. As could be expected, the network architecture has a critical impact on the detection and ELF-VGG outperforms the other variants. The *rep* gap can be explained by the fact that AlexNet is made of wider convolutions than VGG, which induces a higher loss of resolution when computing the gradient. As for *ms*, the higher representation space of VGG may help building more informative features which are a stronger signal to back-propagate. This could also justify why ELF-VGG outperforms ELF-Xception that has less parameters. Another explanation is that ELF-Xception's gradient maps seem smoother. Salient locations are then less emphasized which makes the keypoint detection harder. One could hint at the depth-wise convolution to explain this visual aspect but we could not find an experimental way to verify

it. Surprisingly, ELF-LFNet outperforms the original LF-Net on both HPatches and Webcam and ELF-SuperPoint variant reaches similar results as the original.
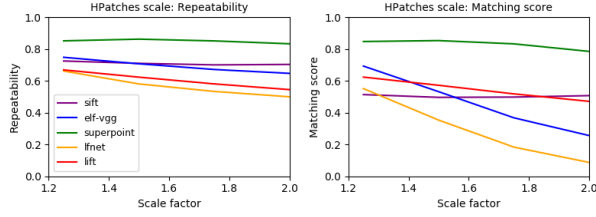

Figure 6: HPatches scale. Left-Right: rep, ms.

Scale Robustness. ELF-VGG is compared with state-of-the art detectors and their respective descriptors (Figure 6). Repeatability is mostly stable for all methods: SIFT and SuperPoint are the most invariant whereas ELF follows the same variations as LIFT and LF-Net. Once again, *ms* better assesses the detectors performance: SuperPoint is the most robust to scale changes, followed by LIFT and SIFT. ELF and LF-Net lose 50% of their matching score with the increasing scale. It is surprising to observe that LIFT is more scale-robust than LF-Net when the latter's global performance is higher. A reasonable explanation is that LIFT detects keypoints at 21 scales of the same image whereas LF-Net only runs its detector CNN on 5 scales. Nonetheless, ELF outperforms LF-Net without manual multi-scale processing.
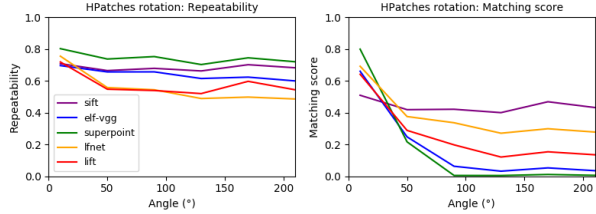

Figure 7: HPatches rotation. Left-Right: rep, ms.

Rotation Robustness. Even though *rep* shows little variations (Figure 7), all learned methods' *ms* crash while only SIFT survives the rotation changes. This can be explained by the explicit rotation estimation step of SIFT. However LIFT and LF-Net also run such a computation. This suggests that either SIFT's hand-crafted orientation estimation is more accurate or that HOG are more rotation invariant than learned features. LF-Net still performs better than LIFT: this may be because it learns the keypoint orientation on the keypoint features representation rather than the keypoint pixels as done in LIFT. Not surprisingly, ELF simple descriptor is not rotation invariant as the convolutions that make the CNN are not. This also explains why SuperPoint also crashes in a similar manner.

These results suggest that the orientation learning step in LIFT and LF-Net is needed but its robustness could be improved.
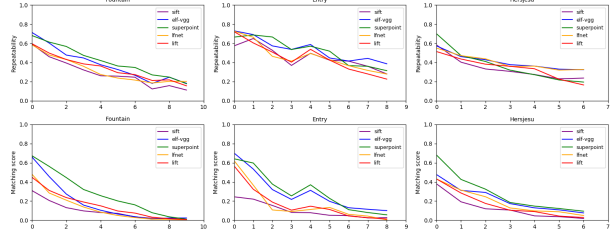

Figure 8: Robustness analysis: 3D viewpoint.

3D Viewpoint Robustness. While SIFT shows a clear advantage of pure-rotation robustness, it displays similar degradation as other methods on realistic rotation-and-translation on 3D structures. Figure 8 shows that all methods degrade uniformly. One could assume that this small data sample is not representative enough to run such robustness analysis. However, we think that these results rather suggest that all methods have the same robustness to 3D viewpoint changes. Even though previous analyses allows to rank the different feature matching pipelines, each has advantages over others on certain situations: ELF or SuperPoint on general homography matches, or SIFT on rotation robustness. This is why this paper only aims at showing ELF reaches the same performances and shares similar properties to existing methods as there is no generic ranking criteria. The recent evaluation run by the CVPR19 Image Matching Challenge [1] supports the previous conclusions.
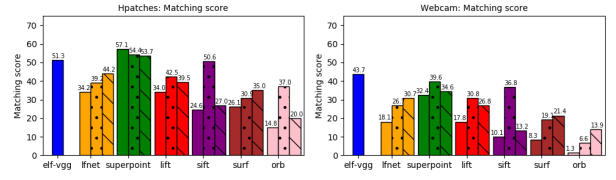

Figure 9: Left-Middle-Right bars: original method, integration of ELF detection, integration of ELF description.

Individual components performance. First, all methods' descriptor are replaced with the simple ELF-VGG-$pool_3$ one. We then compute their new *ms* and compare it to ELF-VGG on HPatches and Webcam (Figure 9, stripes). The description is based on $pool_3$ instead of $pool_4$ here for it produces better results for the other methods while preserving ours. ELF reaches higher *ms* [51.3] for all methods except for SuperPoint [53.7] for which it is comparable. This shows that ELF is as relevant, if not more, than previous hand-crafted or learned detectors. This naturally leads to the question: *'What kind of key-*

8

points does ELF detect ?' There is currently no answer to this question as it is complex to explicitly characterize properties of the pixel areas around keypoints. Hence the open question *'What makes a good keypoint ?'* mentioned at the beginning of the paper. Still, we observe that ELF activates mostly on high intensity gradient areas although not all of them. One explanation is that as the CNN is trained on the vision task, it learns to ignore image regions useless for the task. This results in killing the gradient signals in areas that may be unsuited for matching.

Another surprising observation regards CNN descriptors: SuperPoint (SP) keypoints are described with the SP descriptor in one hand and the simple ELF-VGG one in the other hand. Comparing the two resulting matching scores is one way to compare the SP and ELF descriptors. Results show that both approaches lead to similar *ms*. This result is surprising because SP specifically trains a description CNN so that its feature map is suitable for keypoint description [10]. In VGG training, there is no explicit constraints on the features from the cross-entropy loss. Still, both feature maps reach similar numerical description performance. This raises the question of whether contrastive-like losses, which input are CNN features, can better constrain the CNN representation than simpler losses, such as cross-entropy, which inputs are classification logits. This also shows that there is more to CNNs than only the task they are trained on: they embed information that can prove useful for unrelated tasks. Although the simple descriptor was defined for evaluation purposes, these results demonstrate that it can be used as a description baseline for feature extraction.

The integration of ELF detection with other methods' descriptor (Figure 9, circle) boosts the *ms*. [44] previously suggested that there may be a correlation between the detector and the descriptor within a same method, i.e. the LIFT descriptor is trained to describe only the keypoints output by its detector. However, these results show that ELF can easily be integrated into existing pipelines and even boost their performances.
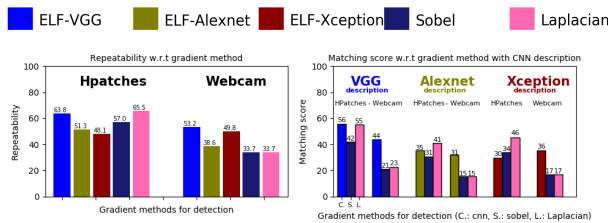

Figure 10: Gradient baseline.

Gradient Baseline The saliency map used in ELF is replaced with simple Sobel or Laplacian gradient maps. The rest of the detection pipeline stays the same and we compute their performance (Figure 10 Left). They are completed with simple ELF descriptors from the VGG, AlexNet and Xception networks. These new hybrids are then compared to their respective ELF variant (Right). Results show that these simpler gradients can detect systematic keypoints with comparable *rep* on very structured images such as HPatches. However, the ELF detector better overcomes light changes (Webcam). On HPatches, the Laplacian-variant reaches similar *ms* as ELF-VGG (55 *vs* 56) and outperforms ELF-AlexNet and ELF-Xception. These scores can be explained with the images structure: for heavy textured images, high intensity gradient locations are relevant enough keypoints. However, on Webcam, all ELF detectors outperform Laplacian and Sobel with a factor of 100%. This shows that ELF is more robust than Laplacian and Sobel operators. Also, feature gradient is a sparse signal which is better suited for local maxima detection than the much smoother Laplacian operator (Figure 11).
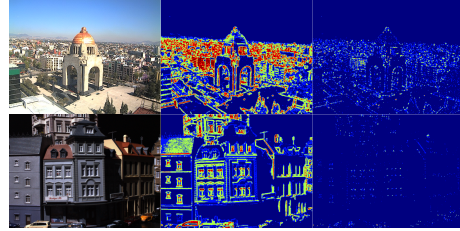

Figure 11: Feature gradient (right) provides a sparser signal than Laplacian (middle) which is more selective of salient areas.

Qualitative results Green lines show putative matches based only on nearest neighbour matching of descriptors. More qualitative results are available in the video [3].

Figure 12: Green lines show putative matches of the simple descriptor before RANSAC-based homography estimation.

CVPR19 Image Matching Challenge [1] This challenge evaluates detection/description methods on two standard tasks: 1) wide stereo matching and 2) structure from motion from small image sets. The *matching score* evaluates the first task, and the camera pose estimation is used for both tasks. Both applications are evaluated on the photo-tourism image collections of popular landmarks [16, 42]. More details on the metrics definition are available on the challenge website [1].

Wide stereo matching: Task 1 matches image pairs across wide baselines. It is evaluated with the keypoints

---

[3] https://youtu.be/oxbG5162yDs

9

ms and the relative camera pose estimation between two images. The evaluators run COLMAP to reconstruct dense 'ground-truth' depth which they use to translate keypoints from one image to another and compute the matching score. They use the RANSAC inliers to estimate the camera pose and measure performance with the "angular difference between the estimated and ground-truth vectors for both rotation and translation. To reduce this to one value, they use a variable threshold to determine each pose as correct or not, then compute the area under the curve up to the angular threshold. This value is thus the mean average precision up to x, or mAPx. They consider 5, 10, 15, 20, and 25 degrees" [1]. Submissions can contain up to 8000 keypoints and we submitted entries to the sparse category i.e. methods with up to 512 keypoints.
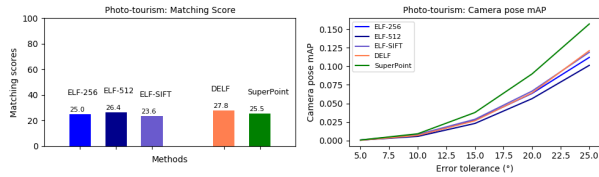

Figure 13: *Wide stereo matching.* Left: matching score (%) of sparse methods (up to 512 keypoints) on photo-tourism. Right: Evolution of mAP of camera pose for increasing tolerance threshold (degrees).

Figure 13 (left) shows the *ms* (%) of the submitted sparse methods. It compares ELF-VGG detection with DELF [27] and SuperPoint, where ELF is completed with either the simple descriptor from pool3 or pool4, and SIFT. The variant are dubbed respectively ELF-256, ELF-512 and ELF-SIFT. This allows us to sketch a simple comparison of descriptor performances between the simple descriptor and standard SIFT.

As previously observed on HPatches and Webcam, ELF and SuperPoint reach similar scores on Photo-Tourism. ELF-performance slightly increases from 25% to 26.4% when switching descriptors from VGG-pool3 to VGG-pool4. One explanation is that the feature space size is doubled from the first to the second. This would allow the pool4 descriptors to be more discriminative. However, the 1.4% gain may not be worth the additional memory use. Overall, the results show that ELF can compare with the SoA on this additional dataset that exhibits more illumination and viewpoint changes than HPatches and Webcam.

This observation is reinforced by the camera pose evaluation (Figure 13 right). SuperPoint shows as slight advantage over others that increases from 1% to 5% across the error tolerance threshold whereas ELF-256 exhibits a minor under-performance. Still, these results show ELF compares with SoA performance even though it is not trained explicitly for detection/description.
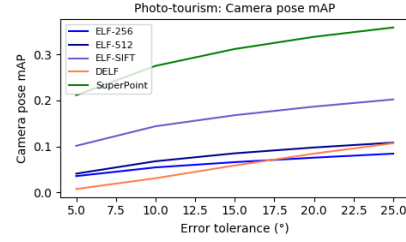

Figure 14: *SfM from small subsets.* Evolution of mAP of camera pose for increasing tolerance threshold.

Structure-from-Motion from small subsets. Task 2 "proposes to to build SfM reconstructions from small (3, 5, 10, 25) subsets of images and use the poses obtained from the entire (much larger) set as ground truth" [1].

Figure 14 shows that SuperPoint reaches performance twice as big as the next best method ELF-SIFT. This suggests that when few images are available, SuperPoint performs better than other approaches. One explanation is that even in 'sparse-mode', *i.e.* when the number of keypoints is restricted up to 512, SuperPoint samples points more densely than the others ($\sim$383 *v.s.* $\sim$210 for the others). Thus, SuperPoint provides more keypoints to triangulate i.e. more 2D-3D correspondences to use when estimating the camera pose. This suggests that high keypoint density is a crucial characteristic of the detection method for Structure-from-Motion. In this regard, ELF still has room for improvement compared to SuperPoint.

6 Conclusion

We have introduced ELF, a novel method to extract feature locations from pre-trained CNNs, with no further training. Extensive experiments show that it performs as well as state-of-the art detectors. It can easily be integrated into existing matching pipelines and proves to boost their matching performances. Even when completed with a simple feature-map-based descriptor, it turns into a competitive feature matching pipeline. These results shed new light on the information embedded inside trained CNNs. This work also raises questions on the descriptor training of deep-learning approaches: whether their losses actually constrain the CNN to learn better features than the ones it would learn on its own to complete a vision task. Preliminary results show that the CNN architecture, the training task and the dataset have substantial impact on the detector performances. A further analysis of these correlations is the object of a future work.

References

[1] Cvpr19 image matching challenge. `https://image-matching-workshop.github.io/challenge/`, 2019.

[2] AANÆS, H., DAHL, A. L., AND PEDERSEN, K. S. Interesting interest points. *International Journal of Computer Vision 97*, 1 (2012), 18–35.

[3] ABADI, M., BARHAM, P., CHEN, J., CHEN, Z., DAVIS, A., DEAN, J., DEVIN, M., GHEMAWAT, S., IRVING, G., ISARD, M., ET AL. Tensorflow: a system for large-scale machine learning. In *OSDI* (2016), vol. 16, pp. 265–283.

[4] ALCANTARILLA, P. F., BARTOLI, A., AND DAVISON, A. J. Kaze features. In *European Conference on Computer Vision* (2012), Springer, pp. 214–227.

[5] BALNTAS, V., LENC, K., VEDALDI, A., AND MIKOLAJCZYK, K. Hpatches: A benchmark and evaluation of handcrafted and learned local descriptors. In *Conference on Computer Vision and Pattern Recognition (CVPR)* (2017), vol. 4, p. 6.

[6] BALNTAS, V., RIBA, E., PONSA, D., AND MIKOLAJCZYK, K. Learning local feature descriptors with triplets and shallowconvolutional neural networks. In *BMVC* (2016), vol. 1, p. 3.

[7] BAY, H., TUYTELAARS, T., AND VAN GOOL, L. Surf: Speeded up robust features. In *European conference on computer vision* (2006), Springer, pp. 404–417.

[8] CALONDER, M., LEPETIT, V., STRECHA, C., AND FUA, P. Brief: Binary robust independent elementary features. In *European conference on computer vision* (2010), Springer, pp. 778–792.

[9] CHOLLET, F. Xception: Deep learning with depthwise separable convolutions. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017* (2017), pp. 1800–1807.

[10] CHOY, C. B., GWAK, J., SAVARESE, S., AND CHANDRAKER, M. Universal correspondence network. In *Advances in Neural Information Processing Systems* (2016), pp. 2414–2422.

[11] DENG, J., DONG, W., SOCHER, R., LI, L.-J., LI, K., AND FEI-FEI, L. Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on* (2009), Ieee, pp. 248–255.

[12] DETONE, D., MALISIEWICZ, T., AND RABINOVICH, A. Superpoint: Self-supervised interest point detection and description. In *CVPR Deep Learning for Visual SLAM Workshop* (2018).

[13] FISCHER, P., DOSOVITSKIY, A., AND BROX, T. Descriptor matching with convolutional neural networks: a comparison to sift. *arXiv preprint arXiv:1405.5769* (2014).

[14] GATYS, L. A., ECKER, A. S., AND BETHGE, M. Image style transfer using convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2016), pp. 2414–2423.

[15] HAN, X., LEUNG, T., JIA, Y., SUKTHANKAR, R., AND BERG, A. C. Matchnet: Unifying feature and metric learning for patch-based matching. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2015), pp. 3279–3286.

[16] HEINLY, J., SCHONBERGER, J. L., DUNN, E., AND FRAHM, J.-M. Reconstructing the world* in six days*(as captured by the yahoo 100 million image dataset). In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2015), pp. 3287–3295.

[17] KAPUR, J. N., SAHOO, P. K., AND WONG, A. K. A new method for gray-level picture thresholding using the entropy of the histogram. *Computer vision, graphics, and image processing 29*, 3 (1985), 273–285.

[18] KRIZHEVSKY, A., SUTSKEVER, I., AND HINTON, G. E. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems* (2012), pp. 1097–1105.

[19] LENC, K., GULSHAN, V., AND VEDALDI, A. Vlbenchmkars. `http://www.vlfeat.org/benchmarks/xsxs`, 2011.

[20] LIN, T.-Y., MAIRE, M., BELONGIE, S., HAYS, J., PERONA, P., RAMANAN, D., DOLLÁR, P., AND ZITNICK, C. L. Microsoft coco: Common objects in context. In *European conference on computer vision* (2014), Springer, pp. 740–755.

[21] LOWE, D. G. Distinctive image features from scale-invariant keypoints. *International journal of computer vision 60*, 2 (2004), 91–110.

[22] MAHENDRAN, A., AND VEDALDI, A. Understanding deep image representations by inverting them. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (2015), pp. 5188–5196.

[23] MATAS, J., CHUM, O., URBAN, M., AND PAJDLA, T. Robust wide-baseline stereo from maximally stable extremal regions. *Image and vision computing 22*, 10 (2004), 761–767.

[24] MELEKHOV, I., KANNALA, J., AND RAHTU, E. Siamese network features for image matching. In *2016 23rd International Conference on Pattern Recognition (ICPR)* (2016), IEEE, pp. 378–383.

[25] MIKOLAJCZYK, K., AND SCHMID, C. A performance evaluation of local descriptors. *IEEE transactions on pattern analysis and machine intelligence 27*, 10 (2005), 1615–1630.

[26] MIKOLAJCZYK, K., TUYTELAARS, T., SCHMID, C., ZISSERMAN, A., MATAS, J., SCHAFFALITZKY, F., KADIR, T., AND VAN GOOL, L. A comparison of affine region detectors. *International journal of computer vision 65*, 1-2 (2005), 43–72.

[27] NOH, H., ARAUJO, A., SIM, J., WEYAND, T., AND HAN, B. Largescale image retrieval with attentive deep local features. In *Proceedings of the IEEE International Conference on Computer Vision* (2017), pp. 3456–3465.

[28] ONO, Y., TRULLS, E., FUA, P., AND K.M.YI. Lf-net: Learning local features from images. In *Advances in Neural Information Processing Systems* (2018).

[29] ROSTEN, E., AND DRUMMOND, T. Machine learning for high-speed corner detection. In *European conference on computer vision* (2006), Springer, pp. 430–443.

[30] RUBLEE, E., RABAUD, V., KONOLIGE, K., AND BRADSKI, G. Orb: An efficient alternative to sift or surf. In *Computer Vision (ICCV), 2011 IEEE international conference on* (2011), IEEE, pp. 2564–2571.

[31] SAVINOV, N., SEKI, A., LADICKY, L., SATTLER, T., AND POLLEFEYS, M. Quad-networks: unsupervised learning to rank for interest point detection. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2017).

[32] SCHONBERGER, J. L., AND FRAHM, J.-M. Structure-from-motion revisited. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2016), pp. 4104–4113.

[33] SELVARAJU, R. R., COGSWELL, M., DAS, A., VEDANTAM, R., PARIKH, D., BATRA, D., ET AL. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *ICCV* (2017), pp. 618–626.

[34] SIMO-SERRA, E., TRULLS, E., FERRAZ, L., KOKKINOS, I., FUA, P., AND MORENO-NOGUER, F. Discriminative learning of deep convolutional feature point descriptors. In *Proceedings of the IEEE International Conference on Computer Vision* (2015), pp. 118–126.

[35] SIMONYAN, K., VEDALDI, A., AND ZISSERMAN, A. Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv preprint arXiv:1312.6034* (2013).

[36] SIMONYAN, K., AND ZISSERMAN, A. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556* (2014).

[37] SMILKOV, D., THORAT, N., KIM, B., VIÉGAS, F., AND WATTENBERG, M. Smoothgrad: removing noise by adding noise. *arXiv preprint arXiv:1706.03825* (2017).

[38] SPRINGENBERG, J., DOSOVITSKIY, A., BROX, T., AND RIEDMILLER, M. Striving for simplicity: The all convolutional net. In *ICLR (workshop track)* (2015).

[39] STRECHA, C., VON HANSEN, W., VAN GOOL, L., FUA, P., AND THOENNESSEN, U. On benchmarking camera calibration and multi-view stereo for high resolution imagery. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on* (2008), Ieee, pp. 1–8.

[40] SUNDARARAJAN, M., TALY, A., AND YAN, Q. Axiomatic attribution for deep networks. In *International Conference on Machine Learning* (2017), pp. 3319–3328.

[41] TAIRA, H., OKUTOMI, M., SATTLER, T., CIMPOI, M., POLLEFEYS, M., SIVIC, J., PAJDLA, T., AND TORII, A. Inloc: Indoor visual localization with dense matching and view synthesis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2018), pp. 7199–7209.

[42] THOMEE, B., SHAMMA, D. A., FRIEDLAND, G., ELIZALDE, B., NI, K., POLAND, D., BORTH, D., AND LI, L.-J. Yfcc100m: The new data in multimedia research. *Communications of the ACM 59*, 2, 64–73.

[43] VERDIE, Y., YI, K., FUA, P., AND LEPETIT, V. Tilde: A temporally invariant learned detector. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2015), pp. 5279–5288.

[44] YI, K. M., TRULLS, E., LEPETIT, V., AND FUA, P. Lift: Learned invariant feature transform. In *European Conference on Computer Vision* (2016), Springer, pp. 467–483.

[45] ZAGORUYKO, S., AND KOMODAKIS, N. Learning to compare image patches via convolutional neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (2015), pp. 4353–4361.

[46] ZEILER, M. D., AND FERGUS, R. Visualizing and understanding convolutional networks. In *European conference on computer vision* (2014), Springer, pp. 818–833.

A Metrics definition

We explicit the repeatability and matching score definitions introduced in [26] and our adaptations using the following notations: let $(\mathbf{I}^1, \mathbf{I}^2)$, be a pair of images and $\mathcal{KP}^i = (kp_j^i)_{j<N_i}$ the set of N_i keypoints in image $\mathbf{I_i}$. Both metrics are in the range $[0,1]$ but we express them as percentages for better expressibility.

Repeatability Repeatability measures the percentage of keypoints common to both images. We first warp $\mathcal{KP}^1$ to $\mathbf{I}^2$ and note $\mathcal{KP}^{1,w}$ the result. A naive definition of repeatability is to count the number of pairs $(kp^{1,w}, kp^2) \in \mathcal{KP}^{1,w} \times \mathcal{KP}^2$ such that $\|kp^{1,w} - kp^2\|_2 < \epsilon$, with ϵ a distance threshold. As pointed by [43], this definition overestimates the detection performance for two reasons: a keypoint close to several projections can be counted several times. Moreover, with a large enough number of keypoints, even simple random sampling can achieve high repeatability as the density of the keypoints becomes high.

We instead use the definition implemented in VL-Bench [19]: we define a weighted graph (V, E) where the edges are all the possible keypoint pairs between $\mathcal{KP}^{1,w}$ and $\mathcal{KP}^2$ and the weights are the euclidean distance between keypoints.

$$V = (kp^{1,w} \in \mathcal{KP}^{1,w}) \cup (kp^2 \in \mathcal{KP}^2)$$
$$E = (kp^{1,w}, kp^2, \|kp^{1,w} - kp^2\|_2) \in \mathcal{KP}^{1,w} \times \mathcal{KP}^2 \tag{1}$$

We run a greedy bipartite matching on the graph and count the matches with a distance less than ϵ_{kp}. With $\mathcal{M}$ be the resulting set of matches:

$$repeatability = \frac{\mathcal{M}}{\min(|\mathcal{KP}^1|, |\mathcal{KP}^2|)} \tag{2}$$

We set the distance threshold $\epsilon = 5$ as is done in LIFT [44] and LF-Net [28].

Matching score The matching score definition introduced in [26] captures the percentage of keypoint pairs that are nearest neighbours both in image space and in descriptor space, and for which these two distances are below their respective threshold ϵ_{kp} and ϵ_d. Let $\mathcal{M}$ be defined as in the previous paragraph and $\mathcal{M}_d$ be the analog of $\mathcal{M}$ when the graph weights are descriptor distances instead of keypoint euclidean distances. We delete all the pairs with a distance above the thresholds ϵ and ϵ_d in $\mathcal{M}$ and $\mathcal{M}_d$ respectively. We then count the number of pairs which are both nearest neigbours in image space and descriptor space i.e. the intersection of $\mathcal{M}$ and $\mathcal{M}_d$:

$$matching\ score = \frac{\mathcal{M} \cap \mathcal{M}_d}{\min(|\mathcal{KP}^1|, |\mathcal{KP}^2|)} \tag{3}$$

One drawback of this definition is that there is no unique descriptor distance threshold ϵ_d valid for all methods. For example, the SIFT descriptor as computed by OpenCV is a $[0, 255]^{128}$ vector for better computational precision, the SuperPoint descriptor is a $[0, 1]^{256}$ vector and the ORB descriptor is a 32 bytes binary vector. Not only the vectors are not defined over the same normed space but their range vary significantly. To avoid introducing human bias by setting a descriptor distance threshold ϵ_d for each method, we choose to set $\epsilon_d = \infty$ and compute the matching score as in [26]. This means that we consider any descriptor match valid as long as they match corresponding keypoints even when the descriptor distance is high.

B Tabular results

	Repeatability		Matching Score	
	[5]	[43]	[5]	[43]
ELF-VGG	63.81	**53.23**	**51.84**	**43.73**
ELF-AlexNet	51.30	38.54	35.21	31.92
ELF-Xception	48.06	**49.84**	29.81	**35.48**
ELF-SuperPoint	59.7	46.29	44.32	18.11
ELF-LFNet	60.1	41.90	44.56	33.43
LF-Net	61.16	48.27	34.19	18.10
SuperPoint	**68.57**	46.35	**57.11**	32.44
LIFT	54.66	42.21	34.02	17.83
SURF	54.51	33.93	26.10	10.13
SIFT	51.19	28.25	24.58	8.30
ORB	53.44	31.56	14.76	1.28
KAZE	56.88	41.04	29.81	13.88
TILDE	**65.96**	52.53	46.71	34.67
MSER	47.82	52.23	21.08	6.14

Table 1: Generic performances on HPatches [5]. Robustness to light (Webcam [43]). (Fig. 5).

	LF-Net	SuperPoint	LIFT	SIFT	SURF	ORB
[5]	34.19	**57.11**	34.02	24.58	26.10	14.76
	44.19	53.71	**39.48**	**27.03**	**34.97**	**20.04**
[43]	18.10	32.44	17.83	10.13	8.30	1.28
	30.71	**34.60**	**26.84**	**13.21**	**21.43**	**13.91**

Table 2: Individual component performance (Fig. 9-stripes). Matching score for the integration of the VGG $pool_3$ simple-descriptor with other's detection. Top: Original description. Bottom: Integration of simple-descriptor. HPatches: [5]. Webcam: [43]

	LF-Net	SuperPoint	LIFT	SIFT	SURF	ORB
[5]	34.19	**57.11**	34.02	24.58	26.10	14.76
	39.16	54.44	**42.48**	**50.63**	**30.91**	**36.96**
[43]	18.10	32.44	17.83	10.13	8.30	1.28
	26.70	**39.55**	**30.82**	**36.83**	**19.14**	**6.60**

Table 3: Individual component performance (Fig. 9-circle). Matching score for the integration of ELF-VGG (on $pool_2$) with other's descriptor. Top: Original detection. Bottom: Integration of ELF. HPatches: [5]. Webcam: [43]

	Repeatability		Matching Score	
	[5]	[43]	[5]	[43]
Sobel-VGG	56.99	33.74	42.11	20.99
Lapl.-VGG	**65.45**	33.74	**55.25**	22.79
VGG	63.81	**53.23**	51.84	**43.73**
Sobel-AlexNet	56.44	33.74	30.57	15.42
Lapl.-AlexNet	**65.93**	33.74	**40.92**	15.42
AlexNet	51.30	**38.54**	35.21	**31.92**
Sobel-Xception	56.44	33.74	34.14	16.86
Lapl.-Xception	**65.93**	33.74	**42.52**	16.86
Xception	48.06	**49.84**	29.81	**35.48**

Table 4: Gradient baseline on HPatches [5] and Webcam [43] (Fig. 10).

C ELF Meta Parameters

This section specifies the meta parameters values for the ELF variants. For all methods, $(w_{NMS}, b_{NMS}) = (10, 10)$.

- Denoise: $(\mu_{noise}, \sigma_{noise})$.

- Threshold: $(\mu_{thr}, \sigma_{thr})$.

- F^l: the feature map which gradient is used for detection.

- simple-des: the feature map used for simple-description. Unless mentioned otherwise, the feature map is taken from the same network as the detection feature map F^l.

Nets	Denoise	Threshold	F^l	simple-desc
VGG	(5,5)	(5,4)	pool2	pool4
Alexnet	(5,5)	(5,4)	pool1	pool2
Xception	(9,3)	(5,4)	block2-conv1	block4-pool
SuperPoint	(7,2)	(17,6)	conv1a	VGG-pool3
LF-Net	(5,5)	(5,4)	block2-BN	VGG-pool3

Table 5: Generic performances on HPatches (Fig. 5). (BN: Batch Norm)

Nets	Denoise	Threshold	F^l	simple-desc
VGG	(5,5)	(5,4)	pool2	pool4
Alexnet	(5,5)	(5,4)	pool1	pool2
Xception	(9,9)	(5,4)	block2-conv1	block4-pool
SuperPoint	(7,2)	(17,6)	conv1a	VGG-pool3
LF-Net	(5,5)	(5,4)	block2-conv	VGG-pool3

Table 6: Robustness to light on Webcam (Fig. 5).

Nets	Denoise	Threshold	F^l	simple-desc
VGG	(5,2)	(17,6)	pool2	pool4

Table 7: Robustness to scale on HPatches (Fig. 6).

Nets	Denoise	Threshold	F^l	simple-desc
VGG	(5,2)	(17,6)	pool2	pool4

Table 8: Robustness to rotation on HPatches (Fig. 7).

Nets	Denoise	Threshold	F^l	simple-desc
VGG	(5,2)	(17,6)	pool2	pool4

Table 9: Robustness to 3D viewpoint on Strecha (Fig. 8).

Nets	Denoise	Threshold	F^l	simple-desc
VGG	(5,5)	(5,5)	pool2	pool3

Table 10: Individual component analysis (Fig. 9)

Nets	Denoise	Threshold	F^l	simple-desc
VGG	(5,5)	(5,4)	pool2	pool4
Sobel	(9,9)	(5,4)	-	pool4
Laplacian	(9,9)	(5,4)	-	pool4

Table 11: Gradient baseline on HPatches and Webcam (Fig. 10).

Figure 15: Enlargement of Figure 3. Saliency maps computed from the feature map gradient $\left| {}^T F^l(x) \cdot \frac{\partial F^l}{\partial \mathbf{I}} \right|$. Enhanced image contrast for better visualisation. Top row: gradients of VGG $pool_2$ and $pool_3$ show a loss of resolution from $pool_2$ to $pool_3$. Bottom: $(pool_i)_{i \in [1,2,5]}$ of VGG on Webcam, HPatches and Coco images. Low level saliency maps activate accurately whereas higher saliency maps are blurred.