# Discriminative Learning of Deep Convolutional Feature Point Descriptors

Edgar Simo-Serra[*,1,5], Eduard Trulls[*,2,5], Luis Ferraz[3], Iasonas Kokkinos[4], Pascal Fua[2], Francesc Moreno-Noguer[5]

[1] Waseda University, Tokyo, Japan [2] CVLab, École Polytechnique Fédérale de Lausanne, Switzerland [3] Catchoom Technologies, Barcelona, Spain
[4] CentraleSupelec and INRIA-Saclay, Chatenay-Malabry, France [5] Institut de Robòtica i Informàtica Industrial (CSIC-UPC), Barcelona, Spain (*: First two authors contributed equally)

## Objective

- Learn compact, discriminative representations of image patches with Convolutional Neural Networks.
- Optimize for comparisons with the $L_2$ distance, i.e. no metric learning. Our descriptors work within existing pipelines.

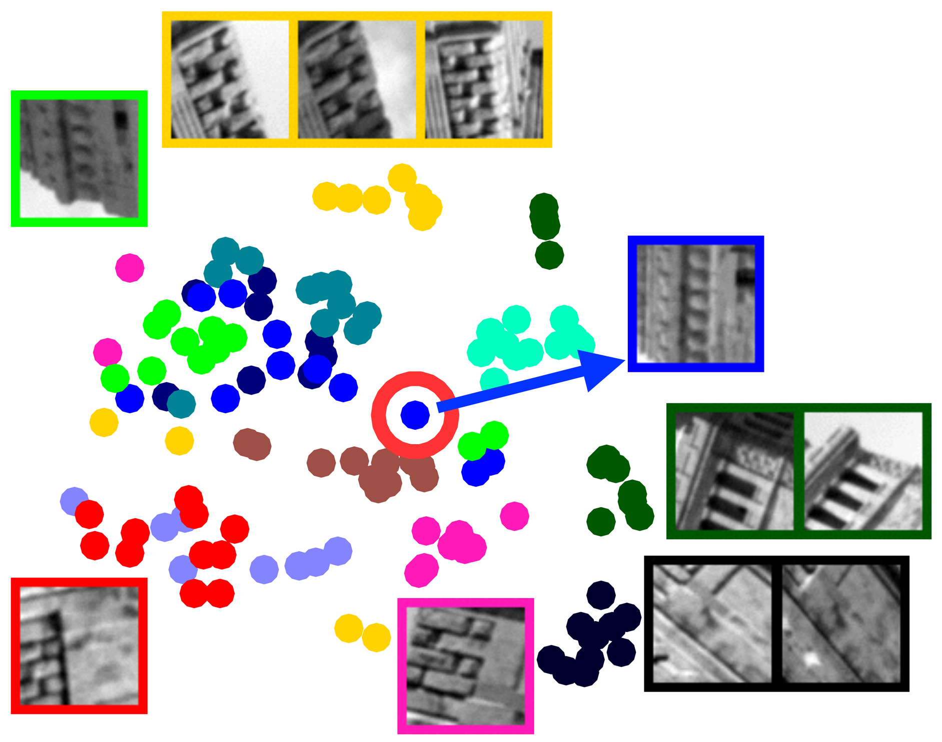
## Main features

- **Drop-in replacement for SIFT:** 128f, compare with the $L_2$ norm.
- **Consistent improvements** over the state of the art.
- Trained in one dataset, but **generalizes very well** to scaling, rotation, deformation and illumination changes.
- Computational efficiency (on GPU: 0.76 ms; dense SIFT: 0.14 ms).

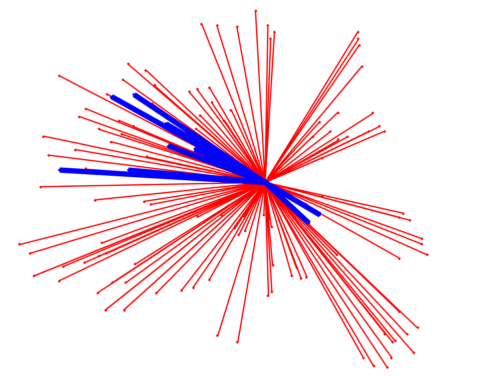
Code is available: https://github.com/etrulls/deepdesc-release

## Key observation

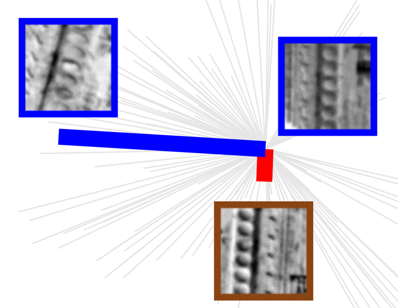
1. We train a Siamese architecture with **pairs of patches.** We want to **bring matching pairs together and otherwise pull them apart.**
2. Problem? Randomly sampled pairs are already **easy to separate**.
3. Solution: To train discriminative networks we use **hard negative and positive mining**. This proves essential for performance.



(a) 12 points/132 patches with t-SNE [8]

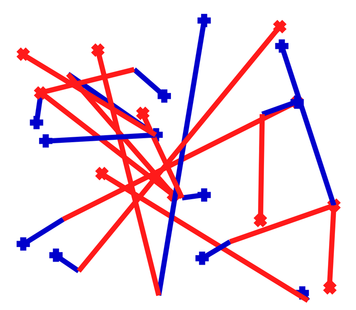


(b) All pairs: pos/neg

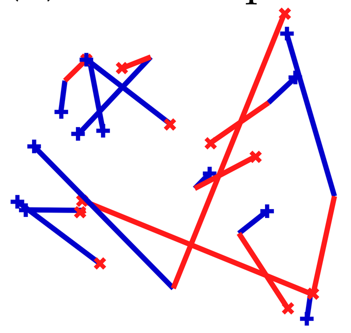


(c) "Hard" pairs: pos/neg

We take samples from [1], for illustration. Corresponding patches are shown with same color:
(a) Representation from t-SNE [8]. **Distance encodes similarity.**
(b) Random sampling: similar (close) positives and different (distant) negatives.
(c) We mine the samples to obtain dissimilar positives (+, long blue segments) and similar negatives (×, short red segments):
(d) Random sampling results in easy pairs.
(e) Mined pairs with harder correspondences.



(d) Random pairs



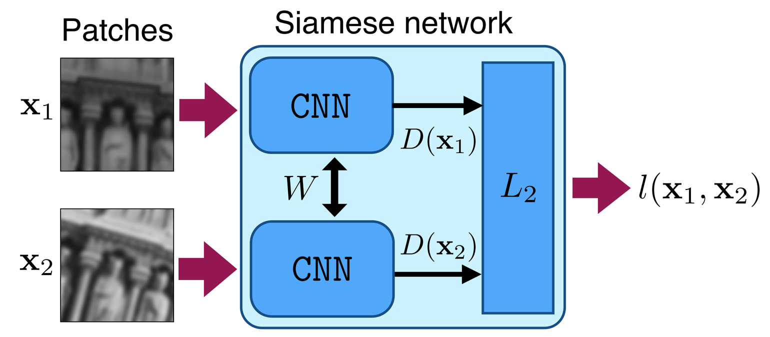
(e) Mined pairs

This allows us to train discriminative models with a small number of parameters (~45k), which also alleviates overfitting concerns.

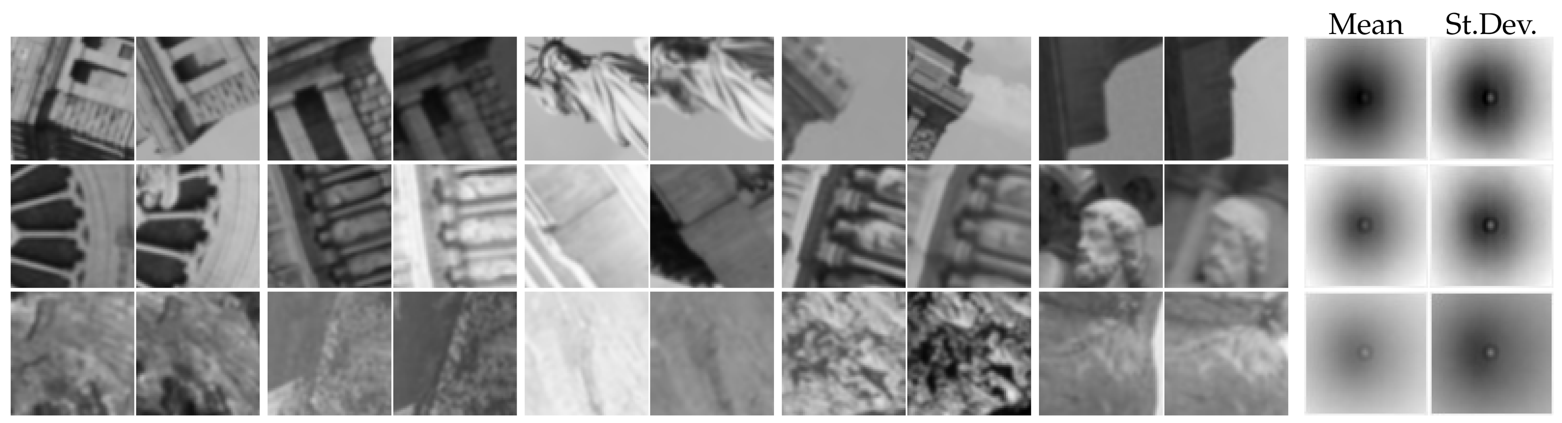
## Model & Training

Our model is a **3-Layer Convolutional Neural Network**. For training we use a **siamese architecture** with weight sharing and SGD.

| Layer | 1 | 2 | 3 |
|---|---|---|---|
| Input size | 64 × 64 | 29 × 29 | 8 × 8 |
| Filter size | 7 × 7 | 6 × 6 | 5 × 5 |
| Output channels | 32 | 64 | 128 |
| Pooling & Norm.tion | 2 × 2 | 3 × 3 | 4 × 4 |
| Nonlinearity | Tanh | Tanh | Tanh |
| Stride | 2 | 3 | 4 |




Train on the **MVS Dataset [1]**. 64 × 64 grayscale patches from SFM: Statue of Liberty (LY, top), NotreDame (ND, center), Yosemite (YO, bottom). ~150k points and ~450k patches each ⇒ $10^6$ **positive pairs and $10^{12}$ negative pairs** ⇒ Efficient exploration with **mining.**




We minimize the hinge embedding loss. With 3D point indices $p_1, p_2$:

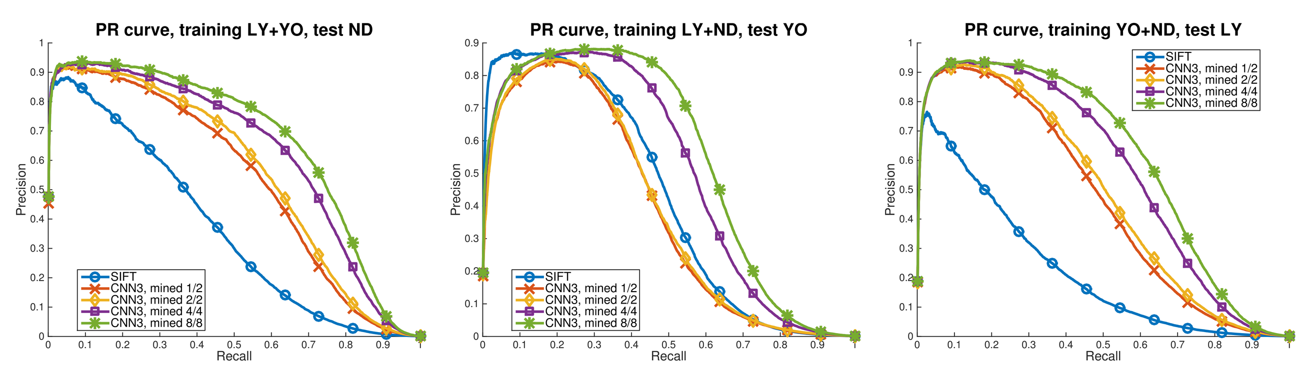
$$l(\mathbf{x}_1,\mathbf{x}_2)=\begin{cases}\|D(\mathbf{x}_1)-D(\mathbf{x}_2)\|_2, & p_1=p_2\\ \max(0, C-\|D(\mathbf{x}_1)-D(\mathbf{x}_2)\|_2), & p_1\neq p_2\end{cases}$$

This penalizes corresponding pairs that are placed far apart, and non-corresponding pairs that are less than $C$ units apart.

**Methodology:** Train over two sets and test over third (*leave-one-out*), with cross-validation. Metric: **precision-recall** (PR). 'Needle in a haystack' setting: pick **10k unique points** and generate **one positive pair and 1k negative pairs for each**, i.e. 10k positives vs. 10M negatives. Results summarized by **'Area Under the Curve'** (AUC).

## Effect of mining

(a) Forward-propagate positives $s_p \geq 128$ and negatives $s_n \geq 128$.
(b) Pick the 128 with the largest loss (for each) and back-propagate.




| $s_p$ | $s_n$ | PR AUC |
|---|---|---|
| 128 | 128 | 0.366 |
| 256 | 256 | 0.374 |
| 512 | 512 | 0.369 |
| 1024 | 1024 | 0.325 |

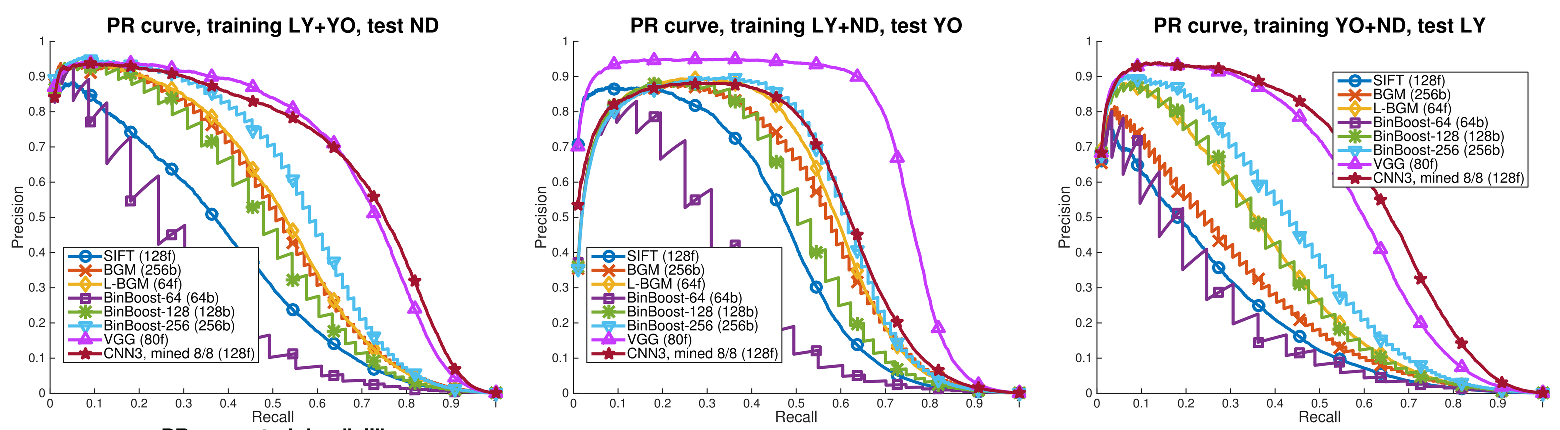
**Table 1:** (a) No mining. Larger batches **do not help.**

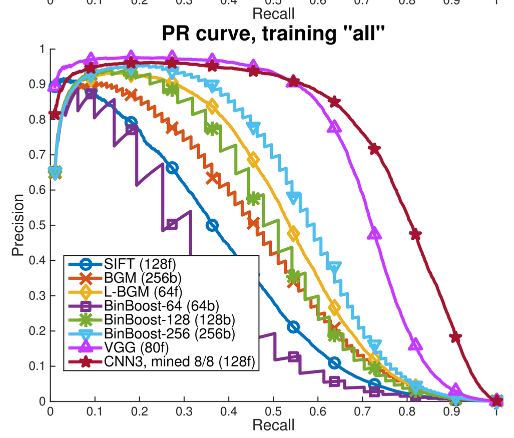
| $s_p$ | $s_n$ | $r_p$ | $r_n$ | Cost | PR AUC |
|---|---|---|---|---|---|
| 128 | 256 | 1 | 2 | 20% | 0.558 |
| 256 | 256 | 2 | 2 | 35% | 0.596 |
| 512 | 512 | 4 | 4 | 48% | 0.703 |
| 1024 | 1024 | 8 | 8 | 67% | **0.746** |

**Table 2:** (b) Mining with $r_p = s_p/128$, $r_n = s_n/128$. The mining cost is incurred **during training only.**

## Comparison with the state-of-the-art on MVS

We benchmark our models against SIFT, BinBoost [7], and VGG [4]. Better performance on 2/3 splits. Why? YO is very different from LY/ND (e.g. mean/std). Training on all three sets: top performance.

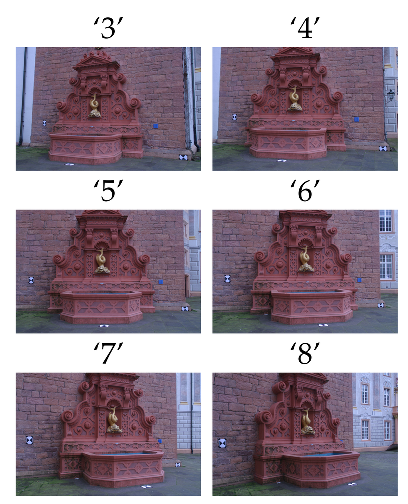



| Test | SIFT (128f) | BGM (256b) | L-BGM (64f) | BinBoost-{64,128,256} (64b) | (128b) | (256b) | VGG (80f) | Ours (128f) |
|---|---|---|---|---|---|---|---|---|
| ND | 0.349 | 0.487 | 0.495 | 0.267 | 0.451 | 0.549 | 0.663 | **0.667** |
| YO | 0.425 | 0.495 | 0.517 | 0.283 | 0.457 | 0.533 | **0.709** | 0.545 |
| LY | 0.226 | 0.268 | 0.355 | 0.202 | 0.346 | 0.410 | 0.558 | **0.608** |
| All | 0.370 | 0.440 | 0.508 | 0.291 | 0.469 | 0.550 | 0.693 | **0.756** |

## Generalization: Wide-Baseline Matching

Data from [5]. We match a set of points from view '3' against '4' to '8' (increasing baseline) and build PR curves, as before. No re-training.




| Descriptor | Training | '3' vs '4' | '3' vs '5' | '3' vs '6' | '3' vs '7' | '3' vs '8' |
|---|---|---|---|---|---|---|
| Ours | LY+YO | **0.923** | 0.690 | **0.456** | 0.218 | **0.088** |
| Ours | LY+ND | 0.919 | 0.677 | 0.424 | 0.197 | 0.058 |
| Ours | YO+ND | 0.922 | **0.685** | 0.439 | **0.228** | 0.058 |
| VGG [4] | YO | 0.894 | 0.632 | 0.400 | 0.174 | 0.067 |
| VGG [4] | ND | 0.880 | 0.590 | 0.372 | 0.182 | 0.058 |
| VGG [4] | LY | 0.879 | 0.582 | 0.365 | 0.166 | 0.064 |
| Daisy [6] | – | 0.835 | 0.594 | 0.363 | 0.172 | 0.032 |
| SIFT [2] | – | 0.772 | 0.532 | 0.308 | 0.138 | 0.053 |

## Generalization: Deformation and Illumination

Our models outperform the state-of-the-art on illumination changes and non-rigid deformations [3] without re-training or fine-tuning.


| Descriptor | Training | Def. | Ill. | Def.+Ill. |
|---|---|---|---|---|
| Ours | LY+YO | 76.568 | 88.434 | 75.933 |
| Ours | LY+ND | 75.702 | 87.521 | 75.606 |
| Ours | YO+ND | **76.731** | 88.898 | **76.591** |
| VGG [4] | YO | 74.120 | 87.342 | 74.765 |
| VGG [4] | ND | 72.629 | 84.690 | 72.599 |
| VGG [4] | LY | 72.602 | 84.848 | 72.565 |
| DaLI [3] | - | 70.577 | **89.895** | 72.912 |
| Daisy [6] | - | 67.373 | 75.402 | 66.197 |
| SIFT [2] | - | 55.822 | 60.760 | 53.431 |

## References

[1] M. Brown, Gang Hua, and S. Winder. Discriminative learning of local image descriptors. *PAMI*, 2011.
[2] D. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 2004.
[3] E. Simo-Serra, C. Torras, and F. Moreno-Noguer. DaLI: Deformation and Light Invariant Descriptor. *IJCV*, 2015.
[4] K. Simonyan, A. Vedaldi, and A. Zisserman. Learning local feature descriptors using convex optimisation. *PAMI*, 2014.
[5] C. Strecha, W. von Hansen, L. V. Gool, P. Fua, and U. Thoennessen. On benchmarking camera calibration and multi-view stereo for high resolution imagery. In *CVPR*, 2008.
[6] E. Tola, V. Lepetit, and P. Fua. DAISY: An efficient dense descriptor applied to wide baseline stereo. *PAMI*, 2010.
[7] T. Trzcinski, M. Christoudias, P. Fua, and V. Lepetit. Boosting binary keypoint descriptors. In *CVPR*, 2013.
[8] L. van der Maaten and G. Hinton. Visualizing data using t-SNE. In *JMLR*, 2008.