Learning Local Image Descriptors with Autoencoders

Nina Žižakić[1,*], Izumi Ito[2], and Aleksandra Pižurica[1]

[1] Department of Telecommunications and Information Processing, TELIN – GAIM, Ghent University – imec
nina.zizakic@ugent.be, aleksandra.pizurica@ugent.be

[2] Department of Human System Science, Tokyo Institute of Technology
ito@ict.e.titech.ac.jp

Summary. In this paper, we propose an efficient method for learning local image descriptors with convolutional autoencoders. We design an autoencoder architecture that yields computationally efficient extraction of patch descriptors through an intermediate image representation. The proposed approach yields significant savings in memory and processing time compared to a reference autoencoder-based patch descriptor. The results demonstrate improved robustness to noise and missing data.

Keywords. Local image descriptors, autoencoders, unsupervised deep learning.

1 Introduction

Local feature descriptors are fundamental to image processing tasks such as image denoising, inpainting, object tracking, and saliency detection. These descriptors can be classified into two categories – the hand-crafted feature descriptors and the learned ones. Two common types of hand-crafted descriptors are distribution-based [1, 2, 3], and binary descriptors [4, 5].

Learned descriptors have recently gained a lot of attention. The success of deep Convolutional Neural Networks (CNNs) in various image processing tasks has encouraged their usage for patch descriptors [6, 7, 8, 9], showing excellent results in patch-matching. The CNN-based learning methods are supervised, i.e. trained with pairs of patches that are labeled as similar or dissimilar. The learned similarities and dissimilarities between image patches may not hold when they are affected by some degradation type that was not included in the training set.

An alternative is unsupervised learning based on autoencoders [10, 11]. Chen et al. [12] applied autoencoders to the general problem of patch descriptors. Their autoencoder-learned descriptor shows promising results, however,

[*] Corresponding author

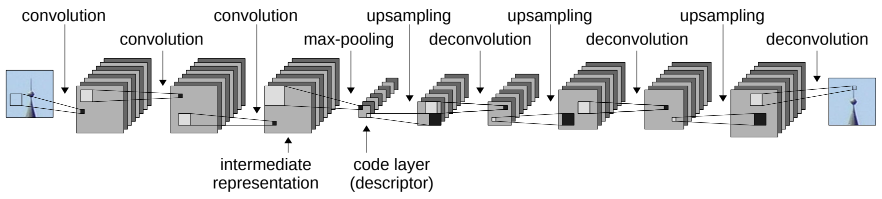

Fig. 1. The proposed autoencoder architecture. There are no max-pooling layers after the first two convolutions in order to obtain an intermediate representation (IR) of image that preserves the spatial information in the height-width plane.

its calculation time and memory is infeasible for higher resolution images. Their fully-connected network does not take advantage of the local similarity property of natural images, has more parameters to be trained, and requires longer training times than the convolutional autoencoder designs. Moreover, the descriptor from [12] does not allow different input sizes and therefore a separate autoencoder needs to be trained for every patch size, which renders the framework impractical for many applications.

We propose a novel autoencoder-based patch descriptor designed for applications with many patch comparisons within a single image. We design a specific network architecture that yields a special image representation that we call the *intermediate representation* (IR). The benefits of having a direct access to IR are twofold: (i) patch descriptors can be obtained from IR with a simple operation, and (ii), IR is structured such that overlapping patches' representations are overlapping themselves, resulting in a unique memory-saving property that, to our knowledge, does not hold for any other patch descriptor.

Besides, the introduction of IR enables incorporating contextual information beyond the patch borders into its descriptor, making the descriptors more robust to erroneous and missing parts of image patches. We employ convolutional layers in our method such that our descriptor can work with patches of different sizes without the need to retrain the autoencoder (which is an important practical advantage over [12]). This flexibility results in faster learning and wider applicability in the processing of natural images.

In the following section, we give a brief introduction to the autoencoders. Section 3 contains the description of our method, and in Section 4 we present and discuss the experimental results. We conclude the work in Section 5.

2 Preliminaries

Autoencoders are unsupervised neural networks used for learning efficient representations of data. An autoencoder consists of two parts, an encoder and a decoder, and is trained by minimising the reconstruction error between the input and output, while imposing some constraints on the middle layer.

Formally, an autoencoder with encoder $\mathcal{E}$ and decoder $\mathcal{D}$ is trained to minimise the loss function $J(X, \mathcal{E}, \mathcal{D}) = \sum_{x \in X} \mathcal{L}(x, \mathcal{D}(\mathcal{E}(x))) + \Omega(\mathcal{E}(x))$, where $x \in X$ is a data sample, $\mathcal{L}$ is some metric and $\Omega(\mathcal{E}(x))$ is an optional sparsity regularisation term imposed on the hidden (code) layer. Autoencoders working with image data usually consist of alternating convolutional and max-pooling layers. The output neuron at location (i, j) in the k-th channel of the l-th convolutional layer is calculated as follows:

$$x_{ij}^{(l,k)} = \sum_{c \in C} \sum_{u=0}^{f^{(l)}-1} \sum_{v=0}^{f^{(l)}-1} w_{uv}^{(l,k)} x_{(i+u)(j+v)}^{(l-1,c)} + b^{(l)}, \tag{1}$$

where C is the set of channel indices, $w^{(l,k)}$ is the convolutional kernel for the l-th layer and k-th channel, $b^{(l)}$ the bias for the l-th layer, and $f^{(l)}$ is the size of the convolutional kernel (filter) for the l-th layer.

Chen et al. [12] proposed learning descriptors with an autoencoder that consists of a single hidden layer, which was fully connected with the input and output layers of the network. The encoder part of the network, i.e. the descriptor of the image patch x, was calculated as $\mathcal{E}(x) = S(W_{\mathcal{E}} x + b_{\mathcal{E}})$, where $W_{\mathcal{E}}$ and $b_{\mathcal{E}}$ are the weights and the bias in the encoding layer respectively, and $S(\cdot)$ is the sigmoid function. The authors used the loss function $J(X, \mathcal{E}, \mathcal{D}) = \frac{1}{2|X|} \sum_{x \in X} (x - \mathcal{D}(\mathcal{E}(x))) + \Omega_1(\mathcal{E}(x)) + \Omega_2(\mathcal{E}, \mathcal{D})$, with $\Omega_1(\cdot)$ being the dimensionality constraint on the middle (code) layer, and $\Omega_2(\cdot)$ the sparsity constraint on the weights of the network. The layers in this network can be represented as convolutions with a kernel of the same size as the input. Hence, they are fully-connected layers and not convolutional in the common sense of the term. The use of these fully-connected layers fails to exploit the local self-similarity property of natural images, requires longer training time compared to convolutional network designs, and requires training a new network for each patch size.

3 Proposed method

Our primary contribution is a novel autoencoder architecture that provides an intermediate representation (IR) of an image. The use of IR has two main advantages: it is less memory-intensive than storing the descriptors of all patches within an image, and it allows a descriptor of a single patch to be extracted from IR with minimal computation. To accommodate this, we take advantage of the properties of the convolutional layers described in Section 2, but discard all the max-pooling layers in the encoder except for the last one. The convolutional layers exploit the self-similarity property of natural images to reduce the computational time while also obtaining better results than fully-connected layers. Moreover, the use of convolutional layers is critical for the ability to extract patch descriptors from the IR, since convolutional layers preserve spatial information in the height-with plane.

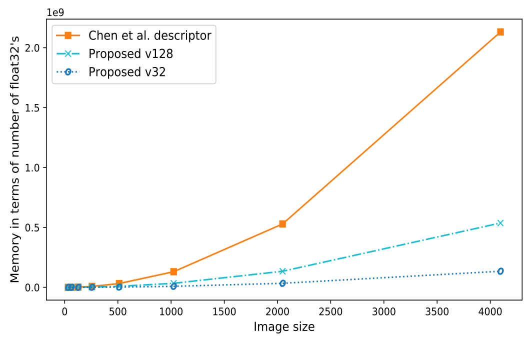

Fig. 2. A comparison of the memory requirements (expressed in the number of float 32's) as a function of image size in pixels, for the two versions of the proposed descriptor (v32 and v128) compared to Chen et al. [12].

Max-pooling is typically applied since it adds extra non-linearity, plays the role of dimensionality reduction, and reduces the number of training parameters and hence the training time. However, successful neural network architectures have been recently reported also without max-pooling [13]. We omit max-pooling after the first two convolutions and employ non-linear activation functions. We leave only one max-pooling layer with large spatial extent at the end of the encoder to reduce the dimension of the code layer. This architecture requires longer training time compared to standard use of max-pooling, but reduces the computational time and memory while using the descriptor. This is beneficial since the training needs to be done only once.

The reduction of computational time and the memory usage follows from the IR in our network. An IR is obtained by propagating the complete image (containing patches of interest) through the convolutional layers in the encoder, but not the max-pooling layer. Figure 1 shows the architecture of our network and the IR.

Let $x := x^{(0,:)}$ be the input image. We define the intermediate representation as $\mathcal{IR}(x) = x^{(L,:)}$, with

$$x^{(L,c)} = \mathcal{A}(\mathcal{C}_{l_L}(\mathcal{A} \dots (\mathcal{C}_{l_1}(x^{(0,:)})))), \tag{2}$$

where L is the number of convolutional layers in the encoder $\mathcal{E}$, $x^{(l_i,c)}$ is the c-th channel of the output of the l_i-th layer, $x^{(l_i,:)}$ denotes all channels of the output of the l_i-th layer, $\mathcal{A}$ is some activation function, and $\mathcal{C}_{l_i}$ is the l_i-th convolutional layer. From the intermediate representation of an image $\mathcal{IR}(x)$, we obtain the descriptor for a patch $x_{(i,j)}$, whose upper left corner is positioned at (i, j), as follows

$$\mathcal{E}(x_{(i,j)}) = \mathcal{MP}(\mathcal{IR}(x)_{(i,j)}), \tag{3}$$

i.e. by performing the max-pooling on the patch of the IR.

For the activation function $\mathcal{A}$ from (2), we have chosen the sigmoid function, $S(x) = \frac{1}{1+e^{-x}}$. This choice was made in order to avoid the case of having many zeros as an output, which was the case with some other activation functions, such as a rectifier. We trained the network with Adadelta optimizer [14], and we used binary cross entropy as a loss function (we scale the pixel values to be in $[0, 1]$):

$$J(X, \mathcal{E}, \mathcal{D}) = \sum_{x \in X} \mathcal{L}(x, \hat{x}) = \sum_{x \in X} (x \log(\hat{x}) + (1 - x) \log(1 - \hat{x})) \tag{4}$$

where $\hat{x} := \mathcal{D}(\mathcal{E}(x))$ is the output of the autoencoder. We trained the autoencoder in two different ways, creating two versions of the descriptor, v32 and v128 (named after the dimensionality of the descriptor for 16×16 patches). For the implementation, we use Keras library for neural networks. The autoencoder has been trained on a total of 150k 16×16 patches cropped from images from the ImageNet dataset [15]. The ratio between training, validation, and test set is $8 : 1 : 1$.

In Figure 2 we compare the effective memory usage required by different descriptors depending on the image size. These results indicate potential for a tremendous decrease in memory usage for applications on a single image. This decrease could make some algorithms that use many patch comparisons feasible for use on larger images.

4 Experimental evaluation

4.1 Robustness to noise

We test the noise robustness of both versions of our descriptor, comparing them to the exhaustive search on pixel intensity values, and the existing descriptor trained with the autoencoder of [12]. We trained all the descriptors on the same set of colour patches.

The evaluation is performed as follows. We select a set of query patches within an image with added Gaussian noise. For each query patch, we retrieve the k most similar patches either by comparing their descriptors or by using exhaustive search over the pixel values. The quality of patch retrieval is evaluated based on the sum of square differences (SSD) between the pixels in query and retrieved patches before the noise was added. The standard deviation of the Gaussian noise was varied between 0 and 50.

Figure 3 summarises the results of our patch retrieval experiments and some visual examples are shown in Figure 4 (left). When no noise is present, the exhaustive search retrieves the patches that are the most similar to the query patch. However, noise deteriorates the performance of the exhaustive search, whereas our descriptor v128 shows little decrease in performance.

Our descriptors also show superior performance compared to the existing descriptor learned with autoencoders. Our method v128 shows significantly

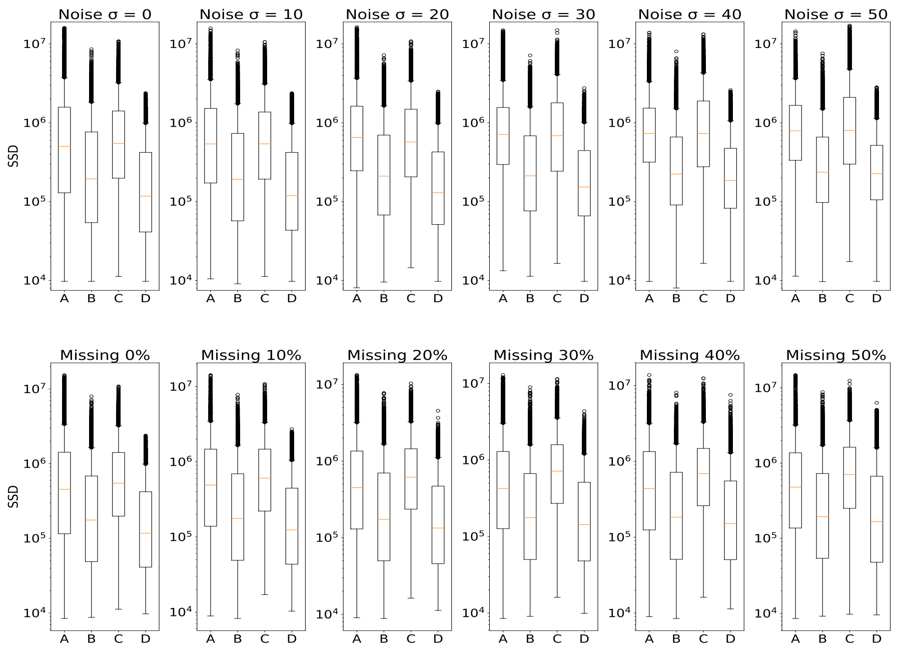

Fig. 3. Comparison of descriptors' robustness to noise (top) and missing data (bottom). A – proposed descriptor v32, B – proposed descriptor v128, C – Chen et al. [12], D – exhaustive search on pixel intensity values. The plots are showing SSDs of ground truth pixel values of patches found by the descriptors (in A-C) and exhaustive search (D), based on the noise (top) and percentage of missing area in a patch (bottom).

better results than Chen [12], while having the same patch descriptor dimensionality. Furthermore, our method v32 that shows similar results to [12] has an order of magnitude lower dimensionality of the descriptor when encoding a single patch. The dimensionality comparison changes even more in our favour when encoding the whole image due to the usage of the IR (Figure 2).

4.2 Robustness to missing data

We set up an experiment to determine the capability of the proposed descriptor when working with patches that contain missing regions. This type of operation is of interest for applications such as inpainting and scene reconstruction from multiview data. The setup is similar to the noise robustness evaluation, but here parts of the query patches have been randomly removed.

For the query patches with missing parts we want to find the best matching undamaged patches. We are searching for the matching patches by comparing

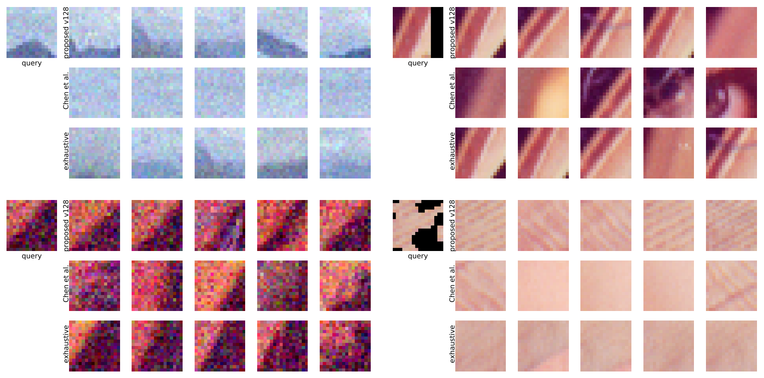

Fig. 4. Noisy patch retrieval (left) and patch retrieval where the query has missing parts (right). For each query, the first row corresponds to the proposed descriptor v128; the second row: the descriptor from [12], and the third row: exhaustive search. The missing parts of the query patches on the right are shown in black.

the descriptors of the non-missing parts. The numerical evaluation is done based on the SSD values of the complete (undamaged) query and the found match. The results are shown in Figure 3 (bottom), again comparing our two descriptors, descriptor from [12], and the exhaustive search over pixel intensity values. Visual comparison is shown in Figure 4 (right).

The conclusions from these experiments are similar to those with the noisy patches. When the missing area in a patch is small, exhaustive search retrieves the best results. However, as the missing area is increasing, our descriptor v128 starts performing better and overtakes the exhaustive search, showing more robustness to missing data than the exhaustive search. Both of our proposed methods outperform the existing method [12], with a slightly larger margin than in the case of noisy patches. More elaborate analysis of the experiments is in our extended journal submission [16].

5 Conclusion

We propose a new method for learning local image descriptors using autoencoders. The proposed approach saves memory and computational time in comparison to existing methods when used for patch search and matching within a single image. We have evaluated the proposed descriptors' robustness to noise and missing data against an existing descriptor learned with autoencoders from [12] and exhaustive search over pixel intensity values. The proposed descriptors show improved results, and superior robustness to both noise and missing data in comparison with exhaustive search.

References

1. D. G. Lowe, “Object recognition from local scale-invariant features,” in *ICCV*. IEEE, 1999, p. 1150.
2. R. Arandjelović and A. Zisserman, “Three things everyone should know to improve object retrieval,” in *2012 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2012, pp. 2911–2918.
3. H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool, “Speeded-up robust features (surf),” *Computer vision and image understanding*, vol. 110, no. 3, pp. 346–359, 2008.
4. M. Calonder, V. Lepetit, C. Strecha, and P. Fua, “Brief: Binary robust independent elementary features,” in *European conference on computer vision*. Springer, 2010, pp. 778–792.
5. E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, “Orb: An efficient alternative to sift or surf,” 2011.
6. S. Zagoruyko and N. Komodakis, “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.
7. E. Simo-Serra, E. Trulls, L. Ferraz, I. Kokkinos, P. Fua, and F. Moreno-Noguer, “Discriminative learning of deep convolutional feature point descriptors,” in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 118–126.
8. V. Balntas, E. Riba, D. Ponsa, and K. Mikolajczyk, “Learning local feature descriptors with triplets and shallow convolutional neural networks.” in *BMVC*, vol. 1, no. 2, 2016, p. 3.
9. K. He, Y. Lu, and S. Sclaroff, “Local descriptors optimized for average precision,” in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.
10. G. E. Hinton and R. S. Zemel, “Autoencoders, minimum description length and helmholtz free energy,” in *Advances in neural information processing systems*, 1994, pp. 3–10.
11. G. E. Hinton and R. R. Salakhutdinov, “Reducing the dimensionality of data with neural networks,” *science*, vol. 313, no. 5786, pp. 504–507, 2006.
12. L. Chen, F. Rottensteiner, and C. Heipke, “Feature descriptor by convolution and pooling autoencoders,” *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences-ISPRS Archives 40 (2015), Nr. 3W2*, vol. 40, no. 3W2, pp. 31–38, 2015.
13. J. T. Springenberg, A. Dosovitskiy, T. Brox, and M. Riedmiller, “Striving for simplicity: The all convolutional net,” *arXiv preprint arXiv:1412.6806*, 2014.
14. M. D. Zeiler, “Adadelta: an adaptive learning rate method,” *arXiv preprint arXiv:1212.5701*, 2012.
15. J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, “Imagenet: A large-scale hierarchical image database,” 2009.
16. N. Žižakić, I. Ito, and A. Pižurica, “Efficient local image descriptors learned with autoencoders,” *(in submission)*.