Pattern Recognition Letters 131 (2020) 398–404

Contents lists available at ScienceDirect

Pattern Recognition Letters

journal homepage: www.elsevier.com/locate/patrec

Improving pattern spotting in historical documents using feature pyramid networks

Ignacio Úbeda[a], Jose M. Saavedra[b], Stéphane Nicolas[c], Caroline Petitjean[c], Laurent Heutte[c,*]

[a] University of Chile, Beauchef 850, Santiago 8370448, Chile
[b] Computer Vision Group, Orand S.A., Estado 360, Santiago 8320180, Chile
[c] Normandie Univ, UNIROUEN, UNIHAVRE, INSA Rouen, LITIS, Technopole du Madrillet, Saint-Étienne-du-Rouvray 76800, France

ARTICLE INFO

Article history:
Received 23 July 2019
Revised 29 January 2020
Accepted 1 February 2020
Available online 3 February 2020

Keywords:
Pattern spotting
Image retrieval
Historical documents
Convolutional neural network

ABSTRACT

Pattern spotting consists of locating different instances of a given object (i.e. an image query) in a collection of historical document images. These patterns may vary in shape, size, color, context and even style because they are hand-drawn, which makes pattern spotting a difficult task. To tackle this problem, we propose a Convolutional Neural Network (CNN) approach based on Feature Pyramid Networks (FPN) as the feature extractor of our system. Using FPN allows to extract descriptors of local regions of the documents to be indexed and queries, at multiple scales with just a single forward pass. Experiments conducted on DocExplore dataset show that the proposed system improves mAP by 73% (from 0.157 to 0.272) in pattern localization compared with state-of-the-art results, even when the feature extractor is not trained with domain-specific data. Memory requirement and computation time are also decreased since the descriptor dimension used for distance computation is reduced by a factor of 16.

1. Introduction

Digitalization of historical document collections helps grant access to our cultural heritage to more users while limiting contact with the real materials. As the amount of data at our disposal has grown due to technological advances, there is now a need for efficient search in vast repositories of digitized images. For historians, there is a need to have an automated software tool that allows them to establish correspondences between documents or parts of documents, whether on textual content or on graphical parts [27].

For the recognition of textual content, in both printed and handwritten documents, recognition techniques have made great progress, especially thanks to deep learning techniques (CNN, RNN, BLSTM) [22]. Word spotting may also be an interesting alternative to full text recognition in an information retrieval perspective, which consists in searching for the multiple instances of a given word in a collection of document images, thanks to a single query image (query by example paradigm) [8]. On the other hand, historical documents contain also some graphical patterns or objects which are often interesting for historians but there is little work indeed on the automatic analysis of their graphical content. Most content-based image retrieval (CBIR) algorithms are used to search for entire images in databases (“image retrieval”) but no tool to locate regions (“sub-image retrieval”) within the historical documents has been proposed. Computers can solve the former task with quite good results but precise localization of patterns, particularly the smallest ones, is still a challenging task.

In [27] the authors explored the question of category level object detection, for semantic based indexing, in the context of a benchmark dataset for cultural heritage studies. They proposed a benchmark image dataset of medieval images with ground-truth information and a detection system that accurately localizes objects. En et al. [5,6] have also proposed a benchmark dataset, called “DocExplore” which is publicly available online [4], and a complete system able to perform image retrieval and pattern spotting, using query by example paradigm, as shown in Fig. 1.

Whereas previous work on the same dataset was based on classical (handcrafted features) approaches [6], in this paper we propose to rely on convolutional neural networks (CNN) for feature extraction and to build upon the method presented in [24]. The main contribution of this work is the development of a CNN-based local feature extraction for *pattern spotting* (retrieval and localization) that allows to have region descriptors at multiple scales on one forward pass.

* Corresponding author.
E-mail addresses: ignacio.ubeda@ing.uchile.cl (I. Úbeda), Laurent.Heutte@univ-rouen.fr (L. Heutte).

https://doi.org/10.1016/j.patrec.2020.02.002
0167-8655/© 2020 Published by Elsevier B.V.

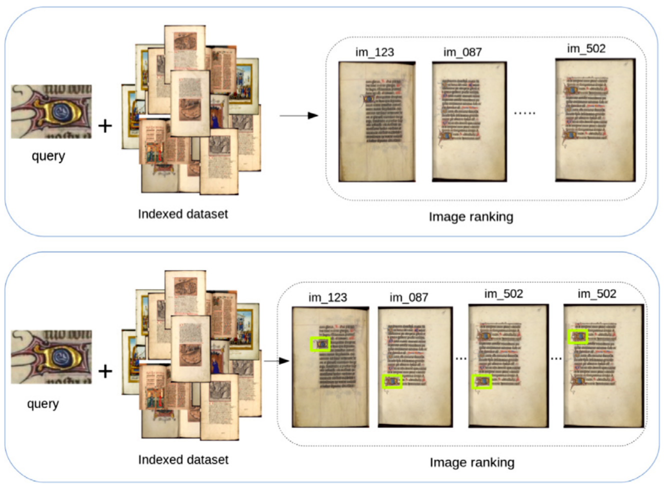

Fig. 1. Document image retrieval (top) and pattern spotting (bottom), using query by example paradigm.

Organization of the paper is as follows. Section 2 discusses related work with a focus on CNN applied in CBIR. In Section 3 we detail our system proposal. Section 4 explains the adopted protocol to evaluate the system and discusses the obtained results. We conclude this paper in Section 5 by proposing future works.

2. Related work

Content-based image retrieval has been an extensively studied topic for a long time. Early works have proposed hand-crafted - such as color or textures based - global features to address this problem [20]. Extracting local rather than global features with SIFT [19], and then using aggregation techniques, such as VLAD [12] or Fisher Vector (FV) [13], to generate compact embeddings[1] have been proposed more recently.

One work [6] with this SIFT-based approach is of particular interest. The system starts by identifying patches with BING (Binarized Normed Gradients) [3]. DenseSIFT descriptors of each patch are then aggregated using VLAD and FV to obtain the final patch embedding. At the end, a query similarity search integrating compression and approximation techniques (Inverted File, Product Quantization and Asymmetric Distance Computation) is used for pattern localization. While this system has shown good performance on the corpus of document images of interest on *image retrieval*, there is a large room for improvement for *pattern spotting*. In addition, the system suffers from a number of weaknesses that make it unsuitable for other types of document images (color information is not currently used, for example) and requires post-processing for fine localization of objects in regions of interest using, for example, traditional matching methods [7].

Since [15], CNN have been successfully used in many computer vision tasks, such as classification or object detection, and CBIR has been no exception. Several works [2,23] have shown that the use of pre-trained networks as feature extractors reports state-of-the-art results compared to classical approaches. The same happens in the context of document analysis [11]. Of course, when the pre-trained CNN is retrained with context-specific data (something refered as “fine-tuning”), results are pushed even higher. However, collecting training samples and retraining the CNN require significant amounts of human and computing resources [28]. A survey about the transition from classical SIFT-based to CNN-based approach in CBIR field can be found in [31].

For sub-image retrieval, local features are more important than global features and for that case, it has been seen that using the outputs from the convolutional layers (refered as “feature map”) can be interpreted as local embeddings describing particular image regions (hereinafter refered as “patches”) [1,14]. Even further, different feature maps can be used to obtain representations of these patches at multiple scales [9].

Other CNN-based approaches have been proposed to obtain patch representations. For example, in [25] they start by sampling patches and then a pre-trained CNN is feeded with each of them. The first fully-connected layer is used along with FV aggregation to finally generate patch embeddings. This gives rise to hybrid models in which CNN is used as the extractor of local features and these are then aggregated using either classical or new techniques [1,14]. Metric learning approaches have also been proposed [10,26,29] for patch-based matching in which not only representations are learned but also a similarity measure between the extracted embeddings.

In [24] we proposed a CNN based approach to obtain patch representations in the context of *pattern spotting*. Our feature extraction (detailed in Section 3) is composed by a pre-trained Feature Pyramid Network (FPN) [16] that provides multi-scale patch representation. The procedure is similar to that used in [21] with one main difference: instead of constructing an image pyramid and then applying a CNN for each level independently, we do the same in one forward pass. Although we showed in [24] that our CNN based approach enabled to reach the best spotting results on the dataset used for evaluation, it had some limitations, especially when querying with small and non square patterns. We show in the next section how we dealt with those issues.

3. Our deep proposal

Our proposal is composed of two stages, one offline and the other online. The offline stage is focused on processing and indexing the collection of historical documents to be queried while the online stage is focused on processing and locating, in the collection of indexed documents, the multiple instances of the object similar to the input query image. An overview of the pipeline is given in Fig. 2. The system is based on the work presented in [24] with several key modifications to cope with the limitations mentioned before, which allow to significantly improve the performance. Particularly in this work, we offer some solutions to deal with: (1) small queries by adding a finer visual representation and (2) non-square queries using multiple embedding representations. Finally, we also added a feature normalization step to the pipeline introduced in [24] to better cope with the correlation of features and fit the similarity measure used. Hereafter, we detail each processing step.

3.1. *Offline phase*

As shown in Fig. 2, the offline phase is made up of four steps for indexing the collection of historical documents.

3.1.1. *Document preprocessing*

The first stage consists in retaining only informative regions of each page, i.e. regions that may contain symbols and/or handwriting but not the background. To this end, we used the algorithm for background removal proposed in [6] that is based on a region growing paradigm starting from the center of the page and, iteratively, adding border pixels until some condition is met.

We fix the input shape of the CNN to $I \times I$ pixels. If the page is smaller than I in both dimensions, it is centered on a black canvas (equivalent to zero-padding). On the other hand, if the page exceeds the I size in one (or both) dimension(s), we divide the page

[1] In this work we use the words: “embedding”, “feature vector” and “representation” indistinctly.

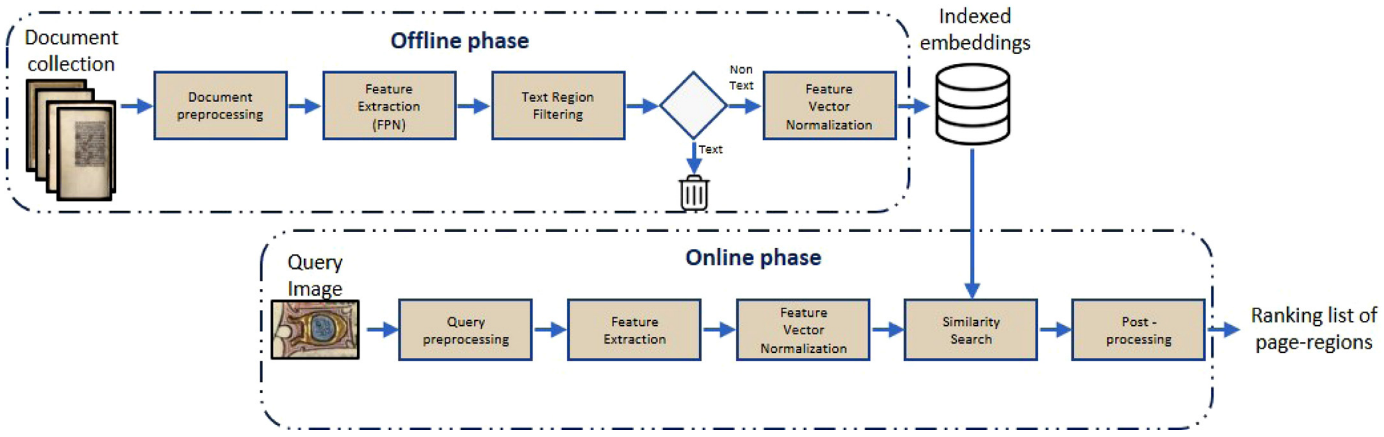

Fig. 2. Overview of the pipeline with the offline and online procedures.

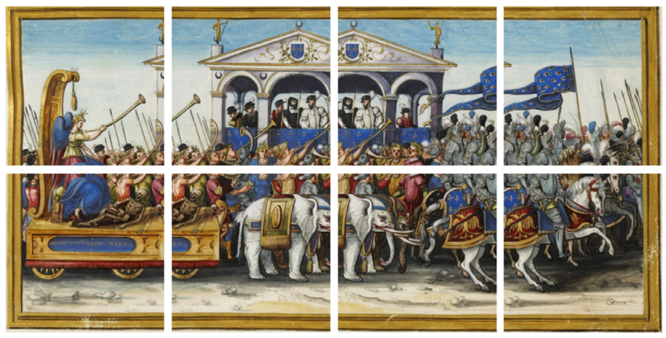

Fig. 3. Sub-pages example. All sub-pages have the same shape ($I \times I$); overlapping may exist between them.

through the following procedure: first, we locate the corner centers matching a $I \times I$ template in each corner; then, we create an equally 2D spaced grid with those corner centers and finally we extract an $I \times I$ patch of the page at each center of the grid. Fig. 3 show the output sub-pages for an example page greater than I in both dimensions.

Note that the "sub-pages" obtained through this procedure might overlap between each other and also some of them might be smaller than $I \times I$ thus zero-padding is also needed for those cases. Note also that the images are not resized in order to keep graphical details because some queries can be very small and we want to spot them as well.

3.1.2. Feature extraction

The RetinaNet network architecture [17], pre-trained on COCO dataset [18] without fine-tuning, is used as feature extractor for both pages to be indexed, and queries. For the pages to be indexed, $\{P_2, P_3, P_4, P_5\}$ levels are extracted from the feature pyramid created by the FPN backbone of RetinaNet[2] during the forward pass. To each "pixel" of each feature map (refered as a "neuron") is associated a feature vector (due to the depth of each feature map) that comprises a visual representation of its receptive field. We call this feature vector an "embedding". Here, shallow layers extract embeddings for smaller regions and deeper layers for bigger ones so we get embeddings in a patch-based manner at different scales for each page.

The storage needed to index all the embeddings depends mainly on the network architecture and the spatial resolution of the input. For our setting, the different levels of the pyramid, P_i, $i \in \{2, 3, 4, 5\}$, have a spatial resolution of $(I/2^i, I/2^i)$ with a depth of 256. Although $I/2^i * I/2^i$ embeddings are available for each level P_i, not all of those embeddings are useful (e.g. the embedding that describes the black canvas is not required). Hence, we can remove them in an upcoming stage to reduce storage consumption and search space in the online procedure.

3.1.3. Text region filtering

Storing the complete pyramid for each page would be memory costly but also the retrieval procedure would be computationally expensive (an exhaustive search would have to be carried out on the complete feature maps). In addition, as a user will always search for a graphical pattern, queries will appear in graphical regions instead of in text regions or in regions corresponding to the - artificially created - black canvas. We can address this problem by training a classifier for removing textual and canvas regions.

We trained a supervised classifier at each level of the pyramid using the 256D embeddings of the complete feature map of the considered level (i.e. the feature vectors used as inputs of the classifier are provided by the network). The label assigned to each feature vector for training is simply obtained by looking at the intersection between the receptive field of each neuron and the bounding boxes of non textual regions of interest (ROI) that we manually labeled on 79 pages of the collection of 1500 pages. Three classes are considered: *black* (for the black canvas pixels), *text* and *non-text*.

Particularly, if the intersection between all non textual ROI and the corresponding receptive field is above a fixed threshold, then the embedding is tagged as *non-text* class, otherwise, as *text* or *black* depending of how many black canvas pixels are inside the receptive field. Then we collect for all the pages of the dataset, all the embedding-label pairs at each level and split them in train/validation/test set with 0.6/0.25/0.15, respectively.

We finally use these classifiers to make predictions for all pages by stacking the feature map of each level in the first dimension $(n_{embeddings}, n_{features})$ and then predicting with the corresponding level classifier. This way, batches of (predicted) non textual embeddings are created for each level and the complete feature pyramid is discarded.

3.1.4. Feature vector normalization

We use Principal Component Analysis (PCA) to transform the batches of embeddings and we apply l_2 normalization on each feature vector as well. Note that instead of using PCA for dimensionality reduction, we use it as a feature decorrelation process, which is desirable for many pattern analysis applications [30]. Furthermore, l_2 normalization is carried out to ensure equivalence of

[2] We keep the same notations as in [16].

different similarity measures (e.g. cosine similarity, dot-product similarity and Euclidean distance).

Both transformation (PCA fitting and transforming) and l_2 normalization are carried out with the complete batch of non textual embeddings obtained from the previous stage at each level independently. Finally, the transformation matrix is stored to apply it during the online procedure as well.

3.2. Online phase

As shown in Fig. 2, the online phase is made up of five steps for locating, in the collection of indexed documents, the multiple instances of the object similar to the input query image.

3.2.1. Query preprocessing

We apply the same procedure for centering the queries in canvas for the online phase but instead of using a black one, we use a "blank" background page from a manuscript, in order to reduce the noise introduced by black color. Note that those black regions are filtered in the offline phase which is not possible to do in the online phase.

3.2.2. Feature extraction

Contrary to the offline phase where a pyramid of feature maps is extracted for each page to be indexed, during the online phase we extract embedding(s) for the query in just one level of the pyramid. In particular, we assign a query of width w and height h to the level P_k by Eq. (1) and we set $k_0 = 4$ as in [16]:

$$k = \left\lfloor k_0 + \log_2(\sqrt{wh}/224) \right\rfloor \tag{1}$$

Our experiments were conducted considering two cases: one embedding and multiple embeddings per query. As the queries are centered, we keep the center neuron at determined level P_k as the embedding for the query in the former case and we keep the center neuron with its n symmetrical neighbors (on the same level P_k) in the latter. We discuss in detail the impact of adding multiple embeddings in Section 4.3.2.

3.2.3. Feature vector normalization

The same normalization procedure described for the offline phase (transformation and l_2 normalization) is applied during the online phase. To this end, the fitted transformation matrix of the assigned level P_k, previously determined during the offline phase, is applied on the query embedding(s). On the other hand, l_2 normalization only requires its own embedding information.

3.2.4. Similarity search

Queries are searched only at the same level to which they were assigned (e.g. if a query is assigned to P_4, then it is looked for only at the P_4 level of batches). Cosine similarity is used as the similarity measure but since the embeddings are l_2 normalized, dot product can also be used to speed up the search. Recall that each embedding is a visual representation of a specific patch of the input image based on the receptive field of the associated neuron. Hence this search will find the closest patch from the pages to the query.

3.2.5. Post-processing

Coordinate translation is applied since the pages are cropped by the background removal stage (and some of them divided into sub-pages). For localization, the query template is centered on the found patch center (already translated). Finally, a filtering step is added to discard the localization if the bounding box is not entirely contained in the original page (particularly useful for big queries) and non-maximum (hereinafter refered to as "non-max") suppression is carried out on the obtained ranking to avoid overlapping bounding box retrievals for the same instance.

Table 1
RandomForest classifier results on the test set for the text region filtering procedure.

	P_2	P_3	P_4	P_5
Recall	0.994	0.997	0.991	0.985

4. Results and discussion

4.1. Experimental protocol

We use the DocExplore dataset [4,5] to evaluate the performance of our system. DocExplore is a publicly available dataset composed of medieval (hand-drawn) illuminated manuscripts dated back from 10th to 16th century. In particular, 1500 pages and 1447 queries among 35 different categories are used for the experimentation. Two tasks, *image retrieval* and *pattern spotting*, are considered for mAP (mean Average Precision) evaluation. The aim of the *image retrieval* task is to retrieve pages that contain the given query, regardless of its position in the page. On the other hand, the *pattern spotting* task aims at localizing the queried graphical pattern, with a bounding box, within the retrieved page (note that a single page may contain multiple instances of the same query). In the latter, a correct retrieval is considered when the IoU (intersection over union) with the ground truth is above 0.5. We used the evaluation kit provided in [4]. Particularly for our experiments, we set $I = 1000$ (input image shape) and we considered the first m retrieved patches provided after non-max suppression for evaluating the system. We set $m = 1000$, which means that if N is the total length of the complete ranking, this value of m yields to $m \ll N$.

4.2. Text region filtering results

Several classifiers were trained and model selection was performed using the validation set. Since the classifier is used for removing textual regions it may be seen as a non textual region proposal procedure. Thus, recall of *non-text* class is the most important metric in this case (high recall of *non-text* class would imply that all non textual regions have correctly been classified). Recall rates for the best classifier (RandomForest) at each pyramid level are given in Table 1: one can see that thanks to high recall rates no potential graphical region is missed.

4.3. Image retrieval and pattern spotting results

Since we introduce two modifications to deal with small queries as well as non-square queries, we focus in this section on those two changes.

4.3.1. Finer visual representation

Small queries are the most difficult to spot and almost 96% of the DocExplore dataset is composed of them. Thus, it is an important problem and we deal with it by including the P_2 level of the RetinaNet model to have a finer (greater resolution) level than P_3. Fig. 4 shows receptive field centers for all neurons at P_2 and P_3 levels. It can be seen that with P_2 a denser grid is now considered and more embeddings could match small query embedding. In particular, the distance between two patch centers of two neighboring neurons in the same feature map is 2^i pixels for level P_i.

The impact of including P_2 is shown in the first two lines of Table 2. Compared with [24], respectively 31% and 57% of improvement are obtained in *image retrieval* and *pattern spotting* tasks with the P_2 and T&N (transformation and normalization of embeddings) setting. The main cost of this improvement is at execution time. In the online phase, two main procedures are carried out: (1) searching for the query embedding in the indexed patch-page

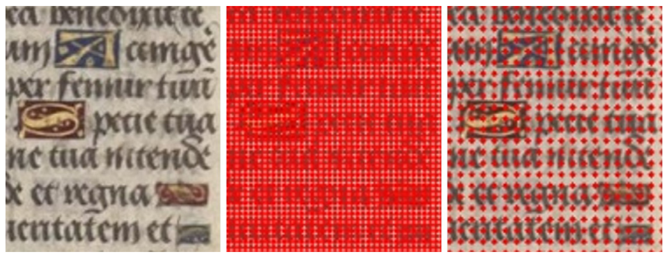

Fig. 4. Receptive field centers (red points) for all neurons at P_2 (middle) and P_3 (right) for the page after background removal (left). The page has been cropped for better visualization. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Table 2
mAP results for *image retrieval* and *pattern spotting* task, with/without P_2, with/without transformation and normalization (T&N) of embeddings. *Results reproduced from [24].

Configuration	*image retrieval*	*pattern spotting*
without P_2*	0.386	0.173
with P_2	0.468	0.242
with P_2 + T&N	0.505	0.272

embeddings and (2) creating the ranking for the query with non-overlapping bounding box.

Both procedures increase their execution time, respectively search and ranking time, with the addition of P_2 because more embeddings have to be indexed due to bigger spatial resolution, increasing the search time and, as said before, P_2 tends to return several patches of the same pattern, increasing the ranking time. In particular, search time is increased by a factor 3 (from roughly 7 to 21 s on a regular laptop) when the query is searched at P_2 instead P_3 and ranking time is increased from 1.5 to 2.3 s per query. In the case of T&N modification, indexed embeddings can be transformed and normalized during the offline phase hence only the query embedding should be processed during the online phase, which is marginal compared with the search time (less than 1 s per query).

Our hypothesis was that P_2 would improve performance for small queries. Because of the way queries are assigned to a level, the only categories that may be affected are the smallest ones (those that were assigned to P_3 in the previous version of the system, and are now assigned to P_2). Our goal here is to identify and characterize the categories which took the most benefit from the addition of P_2 (i.e. we do not take into account the T&N modification). Student's *t*-test was carried out to compare the means of AP (Average Precision) values, with and without P_2, and this was repeated for each category. In Fig. 5 we show a scatter plot of categories with highest improvement (significance level of 0.05), with respect to width and height size. Smallest categories are most benefiting from the P_2 addition, with the exception of `rubanlettrine`, as well as categories which are closed to the dashed line, i.e. relatively square. What these categories have in common is that unlike in the previous system without P_2, now the receptive field of the center neuron is fully contained in the query, hence allowing a noiseless embedding. In Fig. 6 we show an instance of two of these categories with the old receptive field (P_3) and the new one (P_2). Note that in the case of `rubanlettrine`, P_2 receptive field is "blind" because it is only looking at the center of the query instead of P_3 receptive field. However, this turns out to be better as it is now fully contained in the query.

The above analysis suggests that best results are obtained when the receptive field is completely contained in the area covered by the query (noiseless embeddings are extracted). However, blind

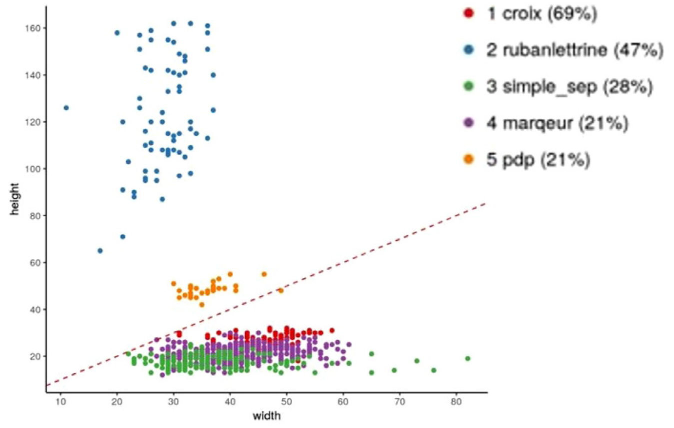

Fig. 5. Scatter plot of top 5 highest improvement categories with P_2 with respect to the query size. Relative improvement is shown in parenthesis.

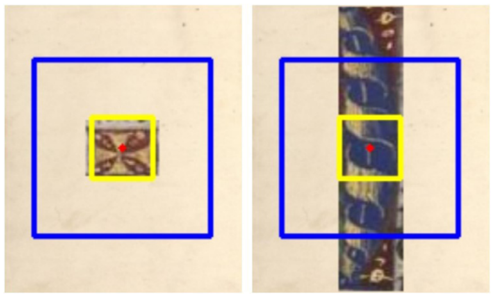

Fig. 6. Receptive field for P_2 (yellow) and P_3 (blue) for an instance of the "croix" (left) and "rubanlettrine" (right) object categories. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

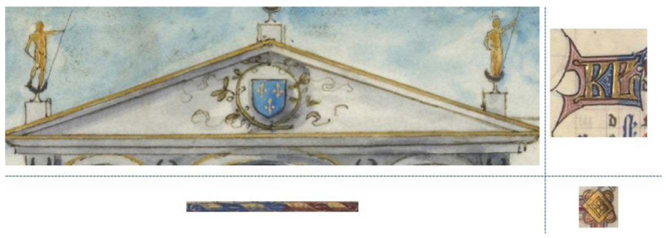

Fig. 7. Examples of queries in each of the 2 × 2 groups. Groups are: big&non-square (top-left), big&square (top-right), small&non-square (bottom-left) and small&square (bottom-right).

receptive fields are not desirable because they may loose crucial query information. A trade-off should be found between maximizing query area covered by the receptive field and minimizing noise introduced by the canvas. Queries that best balance this trade-off are those with shape closer to the shape of the receptive field of the assigned level. Since receptive fields are square because of square convolutional kernels, we expect that best results are coming from square queries as well. We aim at validating this statement with a deeper analysis.

For this analysis we categorize two variables: size $w \times h$ and aspect ratio h/w, each into two levels of categorization: small/big and square/non-square and we assign each query to one of the 2×2 groups created with this categorization. For the cutting points we set $\log(\text{size}) = 10$ and a tolerance of 0.2 in the aspect ratio (i.e. if ar_q is aspect ratio of query q, then the query is square iff $|ar_q - 1| \leq 0.2$). We set these values because of query size distribution and visual validation. Examples of each category are shown in Fig. 7.

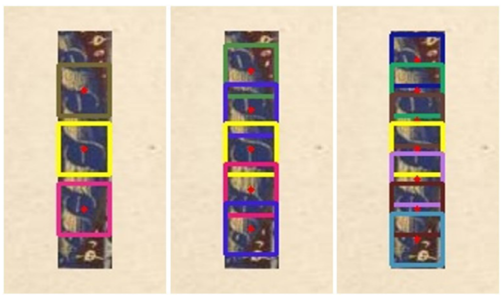

Fig. 8. Multiple receptive fields to cover rubanlettrine. 3 (left), 5 (middle) and 7 (right) embeddings are considered in the example. Each red dot is the center of a receptive field, in yellow the receptive field positioned at the middle of the query and in random colors its symmetrical neighbors. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Table 3
mAP and median AP (medAP) for *pattern spotting* task on each of the 2 × 2 groups for P_2 configuration (i.e. without T&N modification).

size	aspect ratio	freq	mAP	medAP
big	square	0.03	0.665	0.925
small	square	0.13	0.496	0.572
big	non-square	0.01	0.451	0.399
small	non-square	0.83	0.187	0.139

Our goal is to find differences between those groups and characterize best/poor performance queries. Table 3 shows mean and median AP for these groups in addition to the frequency of each of them. Two important observations can be made: (1) the dataset is heavily concentrated in small and non-square queries which turns out to be the most difficult ones to spot and yields worst performance queries, decreasing overall mAP (Table 2); and (2), aspect ratio is the key factor to have good performance rather than size. Small and square are preferable to big and non-square queries. This result makes sense because our approach is based on neurons receptive fields, which are square due to the kernel shape. Improvements dealing with non-square queries are expected to have great impact because they are a large group of the dataset.

4.3.2. Multiple embedding representation

One way to deal with non-square queries would be to consider not only the center neuron as embedding but also more neurons to cover the entire query (recall the trade-off mentioned before). To this end, we conducted some preliminary experiments, considering multiple embeddings for *image retrieval* with promising results. We considered 3, 5 and 7 embeddings per query (the embedding positioned at the middle of the query and its symmetrical 1D-neighbors) as shown in Fig. 8 for the query `rubanlettrine`. Those different patches of the query ("sub-queries") were then looked independently with the indexed page-embeddings and all "sub-query" rankings were merged and re-sorted based on the similarity measure.

Results of this procedure are shown in Table 4 where the central embedding (# embeddings = 1) mAP is presented as well. Compared to the 1-embedding baseline, the 5-embedding increases the mAP for the small and non-square queries group by 18%, which shows this is a promising solution to investigate in the future to face non-square queries. Furthermore, more embeddings do not imply better results (as with 5 and 7), and some queries have now worst average precision with multiple embeddings; we believe that an optimal number of embeddings should be considered for each query instance rather than fixing one single number for queries of any size.

Table 4
mAP for *image retrieval* task on each of the 2 × 2 groups comparison between single and multipe embeddings per query.

		# embeddings			
size	aspect ratio	1	3	5	7
big	square	0.749	0.757	0.762	0.758
small	square	0.742	0.763	0.775	0.771
big	non-square	0.660	0.674	0.686	0.682
small	non-square	0.459	0.529	0.540	0.528
	mAP	**0.505**	**0.567**	**0.577**	**0.567**

Table 5
mAP for *image retrieval* and *pattern spotting* tasks comparison between state-of-the-art system [6] and our current proposal (FPN with P_2 + T&N). We also show results on *image retrieval* task for 5-embedding configuration (Mltp).

	image retrieval	*pattern spotting*
State-of-the-art	0.580	0.157
FPN (P_2 + T&N)	0.505	0.272
FPN (P_2 + T&N + Mltp)	0.577	–

Table 6
mAP and median AP (medAP) for *image retrieval* task on each of the 2 × 2 groups comparison between state-of-the-art system [6] and our current Feature Pyramid Network approach (FPN).

		State-of-the-art		FPN	
size	aspect ratio	mAP	medAP	mAP	medAP
big	square	0.881	1.000	0.749	1.000
small	square	0.801	0.918	0.742	0.854
big	non-square	0.701	0.764	0.660	0.887
small	non-square	0.535	0.500	0.459	0.434

Finally, efficiency is now an issue because each embedding is looked for independently and as such it increases computation time for retrieval during the online phase.

4.4. Comparison with state-of-the-art

We compare our proposal with the state-of-the-art system [6]. Table 5 shows overall mAP for both tasks. Compared with state-of-the-art, our current system - the one without multiple embedding since it has not been tested on *pattern spotting* task yet - shows an improvement of 73% in *pattern spotting* but it is still below the state-of-the-art results in *image retrieval*, by almost 13%. When we use multiple embeddings this difference is reduced just to 0.5%.

On the other hand, the proposed approach uses 256D embeddings compared with the 4096D VLAD descriptors used in [6]. This reduction of the descriptor dimension by a factor of 16 along with the improvement in the spotting task show that CNN features are more discriminative than VLAD descriptors at locating patterns, even when the CNN was not trained with domain-specific data. Better localization and worst retrieval mAP can be explained by the fact that instances of the same category may appear several times on the same page.

Finally, the same 2 × 2 categorization of the queries, as presented in sub-section 4.3.1, is carried out for an in-depth comparison between the state-of-the-art system and our current approach (FPN with P_2 + T&N). Results are respectively shown in Tables 6 and 7 for *image retrieval* and *pattern spotting*. Both tables show that for the state-of-the-art system, size is the key factor for good performance as one can see in Table 7. For our system, aspect ratio is the key factor in FPN to have good AP performance because the best results are always obtained with square queries, regardless of the query size.

Table 7
mAP and median AP (medAP) for *pattern spotting* task on each of the 2 × 2 groups comparison between state-of-the-art system [6] and our current Feature Pyramid Network approach (FPN).

		State-of-the-art		FPN	
size	aspect ratio	mAP	medAP	mAP	medAP
big	square	0.546	0.385	0.681	0.949
small	square	0.102	0.044	0.546	0.639
big	non-square	0.405	0.333	0.509	0.500
small	non-square	0.149	0.101	0.214	0.161

5. Conclusion

We proposed a CNN-based system for spotting and retrieving image patterns in historical documents. The feature maps are used to extract features of local regions (patches) in the input image based on each neuron (“pixel” in the feature map) receptive field. With those feature maps we construct a feature pyramid to have a multi-scale representation of the different patches. In this way, we can extract multi-scale local features for the whole document page with a single forward pass. In order to deal with small, non-square patterns, we have designed and added to our system a specific solution based on multiple receptive fields.

Experiments carried out on the DocExplore dataset show that the proposed system improves mAP by 73% in pattern localization compared with state-of-the-art results, even if descriptor dimension used for distance computation is reduced by a factor of 16. Limitation of the approach is that for the retrieval task, it leads to a mAP which is inferior by 13% compared to state-of-the-art results. However the specific solution based on the description of the query by multiple receptive fields, allows to reduce this limitation and obtain a mAP inferior by 0.5% only when compared to the state-of-the-art for the retrieval task. This new procedure which provides promising results could also be implemented to improve localization results and is part of our ongoing work.

Last but not least, since the feature extractor of our system was not trained with domain-specific data, we believe that results in both tasks may be improved if we are able to integrate transfer learning. The lack of large historical document image datasets for object detection (with ground-truth information) limits however this strategy but looking for a solution to this limitation is also part of our future work. A good starting point would be to fine-tune our feature extractor with the DocExplore dataset, to see if significant improvements can be obtained. Since we do not want to use the same dataset for transfer learning and mAP evaluation (to avoid overfitting), we will look for new historical document datasets or even create an artificially one for this purpose.

Funding

This work was supported by CONICYT through the projects [PFCHA/MAGISTER NACIONAL/2018 - 22180111] and [STIC-Amsud 19-STIC-04]. The work is also part of the PlaIR 2.018 project (2015-2020), co-financed by the European Union with the European Regional Development Fund (ERDF) (HN0005604) and by the Normandy Region.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] A. Babenko, V. Lempitsky, Aggregating deep convolutional features for image retrieval, arXiv:1510.07493 (2015).
[2] A. Babenko, A. Slesarev, A. Chigorin, V. Lempitsky, Neural codes for image retrieval, in: ECCV, 2014, pp. 584–599.
[3] M.-M. Cheng, Z. Zhang, W.-Y. Lin, P. Torr, Bing: binarized normed gradients for objectness estimation at 300 fps, in: IEEE Conference on Computer Vision and Pattern Recognition, 2014, pp. 3286–3293.
[4] DocExplore-dataset, Pattern spotting in medieval document images, 2016,
[5] S. En, S. Nicolas, C. Petitjean, F. Jurie, L. Heutte, New public dataset for spotting patterns in medieval document images, J. Electron. Imaging 26 (1) (2016) 11010.
[6] S. En, C. Petitjean, S. Nicolas, L. Heutte, A scalable pattern spotting system for historical documents, Pattern Recognit. 54 (2016) 149–161.
[7] S. En, C. Petitjean, S. Nicolas, L. Heutte, F. Jurie, Pattern localization in historical document images via template matching, in: ICPR 2016, 2016, pp. 2054–2059.
[8] A.P. Giotis, G. Sfikas, B. Gatos, C. Nikou, A survey of document image word spotting techniques, Pattern Recognit. 68 (2017) 310–332.
[9] Y. Gong, L. Wang, R. Guo, S. Lazebnik, Multi-scale orderless pooling of deep convolutional activation features, in: ECCV, 2014, pp. 392–407.
[10] X. Han, T. Leung, Y. Jia, R. Sukthankar, A.C. Berg, Matchnet: unifying feature and metric learning for patch-based matching, in: IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 3279–3286.
[11] A.W. Harley, A. Ufkes, K.G. Derpanis, Evaluation of deep convolutional nets for document image classification and retrieval, in: ICDAR 2015, IEEE, 2015, pp. 991–995.
[12] H. Jégou, M. Douze, C. Schmid, P. Pérez, Aggregating local descriptors into a compact image representation, in: CVPR 2010, 2010, pp. 3304–3311.
[13] H. Jegou, F. Perronnin, M. Douze, J. Sánchez, P. Perez, C. Schmid, Aggregating local image descriptors into compact codes, IEEE Trans. on PAMI 34 (9) (2011) 1704–1716.
[14] Y. Kalantidis, C. Mellina, S. Osindero, Cross-dimensional weighting for aggregated deep convolutional features, in: ECCV, 2016, pp. 685–701.
[15] A. Krizhevsky, I. Sutskever, G.E. Hinton, Imagenet classification with deep convolutional neural networks, in: Advances in Neural Information Processing Systems, 2012, pp. 1097–1105.
[16] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, S. Belongie, Feature pyramid networks for object detection, in: IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 2117–2125.
[17] T.-Y. Lin, P. Goyal, R. Girshick, K. He, P. Dollár, Focal loss for dense object detection, IEEE Trans. Pattern Anal. Mach. Intell. 42 (2) (2020) 318–327.
[18] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, C.L. Zitnick, Microsoft coco: Common objects in context, in: European Conference on Computer Vision, 2014, pp. 740–755.
[19] D.G. Lowe, et al., Object recognition from local scale-invariant features., in: ICCV, 99, 1999, pp. 1150–1157.
[20] W.-Y. Ma, H.J. Zhang, Benchmarking of image features for content-based retrieval, in: Conference Record of Thirty-Second Asilomar Conference on Signals, Systems and Computers (Cat. No. 98CH36284), 1, IEEE, 1998, pp. 253–257.
[21] H. Noh, A. Araujo, J. Sim, T. Weyand, B. Han, Large-scale image retrieval with attentive deep local features, in: Proceedings of the IEEE International Conference on Computer Vision, 2017, pp. 3456–3465.
[22] J. Sánchez, V. Romero, A.H. Toselli, M. Villegas, E. Vidal, ICDAR2017 competition on handwritten text recognition on the READ dataset, in: ICDAR 2017, 2017, pp. 1383–1388.
[23] A. Sharif Razavian, J. Sullivan, A. Maki, S. Carlsson, A baseline for visual instance retrieval with deep convolutional networks, in: International Conference on Learning Representations, ICLR, 2015.
[24] I. Úbeda, J.M. Saavedra, S. Nicolas, C. Petitjean, L. Heutte, Pattern spotting in historical documents using convolutional models, 5th International Workshop on Historical Document Imaging and Processing, HIP 2019, 2019.
[25] T. Uricchio, M. Bertini, L. Seidenari, A. Bimbo, Fisher encoded convolutional bag-of-windows for efficient image retrieval and social image tagging, in: Proceedings of the IEEE International Conference on Computer Vision Workshops, 2015, pp. 9–15.
[26] K.L. Wiggers, A.S. Britto Jr, L. Heutte, A.L. Koerich, L.S. Oliveira, Image retrieval and pattern spotting using siamese neural network, in: International Joint Conference on Neural Networks, IJCNN 2019, 2019.
[27] P. Yarlagadda, A. Monroy, B. Carque, B. Ommer, Recognition and analysis of objects in medieval images, in: Asian Conference on Computer Vision, Springer, 2010, pp. 296–305.
[28] J. Yue-Hei Ng, F. Yang, L.S. Davis, Exploiting local features from deep networks for image retrieval, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Pecognition workshops, 2015, pp. 53–61.
[29] S. Zagoruyko, N. Komodakis, Learning to compare image patches via convolutional neural networks, in: CVPR, 2015, pp. 4353–4361.
[30] H. Zhao, S. Sun, Z. Jing, Local-information-based uncorrelated feature extraction, Opt. Eng. 45 (2) (2006) 20505.
[31] L. Zheng, Y. Yang, Q. Tian, Sift meets CNN: a decade survey of instance retrieval, IEEE Trans. PAMI 40 (5) (2017) 1224–1244.