Spotting Symbol over Graphical Documents Via Sparsity in Visual Vocabulary

Do Thanh Ha[1(✉)], Salvatore Tabbone[2(✉)], and Oriol Ramos Terrades[3(✉)]

[1] Faculty of Mathematics - Mechanics - Informatics,
VNU - Hanoi University of Science, Hanoi, Vietnam
hadt_tct@vnu.edu.vn
[2] Université de Lorraine - LORIA, Vandœuvre-lès-Nancy Cedex, France
tabbone@loria.fr
[3] Universitat Autònoma de Barcelona - Computer Vision Center, Bellaterra, Spain
oriolrt@cvc.uab.cat

Abstract. This paper proposes a new approach to localize symbol in the graphical documents using sparse representations of local descriptors over learning dictionary. More specifically, a training database, being local descriptors extracted from documents, is used to build the learned dictionary. Then, the candidate regions into documents are defined following the similarity property between sparse representations of local descriptors. A vector model for candidate regions and for a query symbol is constructed based on the sparsity in a visual vocabulary where the visual words are columns in the learned dictionary. The matching process is performed by comparing the similarity between vector models. The first evaluation on SESYD database demonstrates that the proposed method is promising.

Keywords: Spotting graphical symbols · Sparsity · Learned dictionary · Shape context · Interested points

1 Introduction

Among the graphics recognition community, a lot of efforts have been devoted in the last years to deal with the problem of identifying regions likely to contain a certain symbol within graphics-rich documents. One of the first approaches was the retrieval of engineering drawings based on the use of a stochastic models [13]. The main advantage of this method is that it works well even where the query symbol is embedded in, for example, is connected to other parts in the drawing. However, this performance is not good for the complex queries having several elements with spatial relationship between them.

Other techniques [2,3,9,12,17] rely on the structural information inherent in graphical symbols such as points, lines, junctions, regions etc. In that methods, graphical entities are encoded as attributed graphs and then the stage of localization symbols in documents is done using subgraph isomorphism algorithms. In general, the subgraph matching algorithms suffer from a huge computational

K.C. Santosh et al. (Eds.): RTIP2R 2016, CCIS 709, pp. 59–70, 2017.
DOI: 10.1007/978-981-10-4859-3_6

burden, although particular cases of subgraph isomorphism can be solved in polynomial time [8]. Thus, these approaches are insufficient when working with the larger collection of data. In [6,23], some indexing strategies are proposed to reduce the retrieval time and to increase the potential applications of these approaches.

Some of the methods [17,22] work with low-level pixel features on regions of interest of the documents. After ad-hoc segmentation, global pixel-based descriptors of regions [14] are computed and compared with the query symbols. A distance metric is used to decide the retrieval ranks and to check whether the retrievals are relevant or not. However, the limitation of these methods is one-to-one feature matching and they only work for a limited set of symbols.

Like the methods based on low-level pixel features, the methods as in [18,24] also works with ad-hoc segmentation. However, these methods compute the vectorial signatures instead of pixel feature. The disadvantage of these method is that they do not work well in the real-world applications since symbols are effected by noisy images. In addition, the assumptions the symbol always fall into interest region can compute the vectorial signature inside those regions are other limitation of these methods.

In this paper we propose a new two-stage method for symbol localization in graphical documents. In the first stage, the training database, being the local descriptors computed on interest points of documents, is used to learn the visual dictionary. In the second stage, we define the *similarity* property between two descriptors to localize some candidate regions over documents. In addition, to keep only the candidate regions where the query symbol actually is, we propose to use the visual vocabulary to construct the vector model of region. Then, the regions contains the request symbols over documents are found out by comparing vector models.

The organize of this paper as follow: Sects. 2 and 3 describe how to calculate the local descriptor adapted to the graphical document, and how to learn a visual vocabulary from the training set being the local descriptors. The details of the symbol localization process is addressed in Sect. 4. The first evaluations of proposed approach is dedicated in Sect. 5. Finally, we conclude and discuss the future work in Sect. 6.

2 Local Descriptor for Document

Like the shape context, the shape context of interest points (SCIP) [15] also presents the relationship between points of object, but instead of the relationship between contour points, it describes the relationship between the key-points and the contour points, which not only reduces the size of the descriptor but also remains the invariance to scaling and rotation thanks to the information about the dominant orientation of interest points. In addition, the local descriptor as SCIP and the learned dictionary are used to increase the performance of recognition system [5].

This paper also focus on the use of sparse representation over learned dictionary for spotting symbols in graphical documents. When working on the whole

document, the symbols have not been segmented, and using interest points, the contour points being far from them provide less useful information to discriminate objects. Therefore, the SCIP cannot be applied at document level. Instead we define the neighborhood region for each reference interest point as in [4,14]. This region needs to ensure the invariance of SCIP computed inside it, thus it cannot be fix a prior. This difficulty is overcome by using the scale on which the interest point detected. More details, with each interest point, the neighborhood region associated with it is an ellipse that is defined with the centre at this point, and the semi-major axis, the semi-minor axis are decided depending on the scale in which this interest point is detected.

This extension of SCIP descriptor for a document level is called ESCIP descriptor from now on. In fact, ESCIP for the neighborhood region corresponding to one interest point in the document is the SCIPs calculated on this neighborhood region.

3 Visual Vocabulary of ESCIP

Visual vocabulary of ESCIP is the learned dictionary in which visual words are columns in this dictionary. This section describes how to build the learned dictionary of ESCIP descriptors and illustrates how one signal is presented over this dictionary. In general, the learned dictionary of ESCIP is the dictionary constructed from the training dataset $\mathcal{H} = \{\mathcal{H}_1, ..., \mathcal{H}_n\}$ being the ESCIP descriptors extracted from n documents. By applying one of the learned algorithms, we learn the dictionary $A \in \mathbb{R}^{L \times M}$ satisfying that each ESCIP descriptor $h_j \in \mathcal{H}$ in training dataset has an optimally sparse approximation $\bar{x}_j$ in this dictionary satisfying $\|A\bar{x}_j - h_j\|_2 \leq \epsilon$ or finding:

$$\min_{A,x_j} \sum_j \|x_j\|_0 \text{ subject to } \|h_j - Ax_j\|_2 \leq \epsilon, \text{ for all } h_j \in \mathcal{H} \tag{1}$$

This dictionary can be obtained by the learning process. This process iteratively adjusts A via two main stages: *sparse coding stage* and *update dictionary stage*. In the sparse coding stage, all sparse representation x_j of $h_j \in \mathcal{H}$ are found by solving Eq. (2) on the condition that A is fixed.

$$\min_{x_j} \|x_j\|_0 \text{ subject to } \|Ax_j - h_j\|_2 \leq \epsilon \text{ for all } h_j \in \mathcal{H} \tag{2}$$

The Eq. (2) can be solved by the greedy techniques or relaxation one. By comparing greedy algorithms, we notice that orthogonal matching pursuit (OMP) algorithm [16] does not provide a better approximation to the solution, but its computing cost time is lower. Moreover, the OMP can be used to find the approximate solutions instead of exact ones by changing its stopping rule as accumulating nonzero elements in the solution vector until the reconstruction error is less than ϵ. Therefore, we decide to use OMP algorithm in this paper.

In the update dictionary stage, an updating rule is applied to optimize the sparse representation $\mathcal{X} = \{x_j\}$ of all $h_j \in \mathcal{H}$. To the best of our knowledge, there

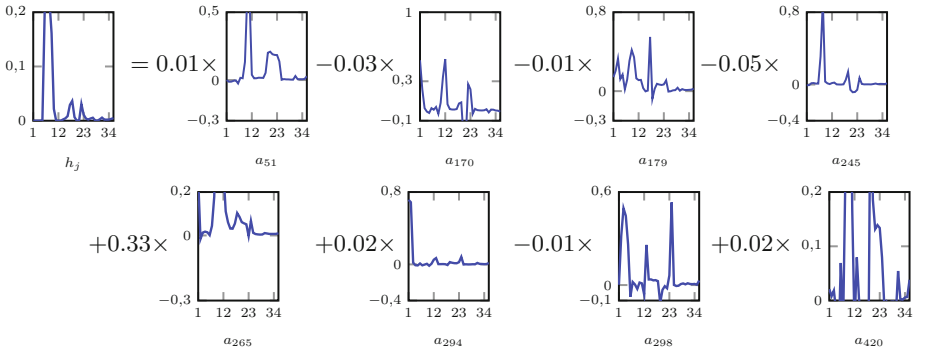


Fig. 1. The presentation of ESCIP descriptor as linear combination of columns in the dictionary A

are 4 well-known learning algorithms named K-SVD [1], MOD [7], ODL [11], and RLS-DLA [20]. The way of updating the dictionary is different from each learning algorithm to others. For example, the K-SVD algorithm makes a modification in the dictionary's columns; the MOD algorithm makes the mean of the set of norm residuals as small as possible. In this paper we use K-SVD algorithm as suggested in [16].

In the K-SVD algorithm, each column a_{j_0} of A is updated sequentially such that the residual error defined in (3) is minimized, where $\mathcal{X}$ and other columns of A are fixed:

$$\|\mathcal{H} - A\mathcal{X}\|_F^2 = \|(\mathcal{H} - \sum_{j \neq j_0} a_j x_j^T) - a_{j_0} x_{j_0}^T\|_F^2 \tag{3}$$

In Eq. (3), the value $(\mathcal{H} - \sum_{j \neq j_0} a_j x_j^T)$ is fixed, therefore the minimum error $\|\mathcal{H} - A\mathcal{X}\|_F^2$ depends only on the optimal a_{j_0} and $x_{j_0}^T$. These optimal solutions $\bar{a}_{j_0}$ and $\bar{x}_{j_0}^T$ can be given by calculating SVD (*Singular Value Decomposition*) over the error matrix defined only on relevant samples. More details about the K-SVD algorithm can be found in [1].

The output of K-SVD algorithm is all optimal sparse representation $\bar{x}_j$ of $h_j \in \mathcal{H}$ and the learned dictionary A, it means each local descriptor h_j can be expressed as a sparse linear combination of the columns in $A = \{a_1, ..., a_M\} \in \mathbb{R}^{L \times M}$, and therefore the sparse vector $\bar{x}_j$ is the new representation for h_j. Figure 1 illustrates the presentation of one local descriptor h_j as linear combination of 8 columns in the dictionary A being $a_{51}, a_{170}, a_{179}, a_{245}, a_{265}, a_{294}, a_{298}, a_{420}$.

4 Spotting Symbols in Graphical Documents

4.1 Document Indexing

Generally, the processes of searching and matching local descriptors computed from interest points usually waste the computing time and the memory. Therefore,

some techniques are proposed to overcome this difficulty such as clustering similar descriptors to define visual words being the centroids of clusters. However, the performance of these methods depends directly on the applied clustering algorithm and the characteristic of data. Very recently, Do *et al.* [5] proposed an approach that uses sparse representations of local descriptors. The performance of this approach is good and promising for symbol recognition. However, to apply this method on document level, beside of finding candidate regions that are considered as the segment symbols (in Sect. 4.2), we also need an effectual way to index the content that helps to match candidate regions in each document.

To index the content, an inverted file structure is built based on the learning dictionary of local descriptors. Particularly, the sparse representation of each local descriptor over A gives information about columns of learning dictionary A used to describe this. If we consider each column of A as one visual word then A becomes the visual dictionary and therefore the group of visual words used to describe this descriptor is known. For example, without loss of generality, let h_i^s being the ESCIP number i-*th* in the document D_s and $\bar{x}_i^s = \{\alpha_1^s, 0, ..., 0, \alpha_k^s, 0, ..., 0, \alpha_t^s, 0, ..., 0, \alpha_l^s, 0, ..., 0\}$ being the sparse representation of h_i^s over A, then h_i^s can be expressed as following:

$$h_i^s = \alpha_1^s \times a_1 + \alpha_k^s \times a_k + \alpha_1^t \times a_t + \alpha_1^l \times a_l \tag{4}$$

Therefore, h_i^s is assigned to the group of visual words $W_i^s = \{a_1, a_k, a_t, a_l\}$ and coefficients $\Delta_i^s = \{\alpha_1^s, \alpha_k^s, \alpha_t^s, \alpha_l^s\}$.

Once the document is described by visual words over visual vocabulary A, we construct an inverted file including two elements: the visual vocabulary and the occurrences. The visual vocabulary is A, and for each visual word in A we store: (1) a list of interest points that its corresponding ESCIP has this word in their sparse representation over dictionary A, (2) the corresponding documents, (3) the group of visual words as well as the coefficients in the representation of these ESCIP (see Fig. 2).

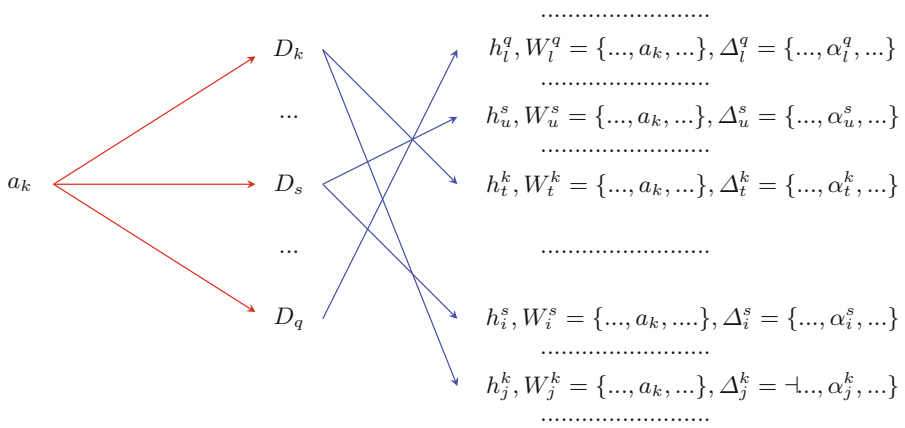


Fig. 2. The inverted file structure

4.2 Location Symbols in Graphical Documents

This section describes our main contribution that includes two steps. The first step is to define candidate regions on documents based on the property of interest points and the similarity in sparse representations of their corresponding local descriptors. In the second step, each candidate region is transformed to the vector by using weight visual words.

Interest Regions over Documents. The interest regions are defined from the interest points of query symbol and the interest points of documents. More specifically, given $q = \{x_q, y_q, \delta_q, \theta_q\}$ and $p = \{x_p, y_p, \delta_p, \theta_p\}$ are two interest points, one from query symbol and one from the document, respectively. The center coordinates (x_r, y_r) of the interest region are defined by the affine transform of the coordinates of q:

$$\begin{pmatrix} x_r \\ y_r \end{pmatrix} = \frac{\delta_p}{\delta_q} G_{\theta_p - \theta_q} \begin{pmatrix} x_c - x_q \\ y_c - y_q \end{pmatrix} + \begin{pmatrix} x_p \\ y_p \end{pmatrix} \tag{5}$$

where $G_{\theta_p - \theta_q}$ is the rotation matrix, (x_c, y_c) is the centre coordinates of the query symbol, and the width w_r, the height h_r and the orientation θ_r of the region are given by:

$$h_r = h \times \frac{\delta_p}{\delta_q}; \quad w_r = w \times \frac{\delta_p}{\delta_q}; \quad \theta_r = \theta_q - \theta_p \tag{6}$$

where h and w are the height and the width of the query symbol. Figure 3 presents an example of how to locate an interested region over the document.

For a particular symbol query, the number of the interest points like p, q is large, therefore the possible regions of interest constructed from two equations

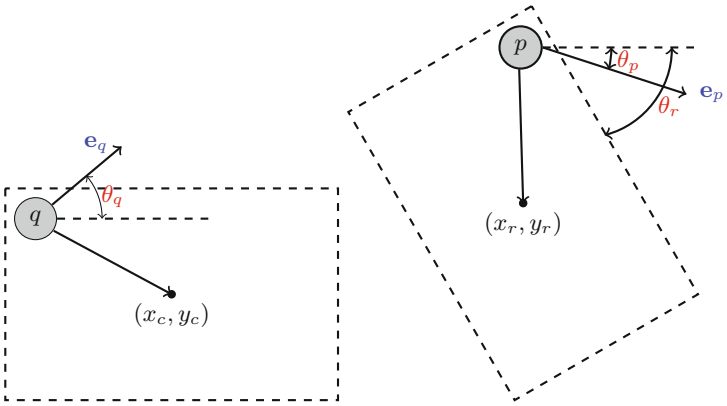


Fig. 3. Example about how to locate an interest region in the document (right) being corresponding to the request symbol (left)

above are large. To select only those where the query symbol may appear we introduce the notion of *similarity* property between pairs of interest points at level c and we select only those regions satisfying this property. We recall that W_p is the set of visual words describing the descriptor h_p computed on p.

Definition 1 (Similarity Property). *Two interest points p and q holds the similarity property at level $c \in (0, 1]$ if the following inequality is satisfied:*

$$c \times |W_q| \leq |W_p \cap W_q| \leq |W_q| \tag{7}$$

Intuitively, we use the similarity property to compare interested points p and q in terms of the visual words used to describe descriptors h_p and h_q. In fact, the value of $c \in (0, 1]$ controls the overlapping degree of W_p and W_q. Moreover, by setting $c = 1$ we force all the visual words used in the representation of h_q appear in the representation of h_p.

Vector Construction for Candidate Region. The *similarity* property permits us to reject regions of document where we can ensure that the query symbol is not found with high confidence degree. However, using only this similarity measure, we will retrieve many false positive instances. Thus, to keep the regions of interest where the query symbol actually is, we propose to construct the vector model for each candidate region and then compare it to the vector model of query symbol.

For each interest point p in $\mathcal{R}$, its descriptor $h_p^{\mathcal{R}}$ is the shape context of interest point calculated in regions $\mathcal{R}$. The optimal sparse representation $\bar{x}_p^{\mathcal{R}}$ of $h_p^{\mathcal{R}}$ is the solution of the Eq. (8) where A is learned dictionary build for training dataset being ESCIP descriptors over graphical documents.

$$\bar{x}_p^{\mathcal{R}} = \min_{x_p^{\mathcal{R}}} \|x_p^{\mathcal{R}}\|_0 \text{ subject to } \|Ax_p^{\mathcal{R}} - h_p^{\mathcal{R}}\|_2 \leq \epsilon \tag{8}$$

The columns of the learned dictionary A play the role of words in a visual word framework and the coefficients play the role of the degree of confidence for visual words. With the purpose of keeping information not only on what visual words in the dictionary are used, but also on the coefficients in the sparse representation, we use the optimal sparse representation $\bar{x}_p^{\mathcal{R}}$ of $h_p^{\mathcal{R}}$ as its characteristic vector and compute the tf and idf factors to build the vector model associated to the candidate region $\mathcal{R}$. On the one hand, we compute the k-th word frequency $tf_k^{\mathcal{R}}$ as:

$$tf_k^{\mathcal{R}} = \frac{\sum_{p \in \mathcal{I}} \bar{x}_p^{\mathcal{R}}(k)}{\sum_{j=1}^{K} \sum_{p \in \mathcal{I}} \bar{x}_p^{\mathcal{R}}(j)} \tag{9}$$

where $\mathcal{I}$ is the set of interest points in $\mathcal{R}$.

On the other hand, the idf factor indicates the importance of the word k for the discrimination between regions. To compute this value, the number of instances of a word k in the whole document have to be computed. However,

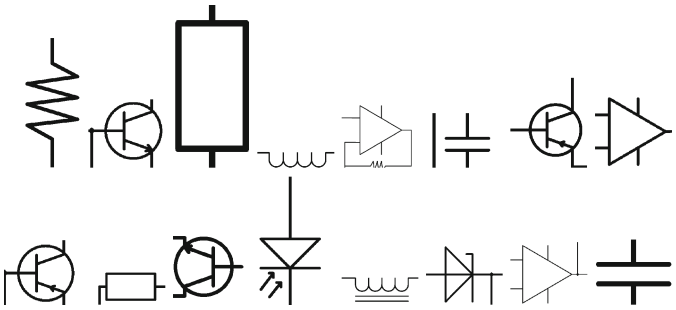


Fig. 4. Some isolated segmented symbols in the reference database.

because of some candidate regions still are false positive instances, thus computing this value on all candidate regions will reduce the precision in discriminating between regions. To overcome this problem, we propose to compute the *idf* factor from an alternative dataset composed of samples of segmented symbols. Figure 4 presents some symbols in the reference database including 1859 segmented symbols. More details, *idf* is calculated as in Algorithm 1:

Algorithm 1. Calculate *idf* factor

1: For each symbol in a reference database: (1a.) Calculate the ESCIP descriptors of this symbol
(1b.) Calculate all sparse representations of descriptors over the learning dictionary A using OMP algorithm
(1c.) The sparse representations give the information of what visual word is used to describe this symbol
2: Let l_k be the number of symbols in which the word k appears and N is the number of symbols in reference database, then $idf_k = \log(\frac{N}{1+l_k})$

Therefore, the vector model of candidate region $\mathcal{R}$ is defined as following:

$$v_{\mathcal{R},k} = tf_k^{\mathcal{R}} \times idf_k \tag{10}$$

4.3 Matching Process

For each query symbol, its vector model v_q is calculated as the same way we calculate the vector model of candidate regions $\mathcal{R}$. Next, the vector model of the query symbol and the vector model of candidate regions, $v_{\mathcal{R}}$ is compared using the cosine distance:

$$\text{distance}(v_q, v_{\mathcal{R}}) = \frac{\langle v_q, v_{\mathcal{R}} \rangle}{|v_q| \times |v_{\mathcal{R}}|} \tag{11}$$

Finally, candidate regions with low cosine distance are discarded and the others are ranked in descend order as being the regions containing the requested symbol.

5 Experiment

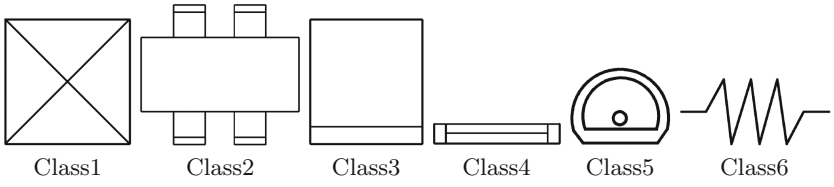
The goal of the experiments carried out in this paper is to evaluate if the performance of symbol spotting method will be improved when sparse representations are used. The preliminary experiment is performed on subset of the SESYD dataset[1] which is a collection of 15 images and 6 different classes as queries are tested (see Fig. 1). This subset is also used in [4] however in this paper beside of verifying the precision of proposed method, we will also present the computing time for spotting symbols.

The training database in the learned dictionary algorithm is local descriptors ESCIP computed on graphical documents. To provide more weight to the region close to the detected interest points in the direction of the interest point orientation and to increase in such way the discrimination capacity of the local descriptor, ESCIP is computed over the ellipse that is defined using information of orientation and scale of interest point. Particularly, if the scale of the interest point is σ, then the value of the semi-minor and the semi-major axes are $\frac{3}{2}\sigma$, 3σ (set by the experience), respectively. The visual vocabulary A is built using the K-SVD algorithm with the number of columns in A to 512, the number of iterations to 350, and the approximation error to $\epsilon = 0.4$.

In fact, there are numerous works have been proposed to deal with the problem of spotting symbols in the graphical documents [10,19,21]. However, to the best of our knowledge there is no complete evaluation for the existing approaches on the same database.

Thus, we decide to compare proposed approach to the method of Nguyen *et al.* [14] since this method is also based on a local descriptors and use inverted files for document indexing. The main difference between them are: firstly, in the proposed approach sparsity technique is used to build a visual vocabulary, while in [14] a visual vocabulary is the set of centroids of clusters obtained using k-means algorithm. Secondly, in our approach, we first define candidate regions based on the number of shared visual words and then we build and comparing

Table 1. The query classes



[1] http://mathieu.delalandre.free.fr/projects/sesyd/index.html.

vector models of candidate regions to filter out some false positive instance and therefore improve the precision rate.

Both methods are compared using the widespread precision and recall measures for retrieval tasks. The precision measure is the ratio between the number of relevant retrieved items and the number of retrieved items. On the one hand, precision rate equals to 1 means that all retrieved examples correspond to the queried symbol. That is, there is not false positives samples in the retrieved

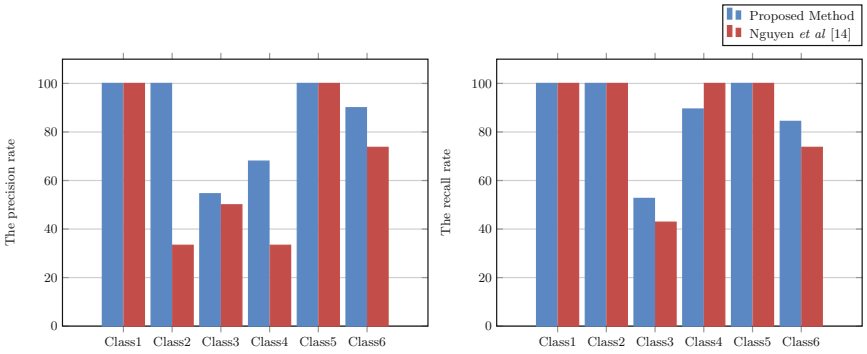


Fig. 5. Spotting results for 6 classes of query in the Table 1

Table 2. The computing time (seconds) that corresponds to each query.

Documents	Classes					
	1	2	3	4	5	6
Document 1	8.6534	11.4800	12.8461	2.6261	21.4594	4.9100
Document 2	8.6961	11.3504	12.8563	2.5374	21.3855	4.8034
Document 3	8.6377	11.4096	12.8259	2.6336	20.7571	4.8560
Document 4	8.7181	11.3959	12.8552	2.6078	20.9517	4.7784
Document 5	8.6534	11.4260	12.8515	2.5890	20.7913	4.8589
Document 6	8.7157	11.3998	12.8658	2.6142	20.9933	4.8065
Document 7	8.6566	11.3706	12.8402	2.6295	20.9692	4.8377
Document 8	8.7488	11.4102	12.8510	2.6098	21.0592	4.7998
Document 9	8.6646	11.3915	12.8696	2.6343	20.9809	4.8657
Document 10	8.7111	11.3695	12.8404	2.5645	20.9928	4.8055
Document 11	8.6586	11.4164	12.8500	2.6580	21.0074	4.8675
Document 12	8.7442	11.3737	12.8369	2.6196	21.2620	4.8178
Document 13	8.6554	11.4137	12.8398	2.6101	20.8022	4.8580
Document 14	8.7265	11.3629	12.8730	2.6024	21.0608	4.8028
Document 15	8.6317	11.4065	12.8353	2.6136	20.5692	4.8669
Average	8.6848	11.3984	12.8558	2.610	20.9361	4.8357

documents and location. Conversely, a lower precision rate, a higher non-relevant (false positive) items included in the results are. On the other hand, the recall rate is the number of relevant items in the collection. It measures the effectiveness of the system in retrieving the relevant items, and it equals 1 in case all the items considered as retrievals are relevant. Indeed, a low recall rate means that relevant items have been missed.

In the Fig. 5 we see that precision and recall rates increase in most cases. Table 2 presents the computing time that corresponds to each query.

6 Conclusion

This paper presents a new approach for symbol spotting systems that uses the visual dictionary being the dictionary constructed from local descriptors. By using learning techniques, the obtained visual vocabulary can be adapted better to the intrinsic properties of the documents datasets. In addition, the proposed approach improves the computing time in the retrieving process by combining sparsity with indexing techniques and therefore only regions in which the queried symbol may appear is considered in the matching phase. First experiments on a subset of benchmark dataset for a symbol spotting application are promising. In the future, we would like to examine the robustness and scalability of this method on other datasets.

References

1. Aharon, M., Elad, M., Bruckstein, A.: K-SVD: an algorithm for designing overcomplete dictionaries for sparse representation. Sig. Process. **54**(11), 4311–4322 (2006)
2. Barbu, E., Héroux, P., Adam, S., Trupin, É.: Using bags of symbols for automatic indexing of graphical document image databases. In: Liu, W., Lladós, J. (eds.) GREC 2005. LNCS, vol. 3926, pp. 195–205. Springer, Heidelberg (2006). doi:10.1007/11767978_18
3. Bodic, P.L., Heroux, P., Adam, S., Lecourtier, Y.: An interger linear program for substitution-tolerant subgraph isomorphism and its use for symbol spotting in technical drawings. Pattern Recogn. **45**(12), 4214–4224 (2012)
4. Do, T.H., Tabbone, S., Terrades, O.R.: Spotting symbol using sparsity over learned dictionary of local descriptors. In: Proceeding of the International Conference on Document Analysis and Recognition, pp. 156–160 (2014)
5. Do, T.H., Tabbone, S., Terrades, O.R.: Sparse representation over learned dictionary for symbol recognition. Sig. Process. **125**, 36–47 (2016)
6. Dutta, A., Llados, J., Pal, U.: A symbol spotting approach in graphical documents by hasing serialized graphs. Pattern Recogn. **43**(3), 752–768 (2013)
7. Engan, K., Aase, S.O., Husoy, J.H.: Frame based signal compression using method of optimal directions (MOD). Proc. Int. Conf. Acoust. Speech Sig. Process. **4**, 1–4 (1999)
8. Eppstein, D.: Subgraph isomorphism in planar graphs and related problems. Graph Algorithms Appl. **3**(3), 1–27 (1999)

9. Llados, J., Marti, E., Villanueva, J.: Symbol recognition by error-tolerant subgraph matching between region adjacency graphs. Pattern Anal. Mach. Intell. **23**(10), 1137–1143 (2001)
10. Lladós, J., Valveny, E., Sánchez, G., Martí, E.: Symbol recognition: current advances and perspectives. In: Blostein, D., Kwon, Y.-B. (eds.) GREC 2001. LNCS, vol. 2390, pp. 104–128. Springer, Heidelberg (2002). doi:10.1007/3-540-45868-9_9
11. Marial, J., Bach, F., Ponce, J., Sapiro, G.: Online dictionary learning for sparse coding. In: Proceeding of the 26th Annual International Conference on Machine Learning, pp. 689–696 (2009)
12. Messmer, B.T., Bunke, H.: Automatic learning and recognition of graphical symbols in engineering drawings. In: Kasturi, R., Tombre, K. (eds.) GREC 1995. LNCS, vol. 1072, pp. 123–134. Springer, Heidelberg (1996). doi:10.1007/3-540-61226-2_11
13. Muller, S., Rigoll, G.: Engineering drawing database retrieval using statistical pattern spotting techniques. In: The Proceeding of GREC 09 Selected Papers from the Thrid International Workshop on Graphics Recognition, Recent Advances, pp. 246–255 (1999)
14. Nguyen, T.-O., Tabbone, S., Boucher, A.: A symbol spotting approach based on the vector model and a visual vocabulary. In: Proceeding of the 10th International Conference on Document Analysis and Recognition, pp. 708–712 (2009)
15. Nguyen, T.-O., Tabbone, S., Terrades, O.R.: Symbol descriptor based on shape context and vector model of information retrieval. In: Proceeding of the 8th the International Workshop on Document Analysis System (2008)
16. Pati, Y., Rezaiifar, R., Krishnaprasad, P.: Orthogonal matching pursuit: recursive function approximation with applications to wavelet decomposition. In: Proceeding of the 27th Annual Asilomar Conference on Signals, Systems, and Computers, pp. 40–44 (1993)
17. Qureshi, R., Ramel, J.-Y., Barret, D., Cardot, H.: Spotting symbols in line drawing images using graph representations. Graph. Recogn. Recent Adv. New Opportunities **5046**, 91–103 (2008)
18. Rusiñol, M., Lladós, J.: Symbol spotting in technical drawings using vectorial signatures. In: Liu, W., Lladós, J. (eds.) GREC 2005. LNCS, vol. 3926, pp. 35–46. Springer, Heidelberg (2006). doi:10.1007/11767978_4
19. Santosh, K.C., Wendling, L.: Graphical symbol recognition. Wiley Encyclopedia of Electrical and Electronics Engineering (2015)
20. Skretting, K., Engan, K.: Recursive least squares dictionary learning algorithm. Sig. Process. **58**(4), 2121–2130 (2010)
21. Tabbone, S., Terrades, O.R.: An overview of symbol recognition. In: Handbook of Document Image Processing and Recognition, pp. 523–551 (2014)
22. Tabbone, S., Wendling, L., Tombre, K.: Matching of graphical symbols in line-drawing images using angular signature information. Int. J. Doc. Anal. Recogn. **6**(2), 115–125 (2003)
23. Tabbone, S., Zuwala, D.: An indexing method for graphical documents. In: Proceeding of the 9th International Conference on Document Analysis and Recognition, pp. 789–793 (2007)
24. Zhang, W., Wenying, L.: A new vectorial signature for quick symbol indexing, filtering and recognition. In: Proceeding of the 9th International Conference on Document Analysis and Recognition, vol. 1, pp. 536–540 (2007)