Learning domain-specific feature descriptors for document images

Kandan Ramakrishnan
Dept. of Computer Science and Engineering
University of Minnesota
Minneapolis, MN 55455
rama0146@umn.edu

Evgeniy Bart
Intelligent Systems Lab
Palo Alto Research Center
Palo Alto, CA 94304
bart@parc.com

***Abstract*—Many machine learning algorithms rely on feature descriptors to access information about image appearance. Using an appropriate descriptor is therefore crucial for the algorithm to succeed. Although domain- and task-specific feature descriptors may result in excellent performance, they currently have to be hand-crafted, a difficult and time-consuming process. In contrast, general-purpose descriptors (such as SIFT) are easy to apply and have proved successful for a variety of tasks, including classification, segmentation, and clustering. Unfortunately, most general-purpose feature descriptors are targeted at natural images and may perform poorly in document analysis tasks. In this paper, we propose a method for automatically learning feature descriptors tuned to a given image domain. The method works by first extracting the independent components of the images, and then building a descriptor by pooling these components over multiple overlapping regions. We test the proposed method on several document analysis tasks and several datasets, and show that it outperforms existing general-purpose feature descriptors.**

***Keywords*-Feature descriptors, feature learning, classification**

I. Introduction

Analyzing document appearance is critical for tasks such as classification and segmentation. Many machine learning algorithms access this appearance information via *feature descriptors*. A familiar example of such a descriptor is SIFT [1]. Since the descriptor is often the algorithm's main point of access to the appearance information, using an appropriate descriptor is crucial for the algorithm to succeed.

Although hand-crafting feature descriptors can achieve excellent results, it is often difficult and time-consuming. In contrast, general-purpose descriptors (such as SIFT) are easy to apply and have proved succesful in a variety of tasks. Unfortunately, most general-purpose feature descriptors (including SIFT, SURF, HoG, and others [1], [2], [3], [4], [5]) are targeted at natural images and may perform poorly in document analysis tasks. In this paper, we propose a method for automatically learning feature descriptors tuned to a given type of images. The method is called Independent Component Feature Transform, or ICFT. It works by extracting the independent components of the images and pooling these components over multiple overlapping regions of the image. The overall architecture is inspired by the biological visual processing hierarchy, as well as by existing feature descriptors. By exploiting example images from the target domain, the descriptor becomes tuned to a particular image type. As a result, it outperforms existing general-purpose descriptors. Note that only example images, but not annotations, are necessary for this tuning. Domain-specific descriptors can thus be learned in a completely unsupervised manner.

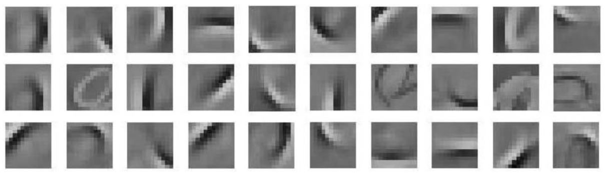
Figure 1. Filter dictionary obtained from the MNIST dataset. Note that the filters are quite different from the traditional edge detectors.

The remainder of this paper is organized as follows. In section II, we survey the relevant previous work. In section III, we describe the proposed feature learning method. The experimental evaluation of this method is presented in section IV, followed by conclusions in section V.

II. Related Work

Perhaps the simplest way to represent the appearance of an image patch is to use the raw pixel values. Although this method may be useful in some cases [6], raw pixel values are not robust to image distortions, such as translation or scaling. In addition, this representation may be highly redundant, especially for large patches. In many applications, descriptors that are more succinct and that have some degree of invariance are desirable.

Various versions of edge and blob detectors (such as Gabor filters) have been used as descriptors [7]. Some of the most popular descriptors today use histograms of responses of these detectors [1], [2], [3], [4], [5]. For example, SIFT computes gradient magnitudes in a region, then pools and histograms them over small sub-regions. This architecture is desirable because filtering the image with edge detection provides sparsity, and accumulating filter responses over small image areas makes the descriptors more invariant under small transformations. In most cases, the sub-regions are simply defined by a regular grid, although adaptive partitioning is also possible [8]. A similar architecture was used in [9], where chain code directions are pooled rather than gradients.

A drawback common to the descriptors presented above is that they were manually designed with a particular kind of images in mind. As a result, their performance degrades when they are applied to images from different or unexpected domains (see section IV).

Methods that allow learning domain-specific feature descriptors are more flexible, because they can be adapted to the stimuli at hand. Supervised methods for learning descriptors have long been used with neural networks (notably, convolutional networks such as LeNet) and related models [6], [10]. The main drawback of these methods is that they require large amounts of labeled training data. In addition, they must be trained for a specific task and thus cannot be trained ahead of time, before the task is determined. In contrast, the method proposed here does not require labeled data to learn feature descriptors, and may be pre-trained.

Unsupervised feature learning methods do not require labeled training examples; as a result, it is much easier to apply them to new domains, since no data needs to be annotated. In addition, they can be trained before a specific task is determined for the new domain. Most of the existing unsupervised methods learn descriptors that optimize reconstruction quality of the original image while encouraging sparsity [11], [12], although methods that don't consider reconstruction quality explicitly have also been used [13]. The proposed ICFT method also belongs to the general category of unsupervised methods. Compared to the approaches described above, the ICFT has a different objective function: it optimizes conditional independence of feature activations, rather than sparseness. This allows the method to deal with non-sparse activations. Similar ideas have been used successfully in a variety of applications [14]. An experimental evaluation of ICFT is presented in section IV.

III. Independent Component Feature Transform

In section III-A, we describe the inspiration for the proposed method. In section III-B, a detailed description of the method itself is given.

A. *Inspiration*

The proposed method is inspired by a widely used model of the animal visual system. This model involves a hierarchy of processing stages. The first stage in the hierarchy computes responses of a dictionary of filters to the input image. These filters (sometimes also called 'simple cells') are often sensitive to edges of different orientations and scales, as well as blobs of varying sizes. The second stage involves pooling filter responses at neighboring image locations by computing the maximum activation within a region. The units that perform this pooling are also called 'complex cells'. These two stages may be stacked several times [15], [16], [13]. As discussed in section II, applying the initial filter dictionary reduces redundancy, while subsequent pooling enables robustness to distortions. Some of the most successful feature descriptors used in computer vision follow a similar architecture [1], [3], [4], [5].

In normal animals, the initial filter dictionaty was observed to consist of edge and blob detectors [17], [18], [19]. However, it is known that visual experience may affect this dictionary [20]. It is therefore of interest to explore the criteria which could lead the visual system to favor one set of filters over another. A variety of possible criteria have beed suggested, including sparsity [18], [21], information maximization [22], and statistical independence [19]. The latter ICA-based approach is particularly interesting, because ICA has some appealing properties: it estimates filters which are as statistically independent as possible and thus may uncover the underlying processes by which the signal was generated [14].

B. *Implementing ICFT*

The ICFT feature descriptor proposed here uses an architecture similar to that described above. The computation of the descriptor involves two stages. In the first stage, a dictionary of filters is applied to the input image at all locations. In the second stage, the responses of these filters are pooled over neighboring locations to produce the final descriptor. This computation process relies on having a basic dictionary of filters. This filter dictionary is learned using ICA on unlabeled example images from the target domain. These two processes (filter dictionary learning and descriptor computation) are detailed next.

1) Learning target domain filters: To learn the filter dictionary, a set of image patches is extracted from the example images of the target domain. In our experiments, patches of size 10×10 pixels were found to work well, but this patch size can easily be adjusted as needed. Depending on the experiment, between 20,000 and 50,000 patches are extracted at random from the example images. ICA is then applied to find the independent components of these patches. We used the FastICA algorithm [14]. Typically, $N = 20$ to $N = 50$ independent components were used. These independent components are filters of size 10×10 pixels, and they are used as the filter dictionary in ICFT. This process is illustrated in Figure 2 (top panel) for the case of $N = 30$ filters.

2) Computing the ICFT descriptor: A typical ICFT descriptor is computed for image patches of size 20×20 pixels. As mentioned above, this computation involves two stages. First, the responses of the N filters in the filter dictionary (found in the previous stage) are computed at all points within the 20×20 image patch. Next, these responses are pooled over neighboring pixels to form the final descriptor.

```
FILTER DICTIONARY LEARNING
Input: a set of example images from the target domain
Output: a set of $N = 30$ filters of size $10 \times 10$ pixels
  1) Select a set of points on a grid in every training image
  2) Extract a $10 \times 10$ image patch around each point
  3) Run FastICA on the resulting set of patches and extract $N = 30$ independent components

COMPUTING 270-ICFT
Input: an image patch of size $20 \times 20$ pixels
Output: a 270-dimensional ICFT descriptor
  1) At each pixel in the input patch, find the response of each of the 30 filters (learned in the previous stage) by computing the dot product of the image and the filter
  2) Divide the $20 \times 20$ patch into nine overlapping $10 \times 10$ regions. For example, the first region spans rows 1–10 and columns 1–10; the second region spans rows 1–10 and columns 6–15; etc.
  3) For each of the nine regions and each of the 30 filters, compute the maximal activation of the filter in the region
  4) Combine the resulting $9 \cdot 30 = 270$ numbers into a 270-dimensional vector
```

Figure 2. Computing ICFT. For details, see section III.

This pooling is performed as follows. The 20×20 patch is divided into nine overlapping regions of size 10×10 pixels each, spaced equally within the 20×20 patch. The outputs of each filter are pooled over each region. The pooling operation for a given filter involves computing the maximal value over that filter's responses in a region. (We have also tried computing the average and root mean square responses, but these performed slightly poorer.) Thus, we obtain nine number for each filter, with each number representing the result of the pooling operation over one of the regions. These numbers are then combined into a $9 \cdot N$-dimensional descriptor. This descriptor computation process is summarized in Figure 2 (bottom panel). Alternatively, in some experiments four non-overlapping 10×10 pixel regions were used to obtain a $4 \cdot N$-dimensional descriptor.

IV. EXPERIMENTS

In this section we illustrate the performance of ICFT on two document analysis tasks.

A. MNIST classification

The MNIST database of handwritten digits [6] was used in this experiment. Thirty independent components were extracted from the MNIST training set. These components are shown in Figure I. As can be seen, the components are quite different from those typically obtained for natural images [18], [19], and are more complex that gradient filters used in SIFT.

Two descriptor types were evaluated: the 120-dimensional descriptor obtained by pooling over four non-overlapping regions within each patch, and the 270-dimensional descriptor obtained by pooling over nine overlapping regions within each patch.

A total of three experiments were performed. In the first experiment, the entire MNIST character image was used as input to the descriptor computation. A single descriptor for each image was thus computed and used for classification. In two additional experiments, the images were split into four or nine non-overlapping regions (on a 2×2 or 3×3 grid, respectively), and characterized by computing a descriptor for each of these regions. These descriptors were then concatenated to form the final feature set. Note that this splitting is in addition to the multiple regions used in computing each individual descriptor.

An SVM with a Gaussian kernel was used to classify digits based on these feature sets. (We have also tried linear SVM, with similar results.) The results are reported in Table I. SIFT descriptors were computed in a similar manner and used for comparison. As can be seen, the proposed 120-dimensional ICFT descriptor outperformed SIFT in all experiments. Note that SIFT is a 128-dimensional descriptor. The 270-dimensional ICFT further improved performance compared to SIFT.

Two conclusions can be made from this experiment. First, using domain-specific filters improves performance compared to using generic filters (such as those used in SIFT). This conclusion holds even when descriptor dimensions are similar. Second, an additional advantage of the proposed method is the flexibility to create a higher-dimensional descriptor when needed (e. g., if the task is difficult and a low-dimensional descriptor doesn't provide adequate performance).

Note that the main purpose of this experiment is to compare ICFT and SIFT, and the MNIST dataset is used as a convenient test bed. We do not suggest that computing descriptors on a regular grid is the optimal character recognition method.

B. Text vs. image classification

In this experiment, image patches were classified as either 'text' (corresponding to text regions in a document) or 'image' (corresponding to figures). This classification task was motivated by OCR engine design. Performing OCR on image regions is not useful, consumes resources, and introduces errors into OCR results. It is therefore useful to classify page regions as either text or images, and exclude image regions from OCR.

The 270-dimensional ICFT descriptor was used for this classification task. The classifier using ICFT performed at 99.5% accuracy, compared to 89% accuracy using SIFT.

Grid type	120-ICFT	270-ICFT	SIFT
1×1	6.10	5.30	7.24
2×2	1.89	1.33	2.11
3×3	2.14	1.58	2.97

Table I
ERROR RATES ON THE MNIST DIGIT CLASSIFICATION TASK. LOWER NUMBERS INDICATE BETTER PERFORMANCE.

The conclusion is that ICFT outperforms general-purpose descriptors when applied to document images.

V. CONCLUSION

We have presented Independent Component Feature Transform (ICFT), a method for automatically learning feature descriptors tuned to a given image domain. The method works by extracting the independent components from images and pooling these components over multiple overlapping regions. The proposed method was validated experimentally on several document analysis tasks and several datasets. We have shown that ICFT outperformed existing general-purpose descriptors.

REFERENCES

[1] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *IJCV*, vol. 60, no. 2, pp. 91–110, 2004.

[2] H. Bay, A. Ess, T. Tuytelaars, and L. V. Gool, "SURF: Speeded up robust features," *CVIU*, vol. 110, no. 3, pp. 346–359, 2008.

[3] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *CVPR*, 2005.

[4] S. Lazebnik, C. Schmid, and J. Ponce, "Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories," in *CVPR*, 2006.

[5] K. Grauman and T. Darrell, "The pyramid match kernel: Discriminative classification with sets of image features." in *ICCV*, 2005, pp. 1458–1465.

[6] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[7] J. Chen, H. Cao, R. Prasad, A. Bhardwaj, and P. Natarajan, "Gabor features for offline arabic handwriting recognition," in *DAS*, 2010.

[8] Z. Zhang, L. Jin, K. Ding, and X. Gao, "Character-SIFT: A novel feature for offline handwritten chinese character recognition," in *DAS*, 2009.

[9] T. Wu and S. Ma, "Feature extraction by hierarchical overlapped elastic meshing for handwritten chinese character recognition," in *ICDAR*, 2003.

[10] Y. LeCun, F.-J. Huang, and L. Bottou, "Learning methods for generic object recognition with invariance to pose and lighting," in *CVPR*, 2004.

[11] K. Kavukcuoglu, M. Ranzato, R. Fergus, and Y. LeCun, "Learning invariant features through topographic filter maps," in *CVPR*, 2009.

[12] G. E. Hinton, S. Osindero, and Y. Teh, "A fast learning algorithm for deep belief nets," *Neural Computation*, vol. 18, pp. 1527–1554, 2006.

[13] T. Serre, L. Wolf, and T. Poggio, "Object recognition with features inspired by visual cortex," in *CVPR*, 2005.

[14] A. Hyvarinen and E. Oja, "Independent component analysis: Algorithms and applications," *Neural Networks*, vol. 13, no. 4–5, pp. 411–430, 2000.

[15] M. Riesenhuber and T. Poggio, "Hierarchical models of object recognition in cortex," *Nature Neuroscience*, vol. 2, pp. 1019–1025, 1999.

[16] I. Lampl, D. Ferster, T. Poggio, and M. Riesenhuber, "Intracellular measurements of spatial integration and the max operation in complex cells of the cat primary visual cortex," *J. Neurophysiol.*, vol. 92, no. 5, pp. 2704–2713, 2004.

[17] D. H. Hubel and T. N. Wiesel, "Receptive fields of single neurones in the cat's striate cortex," *J Physiol*, vol. 148, no. 3, pp. 574–591, 1959.

[18] B. A. Olshausen and D. J. Field, "Emergence of simple-cell receptive field properties by learning a sparse code for natural images," *Nature*, vol. 381, pp. 607–609, 1995.

[19] A. Bell and T. J. Sejnowski, "The 'independent components' of natural scenes are edge filters," *Vision Research*, vol. 37, pp. 3327–3338, 1997.

[20] H. V. B. Hirsch and D. N. Spinelli, "Visual experience modifies distribution of horizontally and vertically oriented receptive fields in cats," *Science*, vol. 168, no. 3933, pp. 869–871, 1970.

[21] M. S. Lewicki and B. A. Olshausen, "A probabilistic framework for the adaptation and comparison of image codes," *J. Opt. Soc. Amer. A*, vol. 16, no. 7, pp. 1587–1601, 1999.

[22] B. Epshtein and S. Ullman, "Feature hierarchies for object classification," in *ICCV*, 2005.