Deep Learning with Sparse Prior
Application to Text Detection in the Wild

Adleni Mallek[1], Fadoua Drira[1], Rim Walha[1], Adel M. Alimi[1] and Frank LeBourgeois[2]
[1]*ReGIM-lab, University of Sfax, ENIS, BP 1173, 3038, Sfax, Tunisia*
[2]*LIRIS, University of Lyon, INSA-Lyon, CNRS, UMR5205, F-69621, Lyon, France*
{*adleni.mallek.tn, fadoua.drira, rim.walha, adel.alimi*}*@ieee.org, franck.lebourgeois@insa-lyon.fr*

Keywords: PCANet, Deep Learning, Sparse Coding, Text Detection.

Abstract: Text detection in the wild remains a very challenging task in computer vision. According to the state-of-the-art, no text detector system, robust whatever the circumstances, exists up to date. For instance, the complexity and the diversity of degradations in natural scenes make traditional text detection methods very limited and inefficient. Recent studies reveal the performance of texture-based approaches especially including deep models. Indeed, the main strengthens of these models is the availability of a learning framework coupling feature extraction and classifier. Therefore, this study focuses on developing a new texture-based approach for text detection that takes advantage of deep learning models. In particular, we investigate sparse prior in the structure of PCANet; the convolution neural network known for its simplicity and rapidity and based on a cascaded principal component analysis (PCA). The added-value of the sparse coding is the representation of each feature map via coupled dictionaries to migrate from one level-resolution to an adequate lower-resolution. The specificity of the dictionary is the use of oriented patterns well-suited for textual pattern description. The experimental study performed on the standard benchmark, ICDAR 2003, proves that the proposed method achieves very promising results.

1 INTRODUCTION

Text in an image is a vital source of information very useful mainly for understanding the content of the image. Nowadays, a lot of studies are focusing on text detection regarding various applications like content-based image retrieval and document image analysis. However, even though the tremendous work done across this subject, text detection in the wild remains a very challenging task in computer vision. The study of the state-of-the-art ascertains that no text detector system, robust whatever the circumstances, exists up to date. The major difficulty is raised by the widely use of digital cameras because the image acquisition process in this case could introduce new added distortions in terms of blur, noise, non-uniform illumination and perspective degradations while taking into consideration the diversity of characters' layout (shape, size, position and color). Moreover, the presence of background objects in natural scene images could be at the origin of many false-positive detections. As illustration, we give in Figure 1 some images extracted from the well-known ICDAR 2003 database. These images clearly illustrate the above depicted challenges in text image detection in the wild.

Figure 1: Extracts form the ICDAR 2003 dataset.

Faced with these different constraints, traditional methods for text detection process have proven to be insufficient. Thus, training a text detector, although it is not a trivial task, represents the possible issue for a good text detection. The state-of-the-art distinguishes between two categories of text detection methods according to the data taken into account (Ye and Doermann, 2015): (1) connected component analysis and (2) sliding window classification.

Mallek A., Drira F., Walha R., Alimi A. and LeBourgeois F.
Deep Learning with Sparse Prior - Application to Text Detection in the Wild.
DOI: 10.5220/0006129102430250
In *Proceedings of the 12th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications (VISIGRAPP 2017)*, pages 243-250
ISBN: 978-989-758-226-4

The first category includes bottom-up methods, that proceed in general by a pre-processing step for noise removal, then a segmentation step for character regions extraction followed by a grouping step for word detection. Stroke Width Transform (SWT) (Epshtein et al., 2010; Neumann and Matas, 2013) and Maximally Stable Extremal Regions (MSER) (Neumann and Matas, 2012) are two representative methods of this category for scene text detection. The work of Gao et al. (Gao et al., 2014) introduces a bottom-up visual saliency model utilizing color feature. The work of Yi et al. (Yi and Tian, 2011) uses color features for the extraction of connected components. Even though the simple implementation behind this overall detection process and which explains its popularity, it remains very sensitive to the size and shape of characters, background noise and degraded text patterns.

An alternative solution is sliding window based methods characterized by the recourse to image patches with different sizes on the image pyramid and the application of text/non-text classifiers (Wang et al., 2011; Lee et al., 2011). Thus, this solution includes top-down methods. In particular, these methods exploit predominant features (e.g. texture, edge map...) of text regions characteristics to extract them from the other object of the background. The classification of text/non-text region could use either heuristic methods (Garcia and Apostolidis, 2000; Zhong et al., 1995) or machine learning methods (Lienhart and Wernicke, 2002). The efficiency of heuristic methods relies on an adequate choice of features and heuristic filters. In order to improve the robustness, machine learning methods investigate relevant features over a given neighborhood accompanied with robust classifier that is trained using machine learning techniques to distinguish textual regions. For example, Anthimopoulos et al. (Anthimopoulos et al., 2013) create hand-crafted features specifically designed for text detection. These are dynamically-normalized edge features which generate local binary patterns within the sliding window image patch. Wang et al. (Wang et al., 2011) use a classifier trained on the histogram of oriented gradients features in a sliding window scenario to find characters in an image, grouping them using a pictorial structures model for a fixed lexicon.

The main limitation of these methods is noticeable for objects having similar structural texture to texts. Moreover it remains not efficient in terms of computation cost. Therefore, the focus of text detection system using machine learning tools is twofold: (1) the choice of a robust learning tool with a low-computational cost to perform an efficient classification and (2) the choice of an adequate feature space for a better description of the textual patterns.

Deep learning techniques have become recently a popular trend in machine learning applications. Their main strengthens is the availability of a learning framework coupling feature extraction and classification. Due to their efficiency, we focus on developing a new texture-based approach for text detection which takes advantage of these techniques. In particular, we propose to deal with the structure of the simple convolution neural network PCANet with sparse prior. The choice of this architecture is explained by the improvement of some shortcomings of classic convolution neural network including the high training computational cost time, special tuning parameters and technical problems. Furthermore, the sparse coding offers an automatic generation of sparse vectors according to a dictionary to encode an input signal. The network stacked of multiple feature extraction stages, each of which comprises a convolutional filter bank layer and a feature pooling layer, and a non-linearity stage. For filter bank in each convolution layer, we present an unsupervised feature learning that can be automatically learned by PCA. The feature maps are generated from the results of convolution and are aggregated by pooling layer via sparse prior taking benefit of coupled dictionaries to migrate from one level-resolution to an adequate lower-resolution level. The predefined dictionaries contain patches of writing patterns extracted from a collection of high-quality character images and thus describe the specificities of writing. The non-linearity stage involves binary hashing then concatenates block-wises histogram to finally fed the results into a training classifier.

The remainder of the paper is organized as follows: Section 2 presents related works dealing with deep learning based text detection methods in the wild. Section 3 gives a description of our contribution; an emphasis is made to describe properly both PCANet, the deep learning network architecture and sparse coding representation for a better comprehension of the overall algorithm. After that, Section 4 gives experimental results for performance evaluation of the proposed method within a comparative study involving ICDAR 2003 well-known database. Section 5 closes this study with overall conclusions and suggestions for future research works.

2 RELATED WORKS

The efficiency of deep learning models has been noticeable in various applications of computer vision

like image classification and object detection. In particular, text/non-text classification in natural images is among the most extensively studied application in the literature. If traditional approaches are based on hand-engineering better sets of features (Srinivas et al., 2016) which are very limited for highly challenging textual scene images, recent studies investigate deep learning, heavily based on convolutional deep neural networks (ConvNet), to compute hierarchical features or representations from raw images. Each neuron is related to a feature where the subsequent layer takes a broad view of the essential features from the previous one (Krizhevsky et al., 2012; Girshick et al., 2014; Simonyan and Zisserman, 2015; Szegedy et al., 2015).

In general, a deep network architecture comprises (1) a convolutional filter bank layer, (2) a nonlinear processing layer, (3) and a feature pooling layer. For each layer, the learning process could be achieved using one of the well-known learning techniques such as restricted Boltzmann machines, regularized auto-encoders (Ciresan et al., 2011) or their variations (Socher et al., 2011), the PCANet... The latter represents a very simple deep learning network that can offer a valuable baseline for reading complex deep learning architectures suited for large-scale image applications (Chan et al., 2014).

The underlying assumption for feature design is the use of text/non-text information for training. Therefore, this information is very important to learn a discriminative representation. The proposed deep learning based methods of Wang et al. (Wang et al., 2012), Jaderberg et al. (Jaderberg et al., 2014) and Gupta et al. (Gupta et al., 2016) compute globally image features that lead to insufficient and no generic text/non-text representation. For a discriminative feature learning, He et al. (He et al., 2016) propose a text CNN model that particularly involves text-related specific characteristics such as text region mask, character label and binary text/non-text information.

Therefore, an efficient deep learning based text detection method must encapsulate discriminative text patterns and a deep architecture with a low-computational cost.

3 CONTRIBUTIONS

This section details the proposed text detection system which relies on the two most popular models of machine learning: deep learning and sparse coding representation. In fact, we propose to proceed by a combination of the merits of the PCANet model and the domain expertise of sparse coding to improve the performance of the text detection system with a faster training and adequate feature representation. Therefore, we could learn in a simple and rapid manner the most appropriate feature representation for text image from data. Figure 2 illustrates an overview of the proposed system. Further details are discussed in the following subsections.

3.1 PCANet: Brief Review

PCANet is an example of a simplified deep learning network. Therefore, it shares with ConvNet many of its fundamental representation and functional properties. In particular, it represents a feature that automatically learns from a given region of interest but takes benefit of a more abstract expression of some properties. The convolution filter bank is Principal Component Analysis (PCA) filters. The non-linear layer is the binary hashing (quantization). The pooling layer is the block-wise histogram of the decimal values of the binary vectors. Therefore, the structure of PCANet model contains the following steps : patch-mean removal, PCA filter convolutions to produce a set of feature maps, binary quantization and mapping, block-wise histograms, and an output classifiers (Chan et al., 2014).

Let us denote N the number of the input training images $\{I_i\}_{i=1}^{N}$ of size $m \times n$. For each image, we take around each pixel a $k_1 \times k_2$ patch. Then, the mean is subtracted from each patch and the feature vector is given. After that, the filter is defined for the next step by taking into account the maximum eigenvalues. For the second stage, the same process is repeated.

PCANet model includes a number of PCA stages followed by an output stage. The number of these stages can be varied, but a typical PCANet has two stages. The work of (Chan et al., 2014) emphasized that the two-stage PCANet model outperforms the single stage in most cases. However, the increase of the number of stages does not always improve the final results performance. In particular, this study focuses on the two-stage PCANet model.

We note that the output stage contains the operations of binary hashing and block-wise histogram. A step function is applied to generate binary values for each patch and these values are converted to decimal values using the binary hashing. The block-wise histogram operates on these decimal values to generate the final output features. These features are then fed into a trained classier like SVM. We suppose that the number of filters in stage 1 after PCA application on the N training images is F_1 and in stage 2 is F_2.

3.2 Sparse Coding: Brief Review

The goal of sparse coding is to automatically construct sparse vectors from the training inputs Y. For sparse representation, the following objective function is used to learn a dictionary D and jointly find a sparse linear combination of the basis vectors by minimizing the data reconstruction errors:

$$D = argmin_{D,\alpha} \|Y - D\alpha\|_2^2 + \|\alpha\|_1 \quad (1)$$

where α is the sparse vectors that represent the training samples Y according to the dictionary D, and λ is a constant that controls the sparsity penalty and fidelity of the approximation to Y.

In the literature, we have noticed the recourse to coupled sparse feature spaces: high-resolution and low-resolution feature spaces mainly in patch-based super-resolution tasks (Walha et al., 2015; Walha et al., 2014; Yang et al., 2010). The given spaces could be nominated as the observation space and the latent space. Both of them must be tied by a mapping function. The underlying assumption is to ensure a collaborative learning between these two coupled dictionaries in order to efficiently reconstruct the signal in the latent feature space given the sparse representation of its corresponding signal in the observation space.

Let us denote $\left\{Y_{l_j}^i\right\}_{i=1...C}^{j=1,2}$ the patch pairs for the level 1 (l_1) and level 2 (l_2) of each cluster i where C is the total number of clusters. The coupled dictionaries $\left\{D_{l_j}^i\right\}_{i=1...C}^{j=1,2}$ are thus defined as follows:

$$D_{l_j}^i = argmin_{D_{l_j}^i,\alpha} \left\|Y_{l_j}^i - D_{l_j}^i\alpha\right\|_2^2 + \|\alpha\|_1 . \quad (2)$$

3.3 Proposed PCANet with Sparse Prior

In this section, we propose an approach that combines the simplicity and efficiency of PCANet architecture and the domain expertise of sparse coding. The main difference, in comparison with the PCANet model described above, is that the feature maps, generated from the results of convolution, are aggregated by pooling layer via sparse prior. This is possible via the use of coupled dictionaries to migrate from one level-resolution to an adequate low-resolution level. In this case, a correct selection of the training database must be undertaken carefully for a successful learning process. Attracted by the competitive results of the Walha et al. work (Walha et al., 2015), we choose to proceed using its training database; even it has been already collected to enhance the spatial resolution of textual images, we will adapt it to the context of our study. In fact, this database is composed of several patch images extracted around character edges of high-quality images. These character images, generated using the graphic library FreeType, encapsulates a variety of sizes, styles and fonts. The high-resolution training set is composed by 124000 patch images. Figure 3 shows samples of the high-resolution training set.

Algorithm 1: PCANet with sparse prior.

Input: Training patches (ICDAR 2003), Coupled dictionaries level 1 and level 2.

1: **Stage I:** First feature extraction via PCA

1.a: First filter construction for layer 1 via PCA application on the N training patches.

1.b: Convolution step: For each training patch, extract L_1 feature maps by convoluting it with PCA based filters bank.

2: **Stage II**: Feature pooling via sparse prior

2.a: Each feature map is represented via sparse coding using coupled dictionaries to migrate from one level-resolution to an adequate low-resolution level.

3: **Stage III**: Second feature extraction via PCA

3.a: Second filter construction for layer 2 via PCA application on the pooled features maps

3.b: Convolution Step: extract L_2 feature maps by convoluting the output of the second stage with the second PCA based filters bank.

4: **Stage IV**: Feature preparing for SVM classifier

4.a: Apply binary hashing

4.b: Concatenate block-wise histogram

Output: Learned Features, Learned SVM Classifier.

As output for the Stage I, we obtain NF_1 feature maps where N is the number of the input image and F_1 is the number of filter of the first stage. Firstly, we extract l patches with a size of $k_1 \times k_2$ of the ith image that is vectorized into column vectors with a stride of s pixels. Thus, we collect all patches extracted from the same input image which forms a matrix with a size T, denoted as

$$T = (k_1 k_2) \times ((\frac{\lceil m - k_1 \rceil}{s} + 1)(\frac{\lceil n - k_2 \rceil}{s} + 1)); \quad (3)$$

$$X_i = [x_{i,1}, x_{i,2}, ..., x_{i,j}, ..., x_{i,T}] \in \Re^{(k_1 \times k_2)T} \quad (4)$$

where $x_{ij} \in \Re^{k_1 \times k_2}$ and $j \in 1..l$ denotes the j^{th} vectorized patch in I_i.

In order to improve the features quality within a neighborhood and removing the noise corresponding to the small eigenvalues of the data covariance matrix, we subtract the mean value of the corresponding patch

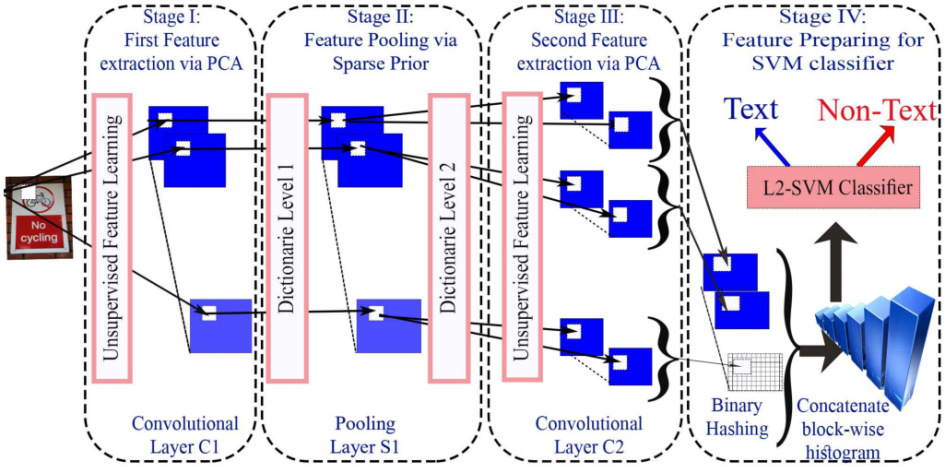

Figure 2: The four stage architecture of the proposed PCANet with sparse prior for text detection.

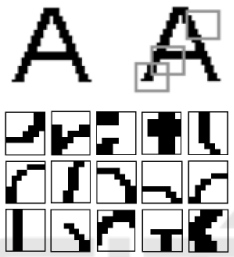

Figure 3: From top to down : Illustration of the selection process of patch images from a character image and samples of the writing pattern from the training database (Walha et al., 2015).

from each column vector in the matrix X_i and obtain the following matrix:

$$\overline{X}_i = [\overline{x}_{i,1}, \overline{x}_{i,2}, ..., \overline{x}_{i,j}, ..., \overline{x}_{i,T}] \in \Re^{(k_1 \times k_2)T} \quad (5)$$

where $\overline{x}_{i,j}$ is a mean-removed patch.

Once the same matrix is applied for all input images, we regroup them to form a large matrix defined as:

$$\overline{X} = [\overline{X}_1, \overline{X}_2, ..., \overline{X}_N] \in \Re^{(k_1 \times k_2)N_{mn}} \quad (6)$$

The eigenvectors associated with the most energetic eigenvalues from the covariance matrix are then extracted. The next step is performed on the matrix XX^T. Therefore, we take the convolutional filters that form a matrix with F_1 principal eigenvectors of XX^T.

The PCA filters that are learned can be expressed as follows:

$$W_f^1 = mat_{k_1 \times k_2}(q_f(XX^T)) \in \Re^{k_1 k_2}, l = 1, 2, ..., F_1 \quad (7)$$

where $mat_{k_1 \times k_2}(v)$ is a function that denotes the mapping relationship from vector v to a matrix $W \in \Re^{k_1 k_2}$, and $q_f(XX^T)$ designates the f^{th} eigenvector of matrix XX^T. Lastly, all the principal eigenvectors of a matrix keep the main variation of all the mean removed training patches and we obtain F_1 filters of size $k1 \times k2$. After that, we convolute each input image with the learned filters to generate filter responses at each pixel location and namely the filtering feature maps results. Each feature map, extracted at corresponding location in the image, represents particular features.

Let the f^{th} filter output of the first stage be as follows:

$$I_i^f = I_i \otimes W_f^1, i = 1, 2, ..., N \quad (8)$$

where $\otimes$ denotes $2D$ convolutional, and I_i padded with zeros before convolution.

For the Stage II, we apply a pooling layer via sparse prior that represents the feature maps using coupled dictionaries. This stage allows to recover a patch of low-resolution level J_i^f from an input patch of one level resolution I_i^f. Given the learned coupled dictionaries $\{D_{l_1}, D_{l_2}\}$ (Walha et al., 2015), sparse coding generates sparse vectors to encode an input patch I_i^f from the dictionary D_{l_1}. This can be mathematically formulated as:

$$\alpha = min_\alpha \left\| I_i^f - D_{l_1}\alpha \right\|_2^2 \quad (9)$$

where α is the sparse representation of I_i^f over D_{l_1}. Then, we select only the representation $\widehat{\alpha}$ that minimizes the local reconstruction error according to the appropriate dictionary $\widehat{D_{l_1}}$:

$$\widehat{\alpha} = min_\alpha \left\| I_i^f - D_{l_1}\alpha \right\|_2^2 \quad (10)$$

The optimal solution $\widehat{\alpha}$ is then applied to generate a low-resolution level of patch J_i^f from the corresponding D_{l_1} dictionary based on: $J_i^f = D_{l_2}\widehat{\alpha}$.

In the case of the Stage III, we apply the same process that is used in the first stage. Firstly, we collect

all the overlapping patches of J_i^f. Then, we join all vectors extracted from which we subtract patch mean to form a matrix denoted as

$$\overline{Y}_i = \left[\overline{y}_{i,f,1}, \overline{y}_{i,f,2}, ..., \overline{y}_{i,f,j}, ..., \overline{y}_{i,f,T}\right] \in \Re^{(k_1 \times k_2)T} \quad (11)$$

where $\overline{y}_{i,j}$ is a mean-removed patch. We further collect patches from all mean-removed patches in the f^{th} filter output, and concatenate the matrix $\overline{Y}_i^f$ denoted as

$$Y^f = \left[\overline{Y}_1^f, \overline{Y}_2^f, ..., \overline{Y}_N^f\right] \in \Re^{(k_1 \times k_2)N_{mn}} \quad (12)$$

The PCA filters are learned and can be expressed as follows:

$$W_f^2 = mat_{k_1 \times k_2}(q_f(YY^T)) \in \Re^{k_1 k_2}, l = 1, 2, ..., F_2 \quad (13)$$

For each input J_i^f of the third stage, we take the first L_2 main eigenvectors as PCA filters convolving with W_f^2 for $f = 1, 2, ..., F_2$

$$V_i^f = \{J_i^f \otimes W_f^2\}_{f=1}^{L_2} \quad (14)$$

The number of features maps of the third stage is $F_1 N F_2$ as output of the second feature extraction.

The stage IV is the output stage, we use binary hashing and histogram statistics which build feature maps to form final representation of the input image as in PCANet (Chan et al., 2014). Each F_1 input feature maps to the second stage obtains F_2 feature maps as outputs $\{J_i^f \otimes W_f^2\}_{f=1}^{F_2}$. We binarize these output maps and apply $\{J_i^f \otimes W_f^2\}_{f=1}^{F_2}$, where $H(.)$ is a Heaviside step function whose value is one for positive entries and zero otherwise.

Around each pixel, we consider the vector of F_2 binary bits as a decimal number. This converts the F_2 outputs produced in the second stag back into a single integer-valued image.

For each of F_1 integer valued images, we partition it into B blocks. We compute the histogram of the decimal values in each block, and concatenate all the B histograms into one vector. After this encoding process, the feature of the input image I_i becomes the set of block-wise histograms. The local blocks can be either overlapping or non-overlapping, depending on applications. The output is then fed into a trained classier.

4 EXPERIMENTAL RESULTS

4.1 Experimental Framework

4.1.1 Datasets and Evaluation Protocols

Our experiments were implemented with Matlab 2015_b platform, and run on a 64-bit Microsoft 10 machine powered by Intel(R) Core(TM) i7-4750HQ CPU 2.00 GHz processor and 8 GB RAM.

In this section, we choose the dataset ICDAR 2003 with challenging text scene images for evaluating our text detection method. This dataset proposed by S. M. Lucas et al. contains 507 natural scene images in total; 258 images from the dataset are prepared for training and 249 images for testing. Most images vary in size from 600×450 to 1280×960 and contain about 4 text regions on average for each image.

To evaluate the text detection, we used three important metrics in performance assessment: Precision (P), Recall (R)and F-Measure (FM). Precision measure is the ratio between the successfully extracted text regions (TP) and all detected regions (E). Recall measure represent the ratio of the successfully extracted text regions (TP) that should be in the ground truth regions (T). The F-Measure is computing the performance of algorithm, which combines the previous two measures. The metric formulas are given by the following definitions:

$$P = |TP| / |E| \quad (15)$$

$$R = |TP| / |T| \quad (16)$$

$$FM = \frac{2 \times P \times R}{(P + R)} \quad (17)$$

4.1.2 Experimental Setup

Our text detection system is based on PCANet architecture and on sparse coding representation. We use samples of ICDAR 2003 dataset to learn PCA filters with L_1 filters in the rst stage and L_1 groups in the third stage; each group contains L_2 filters. To extract the features, we adopted convolution layer that convoluate each input image 32×32 using patch size of $k1 \times k2$ with the learned PCA-based filters to produce a set of feature map. To train dictionary, the number of atoms varies between 128 and 1024. For the classification layer, these outputs are fully connected to train the network by back-propagating the L_2-SVM classication error.

For the experimental study, we choose the number of filter as set $L_1 = L_2 = 8$ that is inspired from the common setting of Gabor filters with 8 orientations. The results are shown in Figure 4.

4.2 Performance Evaluation

In this section, we evaluate the performance of the proposed system on the Robust Reading dataset ICDAR 2003. For text detection, we generate a 2-way

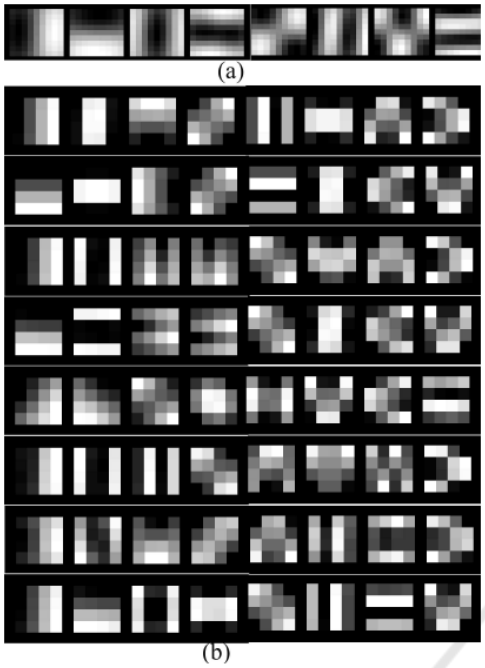

Figure 4: Filters learned on the ICDAR 2003 dataset. (a) 8 filters in the first stage;(b) There are 8 groups in the second stage, and each group contains 8 filters, shown in a column.

text/no-text classication dataset by cropping patches from this dataset.

4.2.1 Impact of the Patch Size

We study the influence of the different patch size y_1 varying between 7 and 11 pixels for the first stage. To calculate the overlapping between adjacent patches from the left to right input image and the patch size y_2 of the third stage, we apply the following formula:

$$y_2 = \frac{(y_1 + 1)}{2}. \tag{18}$$

For ICDAR 2003 dataset, table 1 proves that the best number of patch size is 7 and 4 for Stage I and Stage II respectively.

Table 1: Impact of the patch size for Stage I and Stage II of our proposition: Case study of ICDAR 2003 database.

Patch size		P	R	FM
Stage I	Stage III			
7	4	0.975	0.965	0.97
9	5	0.966	0.958	0.962
11	6	0.96	0.962	0.961

4.2.2 Impact of the Number of Atoms

In this subsection, the patch size is fixed to 7×7 for stage I and 4×4 for stage III. We define the relationship between the classification performance and the number of atoms. To generate the dictionary, we applied the K-means clustering algorithm patches. We used the dictionary of Walha et al. (Walha et al., 2015) to conduct both the two dictionaries generation such as one level-resolution and low-resolution level. Table 2 illustrates that the number of atoms equal to 256 gives better results than the other.

Table 2: Impact of the number of atoms for our proposition: Case study of ICDAR 2003 database.

Number of atoms	P	R	FM
128	0.97	0.962	0.966
256	0.975	0.965	0.97
512	0.976	0.96	0.968

4.2.3 Comparative Methods Study

We compared our proposed text detection system to the-state-of-the-art methods using the comparative metrics already defined above.

Table 3: Comparisons of different methods of text detection: Case study of ICDAR 2003 database.

Algorithm	P	R	FM
Epshtein et al.	0.73	0.60	0.66
Yi and Tian	0.71	0.62	0.62
Huang et al.	0.81	0.74	0.72
Basic PCANet	0.95	0.95	0.95
PCANet/Sparse prior	0.97	0.96	0.97

According to the different metrics such as Precision, Recall and F-Measure established for different algorithms, including basic PCANet(without sparse prior), modified PCANet (with sparse prior) and the works of (Epshtein et al., 2010; Yi and Tian, 2011; Huang et al., 2013), tested on the ICDAR 2003 dataset, table 3 shows that the modified variant of PCANet with sparse prior achieves a better text detection result.

5 CONCLUSIONS

We proposed a novel approach which takes the advantage of unsupervised deep learning. More precisely, we combine the simplicity and efficiency of PCANet architecture and the domain expertise of sparse prior. Our proposition provides competitive results compared with state-of-the art methods but opens more efficient prospects for other methods.

Further investigations will focus on the integration of a dedicated feature to define automatically the filter numbers. The study of neighborhood for a given

image block will be integrated to improve the overall system detection performance.

REFERENCES

Anthimopoulos, M., Gatos, B., and Pratikakis, I. (2013). Detection of artificial and scene text in images and video frames. *Pattern Anal. Appl.*, 16(3):431–446.

Chan, T. H., Jia, K., Gao, S., Lu, J., Zeng, Z., and Ma, Y. (2014). Pcanet: A simple deep learning baseline for image classification. *IEEE Trans. on Image Processing*, 24(12):5017–5032.

Ciresan, D. C., Meier, U., Masci, J., Gambardella, L. M., and Schmidhuber, J. (2011). High performance neural networks for visual object classication. *Technical Report IDSIA-01-11*.

Epshtein, B., Ofek, E., and Wexler, Y. (2010). Detecting text in natural scenes with stroke width transform. *IEEE Computer Vision and Pattern Recognition*, pages 2963–2970.

Gao, R., Uchida, S., Shahab, A., Shafait, F., and Frinken, V. (2014). Visual saliency models for text detection in real world. *PLoS ONE*, 9:114–539.

Garcia, C. and Apostolidis, X. (2000). Text detection and segmentation in complex color images. *In Proc. Int. Conf. on Acoustics, Speech and Signal Processing*, pages 2326–2329.

Girshick, R., Donahue, J., Darrell, T., , and Malik, J. (2014). Rich feature hierarchies for accurate object detection and semantic segmentation. *IEEE Computer Vision and Pattern Recognition*.

Gupta, A., Vedaldi, A., , and Zisserman, A. (2016). Synthetic data for text localisation in natural image. *IEEE Computer Vision and Pattern Recognition*.

He, T., Huang, W., Qiao, Y., and Yao, J. (2016). Text-attentional convolutional neural network for scene text detection. *IEEE Trans. Image Processing*, pages 2529–2541.

Huang, W., Lin, Z., Yang, J., and Wang, J. (2013). Text localization in natural images using stroke feature transform and text covariance descriptors. *IEEE Int. Conf. on Computer Vision*, pages 1241–1248.

Jaderberg, M., Vedaldi, A., and Zisserman, A. (2014). Deep features for text spotting. *European Conf. on Computer Vision*.

Krizhevsky, A., Sutskever, I., and Hinton, G. (2012). Imagenet classification with deep convolutional neural networks. *Neural Information Processing Systems*.

Lee, J.-J., Lee, P.-H., Lee, S.-W., Yuille, A., and Koch, C. (2011). Adaboost for text detection in natural scene. *In Int. Conf. on Document Analysis and Recognition*, pages 429–434.

Lienhart, R. and Wernicke, A. (2002). Localizing and segmenting text in images and videos. *IEEE Trans. on Circuits and Systems for Video Technology*, 12:256–268.

Neumann, L. and Matas, J. (2012). Real-time scene text localization and recognition. *IEEE Computer Vision and Pattern Recognition*, pages 3538–3545.

Neumann, L. and Matas, J. (2013). Scene text localization and recognition with oriented stroke detection. *IEEE Int. Conf. on Computer Vision*, pages 97–104.

Simonyan, K. and Zisserman, A. (2015). Very deep convolutional networks for large-scale image recognition. *Int. Conf. on Learning Representation*.

Socher, R., Pennington, J., Huang, E., Ng, A., and Manning, C. (2011). Semi-supervised recursive autoencoders for predicting sentiment distributions. *In Conf. on Empirical Methods in Natural Language Processing*, pages 151–161.

Srinivas, S., Sarvadevabhatla, R., Mopuri, K., Prabhu, N., Kruthiventi, S., and Radhakrishnan, V. (2016). A taxonomy of deep convolutional neural nets for computer vision. *Frontiers in Robotics and AI*.

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., and Reed, S. (2015). Going deeper with convolutions. *Computer Vision and Pattern Recognition*.

Walha, R., Drira, F., Lebourgeois, F., Garcia, C., and Alimi, A. (2014). Sparse coding with a coupled dictionary learning approach for textual image super-resolution. *Int. Conf. on Pattern Recognition*, pages 4459–4464.

Walha, R., Drira, F., Lebourgeois, F., Garcia, C., and Alimi, A. (2015). Resolution enhancement of textual images via multiple coupled dictionaries and adaptive sparse representation selection. *Int. Journal of Document Analysis and Recognition*, 18(1):87–107.

Wang, K., Babenko, B., and Belongie, S. (2011). End-to-end scene text recognition. *In Int. Conf. on Computer Vision*, pages 1457–1464.

Wang, T., Wu, D. J., Coates, A., and Ng, A. Y. (2012). End-to-end text recognition with convolutional neural networks. *Int. Conf. on Pattern Recognition*, pages 3304–3308.

Yang, J., Wright, J., Huang, T., and Ma, Y. (2010). Image super-resolution via sparse representation. *IEEE Trans. Image Process*, 19(11):2861–2873.

Ye, Q. and Doermann, D. (2015). Text detection and recognition in imagery: A survey. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 37(7):1480–1500.

Yi, C. and Tian, Y. (2011). string detection from natural scenes by structure-based partition and grouping. *IEEE Trans. on Image Processing*, 20(9):2594–2605.

Zhong, Y., Karu, K., and Jain, A. (1995). Locating text in complex color images. *Pattern Recognition*, pages 1523–1536.