A robust hybrid method for text detection in natural scenes by learning-based partial differential equations

Zhenyu Zhao\textsuperscript{a}, Cong Fang\textsuperscript{b}, Zhouchen Lin\textsuperscript{b,c,\ast}, Yi Wu\textsuperscript{a}

\textsuperscript{a}Department of Mathematics, School of Science, National University of Defense Technology, PR China
\textsuperscript{b}Key Laboratory of Machine Perception (MOE), School of EECS, Peking University, PR China
\textsuperscript{c}Cooperative Medianet Innovation Center, Shanghai Jiaotong University, PR China

1. Introduction

Text detection and recognition in natural scene images have received more and more attention in recent years [1–4]. This is because text often provides critical information for understanding the high-level semantics of multimedia content, such as street view data [5,6]. Moreover, the demand of a growing number of applications on mobile devices has brought great interest in this problem. Text detection from natural scene images is very challenging due to the complexity of background, uneven lighting, blurring, degradation, distortion, and the diversity of text patterns.

There have been a lot of methods for scene text detection, which can be roughly divided into three categories: sliding window based methods [7–9], connected component (CC) based methods [5,10,11], and hybrid methods [12]. Sliding window based methods search for possible texts in multi-scale windows in an image and then classify them into positives using a lot of texture features. However, they are often computationally expensive when a large number of windows, with various sizes, need to be checked and complex classification methods are used. CC-based methods firstly extract character candidates as connected components using some low-level features, e.g., color similarity and spatial layout. Then the character candidates are grouped into words after eliminating the wrong ones by connected components analysis (CCA). The hybrid method [12] creates a text region detector to estimate the probabilities of text position at different scales and extract character candidates (connected components) by local binarization. The CC-based and the hybrid methods are more popular than the sliding window based ones because they can achieve a high precision once the candidate characters are correctly detected and grouped. However, such a condition is not often met: the low-level operations are usually unreliable and sensitive to noise, which makes it difficult to extract the right character candidates. The large number of wrong character candidates can cause many difficulties in the post-processing, such as grouping and classification.

Recently, Liu et al. [13] have proposed a framework that learns partial differential equations (PDEs) from training image pairs, which has been successfully applied to several computer vision and image processing problems. It can handle some mid-and-high-level tasks that the traditional PDE-based methods cannot. In [13] they apply learning-based PDEs to object detection, color2-gray, and demosaicking. In [14], they use an adaptive (learning-based) PDEs system for saliency detection. However, these methods [13,14] may not handle text detection well. This is because text
is a man-made object and its interpretation strongly depends on human perception. The complexity of backgrounds, flexible text styles, and variation of text contents make text detection more challenging than previous tasks.

In most cases, texts cover only a small area of the scene image (Fig. 1). So it will be beneficial for post-processing, like character candidate extraction, if we could narrow down the candidates of text region and eliminate annoying backgrounds. Although we cannot expect that the learned PDEs can produce an exactly binary text mask map, because the solution of the learned PDEs should be a more or less smooth function, learning-based PDEs can give us a relatively high quality confidence map as a good reference. So we use learning-based PDEs to design a text region detector. It is much faster than sliding window based methods because its complexity is only $O(N)$, where $N$ is the number of pixels in the image. Some examples containing the detected region candidates are shown in Fig. 1. To make our method complete and comparable to others, we further propose a simple method for detecting texts in the region candidates.

In summary, we propose a new robust hybrid method using learning-based PDEs. It incorporates both top-down scheme and bottom-up scheme to extract texts in natural scene images. Fig. 2 shows the flow chart of our system and Fig. 3 gives an example. A PDEs system is first learnt off-line with $L_1$-norm regularization on the training images. In the top-down scheme, given a test image as the initial condition we solve the learnt PDEs to produce a high quality text confidence map (see Fig. 3(b)). Then we apply a local binarization algorithm (Niblack [15]) to the confidence map to extract text region candidates (see Fig. 3(c)). In the bottom-up scheme, we present a simple connected component based method and apply it to each region candidate to determine accurate text locations. We firstly perform mean shift algorithm and binarization (OTSU [16]) to extract character candidates (see Fig. 3(d)). Then we group these components to text lines simply based on their color and size (see Fig. 3(e)). Next we adopt a two-level classification scheme (character candidates classifier and text candidates classifier) to eliminate the non-text candidates. Then we obtain the final result (Fig. 3(f)). Our system is evaluated on several benchmark databases and has achieved higher F-measures than other methods. Note that the parameters and classifiers are only trained on the ICDAR 2011 database [17] as only this database provides the required training information. But the proposed approach still yields higher precisions and recalls on the SVT databases [5,6] than other state-of-the-art methods. We summarize the contributions of this paper as follows:

- We propose a new hybrid method for text detection from natural scene images. Unlike previous methods, our method consists of loosely coupled top-down and bottom-up schemes, where the latter part can be replaced by any connect component based methods.
- We apply learning-based PDEs for computing a high quality text confidence map, upon which good text region candidates can be easily chosen. Unlike sliding window based methods, the complexity of learning-based PDEs for text candidate proposal is only $O(N)$, where $N$ is the number of pixels. So our learning-based PDEs are much faster. To our best knowledge, this is the first work that applies PDEs to text detection.
- We conduct extensive experiments on benchmark databases to prove the superiority of our method over the state-of-the-art ones in detection accuracy. Note that unlike previous approaches, after computing the text confidence map, all the procedures are very simple. The performance could be further improved if more sophisticated and ad hoc treatments are involved.

The rest of this paper is organized as follows. Section 2 briefly reviews the related work. Section 3 describes the top-down scheme and Section 4 describes the bottom-up scheme. We discuss the relationship between our method and some related work in Section 5. Section 6 presents experiments that compare the proposed method and the state-of-the-art ones on several public databases. We conclude our paper in Section 7.
2. Related work

Most sliding window based methods first search for possible texts in multi-scale windows and then estimate the text existence probability by using classifiers. Zhong et al. [1] adopt image transformations, such as discrete cosine transform and wavelet decomposition, to extract features. They remove the non-text regions by thresholding the filter responses. Kim et al. [7] extract texture features from all local windows in every layer of image pyramid, which enables the method to detect texts at variable scales. They use a fast mode seeking process named continuously adaptive mean-shift (CAMSHIFT) to search for the text positions and an SVM classifier to generate text probability maps. Li et al. [18] use first and second order moments of wavelet decomposition responses as local region features and a neural network classifier to filter the negative candidates. To speed up text detection, Chen and Yuille [8] propose a fast text detector using a cascade AdaBoost classifier, whose weak learners are selected from a feature pool containing gray-level, gradient, and edge features. After classification the windows are further grouped with morphological operations [19], conditional random fields [6] or graph based methods [20]. The advantage of these methods lies in the simple and adaptive training-detection architecture. Yet that a large number of window candidates need to be classified results in their expensive computational cost.

Unlike sliding window based methods, CC-based methods first extract character candidates from images by connected component analysis and then group the character candidates into words. Additional classifier may be used to eliminate false positives. CC-based methods have become the focus of several recent work thanks to its low computational cost. In addition, the located text components can be directly used for recognition. Based on the color uniformity of characters in a text string, Yi and Tian [21] propose a color-based partition scheme, which applies weighted k-means clustering in the RGB space to separate text and background pixels. In [22] Shivakumara et al. filter the image in the frequency domain, by using the Fourier–Laplacian transform, and then apply k-means clustering to identify candidate components. Recently, two methods, Maximally Stable Extremal Regions (MSER) [23,24] and Stroke Width Transform (SWT) [5,25,26] have been widely used because of their effectiveness of extracting character/component candidates. Yin et al. [24] use a pruning algorithm to select appropriate MSERs as character candidates and hybrid features to validate the candidates, achieving state-of-the-art performance on the ICDAR 2011 database [17]. Using SWT, Yao et al. [25] follow Epshtein et al.’s work [5] in pixel-level filtering and grouping. Then, after running heuristic filtering, two classifiers were trained and applied to remove the outliers in components and text lines. Huang et al. [26] develop a novel Stroke Feature Transform (SFT) filter and two Text Covariance Descriptors (TCDS) for text detection and get a significant performance improvement. In implementations of CCs, syntactic pattern recognition methods are often used to analyze the spatial and feature consensus and to define text regions.

The most related work to ours is the recently proposed hybrid method by Pan et al. [12]. They create a text region detector which computes the text confidence values of sub-windows by using the

\[
\begin{align*}
\text{inv}_{ij}(u, v) & = 1, v, u \\
0, 1, 2 & = \|\nabla v\|^2 = v_x^2 + v_y^2, \|\nabla u\|^2 = u_x^2 + u_y^2 \\
3, 4 & = \nabla^2 v = v_{xx} + v_{yy}, \nabla^2 u = u_{xx} + u_{yy} \\
5 & = \text{tr}(H_v) = v_{xx} + v_{yy}, \text{tr}(H_u) = u_{xx} + u_{yy} \\
6, 7 & = \nabla^2 H_v = v_{xx} + v_{yy}, \nabla^2 H_u = u_{xx} + u_{yy} \\
8 & = (\nabla^2 v)^2 = (v_{xx} + v_{yy})^2, (v_{xx} + v_{yy})^2, \nabla^2 H_v = u_{xx} + u_{yy} + u_{yy} \nabla^2 H_u = u_{xx} + u_{yy} + u_{yy} \\
9 & = (\nabla^2 H_v) \nabla^2 H_u = (v_{xx} + v_{yy})^2, \nabla^2 H_v = u_{xx} + u_{yy} + u_{yy} \nabla^2 H_u = u_{xx} + u_{yy} + u_{yy} \\
10 & = (\nabla^2 H_v) \nabla^2 H_u = (v_{xx} + v_{yy})^2, \nabla^2 H_v = u_{xx} + u_{yy} + u_{yy} \nabla^2 H_u = u_{xx} + u_{yy} + u_{yy} \\
11 & = (\nabla^2 H_v) \nabla^2 H_u = (v_{xx} + v_{yy})^2, \nabla^2 H_v = u_{xx} + u_{yy} + u_{yy} \nabla^2 H_u = u_{xx} + u_{yy} + u_{yy} \\
12 & = \text{tr}(H_v) \nabla^2 H_v = v_{xx} + v_{yy}, \text{tr}(H_u) \nabla^2 H_u = u_{xx} + u_{yy} \\
13 & = 1, v, u \\
14 & = \text{tr}(H_v) \nabla^2 H_v = v_{xx} + v_{yy}, \text{tr}(H_u) \nabla^2 H_u = u_{xx} + u_{yy} \\
15 & = \text{tr}(H_v) \nabla^2 H_v = v_{xx} + v_{yy}, \text{tr}(H_u) \nabla^2 H_u = u_{xx} + u_{yy} \\
16 & = \text{tr}(H_v) \nabla^2 H_v = v_{xx} + v_{yy}, \text{tr}(H_u) \nabla^2 H_u = u_{xx} + u_{yy}
\end{align*}
\]

Table 1

Fundamental differential invariants up to the second order, where \(\text{tr}\) is the trace and \(f\) and \(H_j\) are the gradient and the Hessian matrix of function \(f\), respectively.

---

Fig. 3. Text detection process. (a) The original image. (b) Text confidence map. (c) Text region candidates (in green boxes). (d) Character candidates (in different colors). (e) Text candidates (in green boxes). (f) Final result (in green boxes). (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this paper.)
Histogram of Oriented Gradients (HOG) feature and then extract connected components by local binarization. After that they use a conditional random field model to eliminate the non-character components and group text components into text lines/words with an energy minimization model. Our method computes the text confidence value of each pixel by learning-based PDEs and then group the pixels into text region candidates. It removes lots of backgrounds and hence makes the character candidates extraction much easier. After detecting text region candidates, any CC-based methods can be used. So our method has a flexible structure.

Figure 4. Evolution of PDEs. (a) and (b) is an input/output training image pair. (c)–(e) show the evolutionary results of the learned PDEs at time $n = 1, 3, 6$, respectively. (f) is the text confidence map output by the learned PDEs.

3. Top-down: text confidence map computation and text region candidates extraction

As mentioned before, the proposed method has both the top-down scheme and the bottom-up scheme. We introduce the top-down scheme in this section and leave the bottom-up scheme in next section. We first introduce learning-based PDEs and then apply them to compute the text confidence map. Then we adopt a local thresholding method for text region candidates extraction.

3.1. Text confidence map computation

In this subsection, we first give a brief overview on learning-based PDEs. Then we propose to use the $L_1$ norm regularization for the coefficients of PDEs and explain the algorithm for learning-based PDEs. At last, we introduce the numerical implementation for solving the PDEs to compute the text confidence map.

3.1.1. Learning-based PDEs

Learning-based PDEs have been successfully applied to several computer vision and image processing problems, such as denoising, deblurring, demosaicking, and saliency detection [13,14]. It is assumed that the evolution of the image $u$ is guided by an indicator function $v$, which collects large scale information. It is grounded on the translational and rotational invariance of computer vision and image processing problems. Namely, when the input image is translated or rotated, the output image is translated or rotated at the same amount. It can be proven that the governing equations are functions of fundamental differential invariants (Table 1), which form “bases” of all differential invariants that are invariant with respect to translation and rotation [13].
Assuming linear combination of the fundamental differential invariants, the formulation of PDEs is as follows:

$$
\begin{align*}
\frac{\partial u}{\partial t} + \sum_{i=0}^{16} a_i(t) \text{inv}(u, v) &= 0, \quad (x, y, t) \in Q, \\
u(x, y, t) &= 0, \quad (x, y, t) \in \Gamma, \\
u(x, y, 0) &= f_u, \quad (x, y) \in \Omega, \\
\frac{\partial v}{\partial t} + \sum_{i=0}^{16} b_i(t) \text{inv}(u, v) &= 0, \quad (x, y, t) \in Q, \\
v(x, y, t) &= 0, \quad (x, y, t) \in \Gamma, \\
v(x, y, 0) &= f_v, \quad (x, y) \in \Omega,
\end{align*}
$$

(1)

where \( f_u \) and \( f_v \) are the initial functions of \( u \) and \( v \), respectively, which are determined by the input image \( I \) (Fig. 4a), \( \Omega \subset \mathbb{R}^2 \) is the (rectangular) region occupied by the image, \( T \) is the temporal span of evolution which can be normalized as 1, \( Q = \Omega \times [0, T] \), \( \Gamma = \partial Q \times [0, T] \), and \( \partial Q \) is the boundary of \( \Omega \). So learning the PDEs reduces to determining the linear combination coefficients \( a_i(t) \) and \( b_i(t) \) among the fundamental differential invariants. That the coefficients are functions of time \( t \) only and independent of spatial variables is a consequence of the translational and rotational invariance of the PDEs.

The optimal coefficients should minimize the difference between the output of PDEs, when the initial function is the input training image as its initial condition.

3.1.2. Algorithm for solving the coefficients

Liu and Lin et al. [13] propose a gradient descent method to solve for the optimal coefficients \( a(t) \) and \( b(t) \) in (1), where the “gradient” is actually the Gâteaux derivatives [27] of the objective functional \( E \) with respect to the coefficient functions. The gradient descent method is time-consuming and only fits for smooth regularizations on the coefficient functions, such as the squared \( L_2 \) norm in (2).

Zhao et al. [28] propose a new algorithm by minimizing the difference between the expected outputs \( O_m \) (Fig. 4b) and the actual outputs \( u_m(x, y, t) \) of the PDEs at time \( t \). We adopt this algorithm to solve the optimal control problem (3)-(4).

We first discretize the temporal variable \( t \) with a step size \( \Delta t \) and denote \( t_i = i \cdot \Delta t, i = 0, \ldots, N \). In the sequel, for brevity we use \( u_{m,T} \) or \( u_{m,T}(x, y, t) \) instead of \( u_m(x, y, t) \) if no ambiguity can occur. Other notations, such as \( v_{m,T} \), \( a^T \), and \( B^T \) are understood similarly.

Forward scheme is used to approximate the governing equations in (4). Namely,

$$
\begin{align*}
\frac{u_{m,T+1} - u_{m,T}}{\Delta t} + \sum_{i=0}^{16} a_i(t) \text{inv}(u_{m,T}, v_{m,T}) &= 0, \quad n \geq 0, \\
v_{m,T} &= v_{m,T-1} + \Delta t \left( B_m^{-1} \right)^T \cdot \text{inv}(u_{m,T}, v_{m,T}), \quad n \geq 1,
\end{align*}
$$

(5)

Following Zhao et al. [28]’s scheme, we relax to minimize the difference between the expected outputs \( O_m \) and the actual outputs \( u_{m,T} \) of the PDEs in time order. Then the problem reduces to

$$
\begin{align*}
\min_{a^T, B^T} L^{n+1} (a^T, B^{-1}) &= \frac{1}{2} \sum_{m=1}^{M} \int_{\Omega} \left( O_m - u_{m,T+1} \right)^2 \, d\Omega \\
&+ \lambda_1 \| a^T \|_1 + \lambda_2 \| B^{-1} \|_1, \\
\end{align*}
$$

(6)

at each time \( t_n \), where \( \| a^T \|_1 = \sum_{i=0}^{16} \| a_i(t) \| \) is the sum of the absolute values of its components and \( \| B^{-1} \|_1 = \sum_{j=1}^{3} \sum_{i=0}^{68} \| B^{-1}_{ij} \| \).

For \( n \) (time \( t_n \)), to solve for \( a^T \), by using the first equation in (5) problem (6) reduces to minimize

$$
\begin{align*}
L(a^T) &= \frac{1}{2} \sum_{m=1}^{M} \int_{\Omega} \left[ O_m - u_{m,T} - \Delta t \left( a^T \right)^T \cdot \text{inv}(u_{m,T}, v_{m,T}) \right]^2 \, d\Omega \\
&+ \lambda_1 \| a^T \|_1,
\end{align*}
$$

(7)
When using binomial expansion and interchanging integration and summation, the first term of (7) can be rewritten as follows:

\[
\frac{(\Delta t)^2}{2} \sum_{m=1}^{M} \int_{\Omega} (u^{n}_m) \cdot \nabla (u^{n}_m) \cdot \nabla (u^{n}_m) \cdot a^0 \, d\Omega
\]

\[-\Delta t \sum_{m=1}^{M} \int_{\Omega} (O_{m} - u^{0}_m) \nabla (u^{n}_m) \cdot \nabla (u^{n}_m) \cdot a^0 \, d\Omega
\]

\[+ \frac{1}{2} \sum_{m=1}^{M} \int_{\Omega} (O_{m} - u^{0}_m)^2 \, d\Omega
\]

\[= \frac{(\Delta t)^2}{2} \sum_{m=1}^{M} \int_{\Omega} \nabla (u^{n}_m) \cdot \nabla (u^{n}_m) \cdot (u^{n}_m) \cdot a^0 \, d\Omega
\]

\[-\Delta t \sum_{m=1}^{M} \int_{\Omega} (O_{m} - u^{0}_m) \nabla (u^{n}_m) \cdot \nabla (u^{n}_m) \cdot a^0 \, d\Omega
\]

\[+ \frac{1}{2} \sum_{m=1}^{M} \int_{\Omega} (O_{m} - u^{0}_m)^2 \, d\Omega.
\]

We can see that it is a quadratic programming problem with L1-norm regularization for a0. Many efficient algorithms that guarantee globally optimal solutions have been proposed, such as Gradient Projection (GP) [29], Homotopy [30], Iterative Shrinkage-Thresholding (IST) [31], Accelerated Proximal Gradient (APG) [32], and Alternating Direction Method (ADMM) [33].

3.1.3. Numerical implementation

We need to prepare the input/output training image pairs (in our experiments, we only use M=60 pairs) before learning the coefficients. The input training image is like Fig. 4(a), which is the initial value of \(\mathbf{u}\). The initial value of \(\mathbf{v}\) is taken as \(1\) at each pixel. The output training image is the text mask image, like Fig. 4(b).

To compute the spatial derivatives and integrations, we need to do spatial discretization. We use central differences to approximate the derivatives:

\[
\frac{df}{dx} = \frac{f(x+1) - f(x-1)}{2},
\]

\[
\frac{df}{dy} = \frac{f(x+1) - f(x-1)}{2}.
\]

The discrete forms of \(\frac{df}{dx}, \frac{df}{dy}\), and \(\frac{d^2f}{dx^2}\) can be defined similarly. In addition, we discretize the integrations as:

\[
\int_{\Omega} f(x,y) \, d\Omega = \frac{1}{|\Omega|} \sum_{(x,y) \in \Omega} f(x,y).
\]

where \(|\Omega|\) is the number of pixels in \(\Omega\).

We choose \(A_1 = \lambda_2 = 10^{-3}\) and show the evolution of one image governed by learnt PDEs in Fig. 4. It can be seen that at the beginning steps the text confidence values may not be very high (see Fig. 4(c)). However, the values increase gradually with the evolution of PDEs (see Fig. 4(d)–(f)).

3.2. Text region candidates detection

When given a test image, we compute the confidence map (Fig. 4(f)) by solving the learned PDEs, with the test image being the initial condition. Then we adopt Niblack's local binarization algorithm [15] due to its high efficiency and robustness to image degradation. After binarization, we find out the 8-connected atom components and then group them into text region candidates.

For clustering, we consider the atom components as the vertices \(V\) and build an undirected graph \(G = (V,E)\). Two vertices are connected if and only if the following three conditions are satisfied. Let \((x_i,y_i)\) denote the top left corner of the bounding rectangle of atom component \(V_i\), and \(w_i\) be the height and the width of the bounding rectangle, respectively, and \([c_{11},c_{22},c_{33}]\) be the means of three color channel values of \(V_i\). Then the conditions on \(V_i\) and \(w_i\) are:

- The bounding rectangles should align. So \(\frac{\min(|y_i - y_j|, |y_i + h_i - y_j - h_j|)}{\min(h_i, h_j)}\) should be less than a threshold \(T_1\).
- The two atom components should not be too far from each other. So the interval \(\frac{\min(|x_i + x_j - w_i|, |x_i - x_j - w_i|)}{\min(w_i, w_j)}\) should not be greater than a threshold \(T_2\).
- The atom components in the same region need to have similar colors. So the color difference \(\sqrt{(c_{11} - c_{1j})^2 + (c_{22} - c_{2j})^2 + (c_{33} - c_{3j})^2}\) should be lower than a predefined threshold \(T_3\).

In our experiments, we set \(T_1 = 0.5, T_2 = 2, T_3 = 50\). We find the connected components of the graph \(G\) as different clusters. After clustering, we calculate the minimum circumscribed
rectangle as the text region candidates. We further use simple heuristic rules, such as the ratio of the width to the height of the minimum circumscribed rectangle, to remove wrong text region candidates. All thresholds and parameters for the geometric tests were learned on the fully annotated training set ICDAR 2011 [17].

4. Bottom-up: text detection in the region candidates

Although using learning-based PDEs for text confidence map computation is our major contribution, to make our method complete and comparable to others, in this section we propose a simple framework to extract text strings in the detected text region candidates. It can be replaced by any CC-based methods and the performance can be further improved if more advanced CC-based methods are employed. The framework is the bottom-up scheme and is mainly inspired by [21,26]. It consists of three main steps: character candidates extraction, text candidates construction, and final result verification.

4.1. Character candidates extraction

For character candidates extraction, color and edge/gradient features [21] are conventionally used. Recently, stroke width (e.g., SWT [5,25,26]) and region features (e.g., MSER [23,24]) have been widely employed. Here we use two methods, binarization and mean shift [34], which simply use the color features. For binarization, we just adopt the method OSTU [16]. For mean shift, we select three are the same as those in Section 3.2. Following the same

denotations and denoting \( |V| \) as the number of points in \( V \), we present three more conditions as follows:

- The area of each character in a text line should be similar. So the difference
  \[
  \min(|V_i|, |V_j|)
  \]
  should not be greater than \( T_a \).
- Considering the capital and lowercase characters, the height ratio
  \[
  \frac{|h_i - h_j|}{\max(h_i, h_j)}
  \]
  should be less than \( T_s \).
- Considering the different widths of characters, the width ratio
  \[
  \frac{|w_j - w_i|}{\max(w_i, w_j)}
  \]
  should be less than \( T_w \).

In our experiments, we set \( T_s = 0.25 \), \( T_w = 3 \), \( T_a = 50 \), \( T_s = 3 \), \( T_w = 0.6 \), and \( T_w - 0.6 \). After clustering, we calculate the minimum circumscribed rectangle as the text candidates.

4.2. Text candidates construction

There are two general approaches, rule-based methods [39,25,10] and cluster-based methods [12,24], for text candidates construction. The rule-based methods assume that characters in a word can be fit by one or more top and bottom lines and text candidate need to connect three or more character candidates. Pan et al. [12] and Yin et al. [24] cluster the character candidates with a learned distance metric. Here we simply follow the same way as we extract the text region candidates to group the character candidates into text candidates. We consider the character candidates as the vertices \( V \) and build an undirected graph \( G = (V, E) \). Then we find the connected components of the graph as clusters. To build the edge \( E \) between two character candidates (vertices), we check whether the following six conditions are satisfied. The first three are the same as those in Section 3.2. Following the same

<table>
<thead>
<tr>
<th>Methods</th>
<th>Description</th>
<th>Precision</th>
<th>Recall</th>
<th>F-measure</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pan et al. [12]</td>
<td>Hybrid</td>
<td>0.67</td>
<td>0.67</td>
<td>0.71</td>
</tr>
<tr>
<td>Huang et al. [26]</td>
<td>CC (SWT)</td>
<td>0.81</td>
<td>0.74</td>
<td>0.72</td>
</tr>
<tr>
<td>Yao et al. [25]</td>
<td>CC (SWT)</td>
<td>0.69</td>
<td>0.66</td>
<td>0.67</td>
</tr>
<tr>
<td>Epstein et al. [5]</td>
<td>CC (SWT)</td>
<td>0.73</td>
<td>0.60</td>
<td>0.66</td>
</tr>
<tr>
<td>Chen et al. [46]</td>
<td>CC (MSER)</td>
<td>0.73</td>
<td>0.60</td>
<td>0.66</td>
</tr>
<tr>
<td>Neumann and Matas [48]</td>
<td>CC (MSER)</td>
<td>0.65</td>
<td>0.64</td>
<td>0.63</td>
</tr>
<tr>
<td>Wang et al. [49]</td>
<td>CC</td>
<td>0.77</td>
<td>0.61</td>
<td>0.68</td>
</tr>
<tr>
<td>Yi and Tian [50]</td>
<td>CC</td>
<td>0.71</td>
<td>0.62</td>
<td>0.63</td>
</tr>
<tr>
<td>Zhang and Kasturi [51]</td>
<td>CC</td>
<td>0.73</td>
<td>0.62</td>
<td>-</td>
</tr>
<tr>
<td>Yi and Tian [21]</td>
<td>CC</td>
<td>0.71</td>
<td>0.62</td>
<td>0.62</td>
</tr>
<tr>
<td>Lee et al. [38]</td>
<td>Sliding window</td>
<td>0.66</td>
<td>0.75</td>
<td>0.70</td>
</tr>
</tbody>
</table>

4.3. Final result verification

There may be many false positives in text candidates because a small piece of components/patches may not contain sufficient information for classification. Recently, a two-level classification scheme (character candidates classifier and text candidates classifier) is widely used for text verification. Inspired by Yin et al. [24],

<table>
<thead>
<tr>
<th>Methods</th>
<th>Description</th>
<th>Precision</th>
<th>Recall</th>
<th>F-measure</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pan et al. [12]</td>
<td>Hybrid</td>
<td>0.87</td>
<td>0.67</td>
<td>0.76</td>
</tr>
<tr>
<td>Huang et al. [26]</td>
<td>CC (SWT)</td>
<td>0.81</td>
<td>0.74</td>
<td>0.72</td>
</tr>
<tr>
<td>Yao et al. [25]</td>
<td>CC (SWT)</td>
<td>0.69</td>
<td>0.66</td>
<td>0.67</td>
</tr>
<tr>
<td>Epstein et al. [5]</td>
<td>CC (SWT)</td>
<td>0.73</td>
<td>0.60</td>
<td>0.66</td>
</tr>
<tr>
<td>Chen et al. [46]</td>
<td>CC (MSER)</td>
<td>0.73</td>
<td>0.60</td>
<td>0.66</td>
</tr>
<tr>
<td>Neumann and Matas [48]</td>
<td>CC (MSER)</td>
<td>0.65</td>
<td>0.64</td>
<td>0.63</td>
</tr>
<tr>
<td>Wang et al. [49]</td>
<td>CC</td>
<td>0.77</td>
<td>0.61</td>
<td>0.68</td>
</tr>
<tr>
<td>Yi and Tian [50]</td>
<td>CC</td>
<td>0.71</td>
<td>0.62</td>
<td>0.63</td>
</tr>
<tr>
<td>Zhang and Kasturi [51]</td>
<td>CC</td>
<td>0.73</td>
<td>0.62</td>
<td>-</td>
</tr>
<tr>
<td>Yi and Tian [21]</td>
<td>CC</td>
<td>0.71</td>
<td>0.62</td>
<td>0.62</td>
</tr>
<tr>
<td>Lee et al. [38]</td>
<td>Sliding window</td>
<td>0.66</td>
<td>0.75</td>
<td>0.70</td>
</tr>
</tbody>
</table>
Yao et al. [25], and Huang et al. [26], we utilize a two-level classification scheme for the final result verification. The two levels of features, character level features and text level features are as follows:

1. **Character level features**: There are four kinds of features. The first kind is the occupation percentage of the component and the second is the ratio between the height and width. The third is the ratio of the larger value (between width and height) and its stroke distance mean. The last is a 36-dimensional feature generated by the TCD-C descriptors proposed by Huang et al. [26], but without the feature of stroke width values.

2. **Text level features**: There are three different types of descriptors. Firstly, a 55-dimensional feature generated by the geometric TCD-T descriptors proposed by Huang et al. [26] is used. Secondly, a 16-bin HOG [40] is computed from the text candidate of all the components. Inspired by [41], we compute the correlation matrix between the bins, generating a 136-dimensional feature. Thirdly, there are 42 global features, consisting of local energy of Gabor filter [42,19] and statistical texture measure of image histogram [41,19]. In total, they form a 233-dimensional feature.

After choosing the features of character and text candidates we use the GML AdaBoost toolbox to train two independent classifiers, i.e., the character-level classifier and the text-level classifier. They generate a confidence value \( p_1(c) \) for each character candidate \( c \) and an initial confidence value \( p_2(T) \) for each text candidate \( T \). As introduced in Section 4.1, we first use a simple thresholding method to eliminate the negative character candidates. Then we group the positive ones into text candidates. For a text candidate \( T \) which contains \( n \) character candidates \( c_i, i = 1, \ldots, n \), the final confidence value is defined as 

\[
P(T) = \sum_{i=1}^{n} p_1(c_i)/2n + p_2(T)/2.
\]

The final result verification is produced by simply thresholding the final confidence value.

5. **Discussions**

In this section, we discuss the connection between our methods and some related work.

5.1. **Comparison with existing learning-based PDEs**

Recently, Liu et al. [43,13] and Zhao [28] combine the learning strategy with PDE-based methods for image processing. Although both theirs and our work aim at learning a PDEs system, our work is different from them. Liu et al. [43,13] use the \( L_2 \)-norm regularization for the coefficients, while Zhao et al. [28] utilize the \( L_\infty \)-norm. Here, we exploit the \( L_1 \)-norm regularization which is more suitable for color image processing due to the significantly increased number of fundamental differential invariants. At the best we know, we are the first to apply PDEs to text detection.

![Fig. 5. Successful examples (top two rows) and failure examples (bottom row) from the ICDAR databases.](image-url)
5.2. Comparison to Pan et al.’s hybrid method

As mentioned before, Pan et al.’s hybrid method [12] proposes a text region detector based HOG pyramid and a boosted cascade classifier to produce a text confidence map. Then they use a local binarization algorithm to get the character candidates from the text confidence map. After that, they use a conditional random fields model to eliminate the non-characters and group text components into text lines/words using an energy minimization method. In comparison, we use learning-based PDEs to compute a text confidence map and detect text region candidates from the map. Then any CC-based methods can work on these region candidates. Experiment on ICDAR 2005 in Section 6 achieves a F-measure which is 0.07 higher than Pan et al.’s [12].

6. Experimental results

In this section, we compare our method with several state-of-the-art methods on a variety of public databases, including the ICDAR 2005 [44,45], ICDAR 2011 [17], and the two street view text databases, i.e., the SVT 2010 database [5] and the SVT 2011 database [6]. The performance of these methods is quantitatively measured by precision (P), recall (R), and F-measure (F). They are computed using the same definitions as those in [44,45] at the image level. The overall performance values are computed as the average values of all images in the database.

The training output images for learning-based PDEs are the text mask binarization maps (see Fig. 4). As only the ICDAR 2011 database gives the mask maps, all the parameters (include the coefficients of PDEs and the classifiers) are learned on the ICDAR 2011 training database. We use the same parameters when testing our method on ICDAR 2005 and the two SVT databases.

6.1. Experiments on the ICDAR databases

ICDAR 2005 [44,45] and ICDAR 2011 [17] have been widely used as the benchmarks for text detection in natural images. The ICDAR 2005 database includes 509 color images with image sizes varying from 307 × 93 to 1280 × 960 pixels. It contains 258 images in the training set and 251 images for testing. The ICDAR 2011 database contains 229 training images and 255 testing ones. Both databases are evaluated in the word level, and have 1114 and 1189 words annotated in their test sets, respectively.

The performances of the proposed approach on the two databases are shown in Tables 2 and 3. By comparison with results presented in [26,12,24], our precisions and F-measures are higher while the recalls are comparable with the highest recall values. It demonstrates a significant improvement over representative methods, such as the SWT-based methods [26,25,5] and MSER-based methods [46,24,47]. Since our bottom-up processing is simple and similar to them, our performance improvement could be attributable to learning-based PDEs that produce relatively good text confidence maps.

The top two rows of Fig. 5 show some successful results. They suggest that our system is robust against large variations in text font, color, size, and geometric distortion. The bottom row of Fig. 5 shows some failure cases, including a single large-scale character, highly blurred texts, and unusual fonts.

6.2. Experiments on the SVT databases

We also test our method on the two widely used street view text databases, the SVT 2010 database [5] and the SVT 2011 database [6]. The SVT 2010 database consists of 307 color images of sizes ranging from 1024 × 1360 to 1024 × 768 pixels. The SVT 2011 database contains 647 words and 3796 letters in 249 images collected from Google Street View. However, the format of the ground truth of SVT 2011 database [6] is different from those of...
previous databases as only some words are annotated. Both
databases are more challenging than the ICDAR ones. The images
in these two database are noisier and they contain many repeating
structures such as windows and bricks. Some images are captured
in a low light environment and texts tend to be small, blurred, and
of low contrast (see Fig. 6). Note that we still use the same
parameters (including the coefficients of PDEs and the classifiers) trained on the ICDAR 2011 for these two SVT databases. In Tables 4 and 5, we show the performances on the two SVT databases. It can be seen that our method has a better performance even if it was trained on the ICDAR 2011 database. For the SVT 2011 database, the detection task is very different: localizing words in images (if present), where the words are from a lexicon. Our method has no prior knowledge about the content of the image and its output is not limited by a fixed lexicon. Nonetheless, we still get a higher F-measure of 0.49 using the same evaluation protocol as in the previous section. Fig. 6 shows some detection examples on these challenging databases.

6.3. Effects of learning-based PDEs

In this section we show the effectiveness of learning-based PDEs by comparing the results with and without learning-based PDEs. When learning-based PDEs are not utilized to propose text region candidates, we simply apply the bottom-up procedure to the whole image, i.e., we extract the character candidates on the whole image directly (see Fig. 7(b)). Some comparison examples are shown in Fig. 7. The complexity of the background makes the traditional method difficult to extract the right character candidates on the whole image directly. In contrast, the learning-based PDEs generates region candidates for each image, correctly removing a lot of distractive background. As shown in Fig. 3, some of the text region candidates are so good that we can get correct character candidates simply by image binarization. So we can extract more correct texts, enhancing both the precision and the recall. Table 6 shows the final results comparison on the ICDAR 2011 database with and without learning-based PDEs. All of the precision, recall, and F-measure have been greatly improved if learning-based PDEs are used.

7. Conclusions

In this paper, we propose a novel hybrid approach for text detection in natural scene images. We apply learning-based PDEs to provide text region candidates and devise a simple connected components based method to locate the texts accurately in each text region candidate. Experiment results show the robustness and superiority of our method when compared to many state-of-the-art approaches. In the future, we plan to develop even better operators via learning-based PDEs for robust text region candidates detection from complex backgrounds. We also attempt to apply learning-based PDEs to other high-level vision tasks, such as segmentation and recognition of specific objects and video analysis.

Acknowledgment

Zhenyu Zhao is supported by National Natural Science Foundation of China (NSFC) (Grant no. 61473302), Zhourchen Lin is supported by National Basic Research Program of China (973 Program) (Grant no. 2015CB352502), NSFC (Grant nos. 61272341 and 61231002), and Microsoft Research Asia Collaborative Research Program.

Table 6

<table>
<thead>
<tr>
<th>Scheme</th>
<th>Precision</th>
<th>Recall</th>
<th>F-measure</th>
</tr>
</thead>
<tbody>
<tr>
<td>With learning-based PDEs</td>
<td>0.88</td>
<td>0.69</td>
<td>0.78</td>
</tr>
<tr>
<td>Without learning-based PDEs</td>
<td>0.79</td>
<td>0.63</td>
<td>0.70</td>
</tr>
</tbody>
</table>

References

Zhenyu Zhao received the B.S. degree in mathematics from University of Science and Technology in 2009, and the M.S. degree in system science from National University of Defense and Technology in 2011. He is currently pursuing the Ph.D. degree in applied mathematics, National University of Defense and Technology. His research interests include computer vision, pattern recognition and machine learning.

Cong Fang received the bachelor’s degree in electronic Science and technology (for optoelectronic technology) from Tianjin University in 2014. He is currently pursuing the Ph.D. degree with the School of Electronics Engineering and Computer Science, Peking University. His research interests include computer vision, pattern recognition, machine learning and optimization.

Zhouchen Lin received the PhD degree in applied mathematics from Peking University in 2000. Currently, he is a professor at the Key Laboratory of Machine Perception (MOE), School of Electronics Engineering and Computer Science, Peking University. He is also a chair professor at Northeast Normal University. He was also a guest researcher at the Institute of Computing Technology, Chinese Academic of Sciences. His research interests include computer vision, image processing, machine learning, pattern recognition, and numerical optimization. He is an associate editor of IEEE T. Pattern Analysis and Machine Intelligence and International J. Computer Vision and a senior member of the IEEE.

Yi Wu is a professor in the Department of Mathematics and System Science at the National University of Defense Technology in Changsha, China. He earned bachelor’s and master’s degrees in applied mathematics at the National University of Defense Technology in 1981 and 1988. He worked as a visiting researcher at New York State University in 1999. His research interests include applied mathematics, statistics, and data processing.


