www.ietdl.org

Published in IET Computer Vision
Received on 19th January 2014
Revised on 20th May 2014
Accepted on 21st May 2014
doi: 10.1049/iet-cvi.2014.0022

ISSN 1751-9632

Scene text recognition by learning co-occurrence of strokes based on spatiality embedded dictionary

Song Gao, Chunheng Wang, Baihua Xiao, Cunzhao Shi, Wen Zhou, Zhong Zhang

The State Key Laboratory of Management and Control for Complex Systems, Institute of Automation Chinese Academy of Sciences, Beijing, People's Republic of China
E-mail: chunheng.wang@ia.ac.cn

Abstract: Text information contained in scene images is very helpful for high-level image understanding. In this study, the authors propose to learn co-occurrence of local strokes for scene text recognition by using a spatiality embedded dictionary (SED). Unlike spatial pyramid partitioning images into grids to incorporate spatial information, the authors SED associates every codeword with a particular response region and introduces more precise spatial information for robust character recognition. After localised soft coding and max pooling of the first layer, a sparse dictionary is learned to model co-occurrence of several local strokes, which further improves classification performance. Experimental results on two scene character recognition datasets ICDAR2003 and CHARS74 K demonstrate that their character recognition method outperforms state-of-the-art methods. Besides, competitive word recognition results are also reported for four benchmark word recognition datasets ICDAR2003, ICDAR2011, ICDAR2013 and street view text when combining their character recognition method with a conditional random field language model.

1 Introduction

A robust scene text extraction system can be used in lots of areas such as image retrieval, intelligent transportation, robot vision etc. To obtain text information from scene images, two stages are usually included: text detection and text recognition. Text regions are localised from the whole scene images in text detection stage. In text recognition stage, definite text information is generated from the cropped text blocks. In the past years, many efficient systems have been proposed by researchers to detect scene texts, whereas scene text recognition has not been fully studied. In this paper, we focus on the scene text recognition and perform two different tasks, namely scene character recognition and scene word recognition.

Most scene text recognition techniques could be divided into two categories: optical character recognition (OCR)-based methods and object recognition-based methods.

OCR-based methods rely on the off-the-shelf OCR techniques which have been highly developed in the past decades. These methods usually focus on scene text binarisation. For example, Chen and Yuille [1] use Adaboost to detect texts and extend Niblack's binarisation method before feeding binarised texts into OCR engine. Neumann and Matas [2] use maximally stable extremal regions (MSER) extraction to perform scene texts binarisation. Traditional OCR techniques are designed for scanned documents which are usually easy to binarise. However, scene text binarisation is difficult because of complex backgrounds, heavy occlusions and different lighting conditions.

Object recognition-based methods [3–5] skip the binarisation stage and each kind of scene character is regarded as a special object. These methods firstly extract features from one image patch which is considered as containing a single character. Then the features are fed into various pre-trained classifiers to obtain a label. Wang *et al.* [3] choose random fern as the basic classifier and utilise pictorial structures to recognise words. Mishra *et al.* [4] train multi-class support vector machines (SVMs) to recognise scene characters and use conditional random field (CRF) to generate word recognition results. In our previous work [5], we use part-based tree structure to detect characters and also combine with a CRF to perform words recognition. Recently, object recognition-based methods are inspiring more and more enthusiasm from the computer vision community for their simplicity, efficiency and robustness. Thus, in this paper, we adopt an object recognition-based method. Especially, we introduce the popular coding/pooling scheme for scene text recognition.

Coding/pooling pipeline has been quite successful for object recognition in recent years. Various coding methods are proposed, including nearest neighbour vector quantisation, soft assignment [6], localised soft coding [7] and sparse coding [8]. As for pooling, average and max pooling are frequently used. Boureau *et al.* compare different coding and pooling methods thoroughly in [9, 10]. When incorporating spatial information into coding/pooling scheme, spatial pyramid (SP) [11] and its variants [12] have been the predominant approaches. SP usually partitions one image into a set of regions beforehand and then codes them

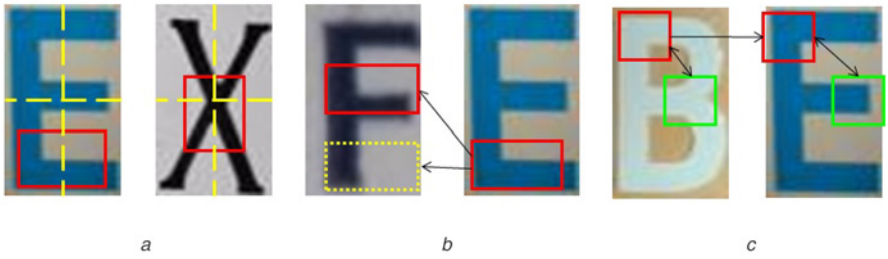

Fig. 1 *Motivation*

a Rough division (lines) in SP separates discriminative strokes (rectangles)
b Discriminative part to tell 'E' from 'F' appears in another location of 'F' which may bring classification confusion if spatial information is ignored
c Strokes in dark grey/red rectangles cannot tell 'B' from 'E', but introducing co-occurrence of local strokes (plus light grey/green rectangles) may solve this problem

separately before concatenating the code vectors. However, as for scene text character recognition, sizes of most character images are usually very small so that image patches of SP may not be able to provide more discriminative information of character structures. Besides, rough regions division in SP may lose the power of discriminative stroke structures which are also divided as shown in Fig. 1*a*. Dropping SP and ignoring spatial information as in [13] can bring about character classification confusion. That is because one part of a character may appear in another location of another character as in Fig. 1*b*. Therefore when using coding/pooling scheme for scene character recognition, it is necessary to find a way to incorporate spatial information beyond SP.

Therefore, to overcome the drawbacks of SP for scene character recognition, we propose to build a new type of dictionary called spatiality embedded dictionary (SED) to include more precise spatial information than SP. In SED, each codeword represents a particular character stroke in a special location. A sparse dictionary is further learned based on SED to model co-occurrence of several local strokes for robust character recognition as shown in Fig. 1*c*. Generally, we make three contributions:

(1) We propose a new type of dictionary called spatial embedded dictionary to include more precise spatial information than SP for scene character recognition.
(2) Based on SED, coding can be performed locally spatially, which alleviates computation burden and retains discrimination power at the same time.
(3) After first-layer coding and pooling, a sparse dictionary is learned to model co-occurrence of several local strokes to further improve classification performance.

The proposed mechanism has achieved 82.7% on ICDAR2003 scene character recognition dataset and 67.5% on CHARS74 K dataset which outperforms state-of-the-art methods. When the proposed method is combined with sliding window technique and a CRF word recognition model, our character recognition method can be applied to word recognition. For word recognition, whole-word recognition accuracy is used as evaluation criteria rather than single character classification accuracy. We report competitive results on four benchmark word recognition datasets ICDAR2003, ICDAR2011, ICDAR2013 and street view text (SVT).

This paper is organised as follows. Section 2 introduces the related work of coding/pooling pipeline. Section 3 outlines framework of the proposed method. Details of the proposed method are presented in Section 4. Afterwards, experimental results are given in Section 5. Finally, conclusions are drawn in Section 6.

2 Related work

The bag-of-words model has been quite popular for image categorisation in recent years. This representation quantises continuous high-level image features to a histogram of 'visual words'. The method usually consists of two parts: coding and pooling. In this section, we mainly introduce the commonly used coding and pooling methods. For coding, methods like nearest neighbour vector quantisation, soft assignment, localised soft assignment, sparse coding, locality-constrained linear coding (LLC) are explained. For pooling methods, average pooling and max pooling are introduced. Besides, how SP incorporates spatial information for the bag-of-words model is also illustrated.

To illustrate more clearly, notations are introduced as follows. Let d_i $(d_i \in R^{n_d})$ denote a visual codeword where n_d is the dimensionality of one codeword. Assuming there are n codewords in the dictionary D $(D \in R^{n_d * n})$, then D is represented as $D = \{d_1, d_2, \ldots, d_n\}$. Let ϕ_i $(\phi \in R^{n_d})$ be the ith local descriptor of an image. Let U_i $(U_i \in R^n)$ be the coding coefficient vector of ϕ_i, with u_{ij} being the coefficient with respect to codeword d_j.

For nearest neighbour vector quantisation, there is only one non-zero coding coefficient for descriptor ϕ_i. This non-zero coefficient corresponds to the codeword which is nearest to the descriptor according to a distance metric. In practice, Euclidean distance is usually used as follows

$$u_{ij} = \begin{cases} 1, & j = \arg\min_a \|\phi - d_a\|, \ d_a \in D \\ 0, & \text{otherwise} \end{cases} \quad (1)$$

In soft assignment [6], the jth coding efficient for descriptor ϕ_i represents the relative possibility of the descriptor assigned to the jth codeword subject to all other codewords as follows

$$u_{ij} = \frac{\exp\left(-\beta \left\|\phi - d_j\right\|^2\right)}{\sum_{a=1}^{n} \exp\left(-\beta \left\|\phi - d_a\right\|^2\right)} \quad (2)$$

where β is the parameter to control the softness of assignment.

Soft assignment is further improved by Liu *et al.* [7] and they propose localised soft assignment. Rather than calculating the relative possibility of the descriptor assigned to the jth codeword subject to all other codewords, a local neighbourhood around the descriptor is defined first. Then the relative possibility is calculated in the local neighbourhood. Let $NN_K(\phi_i)$ be the set of K nearest neighbours for descriptor ϕ_i in dictionary D. Then the

localised soft coding is defined as follows

$$u_{ij} = \frac{\exp(-\beta \times \mathrm{dis}(\phi_i, d_j))}{\sum_{a=1}^{K} \exp(-\beta \times \mathrm{dis}(\phi_i, d_a))} \tag{3}$$

where

$$\mathrm{dis}(\phi_i, d_j) = \begin{cases} \|\phi_i - d_j\|^2, & \text{if } d_j \in NN_K(\phi_i) \\ \infty, & \text{otherwise} \end{cases}$$

Sparse coding [8] represents a local descriptor ϕ_i by a linear combination of a sparse set of basis vectors. The coefficient vector $\boldsymbol{U}_i$ is obtained by solving an l_1-norm regularised approximation problem

$$\boldsymbol{U}_i = \arg\min_{U} \|x_i - DU_i\|_2^2 + \lambda \|U_i\|_1 \tag{4}$$

LLC [14] is similar to sparse coding. The difference is that LLC firstly introduces a penalty term to select accountable codewords nearby the descriptor ϕ_i in Euclidean space. Then the descriptor ϕ_i is represented with a linear combination of selected codewords.

After all local features of an image are coded, a pooling operation is further performed, such as average pooling and max pooling. With average pooling, the jth component of the pooled vector is calculated as $\left(\sum_{i=1}^{m} u_{ij}\right)/m$, where m is the number of local features. As for max pooling, the jth component of the pooled vector is calculated as $\max_i u_{ij}$, where $i = 1, 2, \ldots, m$.

SP [11] incorporates spatial information by partitioning one image into several local regions. Then coding and pooling are conducted within every local region. After pooling, all these pooled vectors of different local regions are connected sequentially to form a large vector. This large vector is regarded as the final image representation. However, as stated in the introduction section, SP is not suitable for scene character recognition because scene character images are usually too small to be partitioned. Thus, it is appealing to find an appropriate way to model spatial layout beyond SP for scene text recognition.

3 Framework

The proposed system consists of five parts: (i) building SED; (ii) coding and pooling based on SED; (iii) learning a sparse dictionary to model co-occurrence of strokes; (iv) training character detectors; and (v) word recognition based on a CRF language model. The overall framework for scene word recognition is given in Fig. 2. For scene character recognition, language model is ignored and trained character detectors are used to classify testing images directly.

4 Proposed method

4.1 Building SED

4.1.1 Overview of SED: Instead of using K-means to cluster all descriptors regardless of their positions as before, SED performs codeword collection considering descriptors' positions. In SED, spatial information is incorporated into dictionary directly by reserving a local response region for every codeword. Thus, SED can include more precise spatial information than SP, which partitions images into local regions and codes them sequentially. Based on SED, coding can be performed locally spatially depending on the spatial relationship between codeword and descriptor as shown in Fig. 3. That can alleviate computation burden and retain discrimination power at the same time. The procedure of how SED incorporates spatial information is given in Fig. 3.

The SED building process is illustrated as follows. For better clarity, we first give out a brief statement for every step before going into details:

(a) All character training images are scaled to the same size and partitioned into uniform blocks.

Every character training images are normalised to the same size height = H and width = W. Then every image is partitioned into n_h*n_w blocks (dotted lines in Fig. 4*a*), in which n_h is the number of partitioned blocks in vertical and n_w is the number of partitioned blocks in horizontal.

(b) HOG features [15] are extracted within every block and then connected to represent every character training image.

For every training image, histogram of gradients (HOG) features [15] with size of n_{hog} dimensions are extracted within every block and connected sequentially to form a one-dimensional (1D) feature vector as the overall

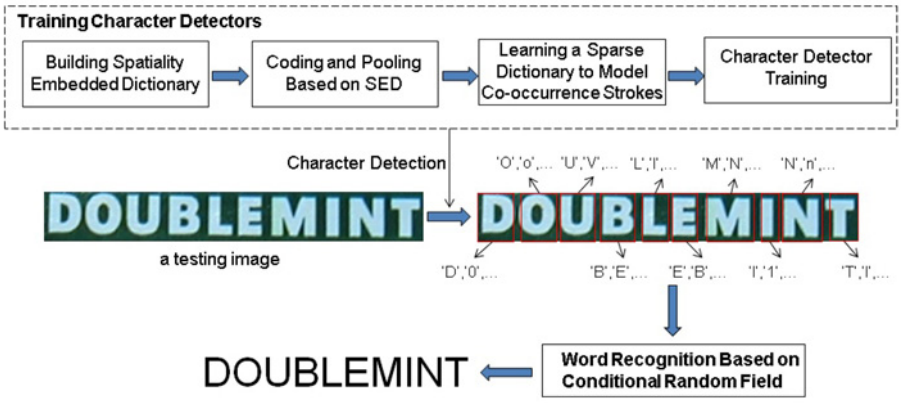

Fig. 2 *Framework of the proposed method for scene word recognition*

doi: 10.1049/iet-cvi.2014.0022

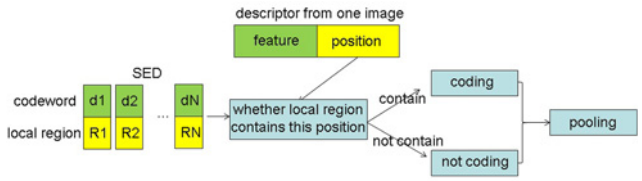

Fig. 3 *How SED incorporates spatial information? Whether to code one entry for one descriptor or not depends on the spatial relationship between codeword and descriptor*

Only when response region of one codeword covers the descriptor's position, the corresponding entry should be coded

representations. It should be noted the 1D vector has a dimensionality of $n_h * n_w * n_{\text{hog}}$.

(c) Within each character category, images are clustered based on the overall representations.

Assuming class c_i ($i \in \{1, 2, 3, \ldots, N_c\}$, N_c is the number of categories) has $n_{\text{train, c}_i}$ training images, K-means clustering is performed based on the above overall representations to obtain $n_{\text{train, c}_i}/k_{c_i}$ centres, where k_{c_i} is the parameter to control the number of clustering centres. For every clustering centre of class c_i, we reshape the overall representation 1D vector to 3D matrices with three dimension sizes of n_h, n_w and n_{hog}.

(d) Codewords are then extracted from the clustering centres sequentially.

We extract sub 3D matrices with three dimension sizes of $n_{h,\text{d}}$, $n_{w,\text{d}}$ and n_{hog} from the clustering centres densely (from left to right and from top to bottom), where $n_{h,\text{d}}$ is the number of blocks in vertical and $n_{w,\text{d}}$ is the number of blocks in horizontal. Extraction interval is set to be $n_{w,\text{d}}/2$ for horizontal and $n_{h,\text{d}}/2$ for vertical. The number of collected codewords from one clustering centre depends on the choice of $(n_{h,\text{d}}, n_{w,\text{d}})$. These sub 3D matrices (solid rectangle in Fig. 4*a*) are then stretched to 1D vectors with sizes of $n_{h,\text{d}} * n_{w,\text{d}} * n_{\text{hog}}$ dimensions. These 1D vectors are regarded as our codewords d_j.

(e) For every codeword d_j, we record a response region r_j, which is around the codeword sampling position.

To illustrate the definition of response region clearly, X–Y coordinate is introduced and coordinates of top left corner (green point in Fig. 4*a*) is regarded as coordinates of one codeword. Then response region for codeword d_j can be represented by coordinates of the points (magenta points in Fig. 4*a*) around the top left corner as r_j. It should be noted that the codeword sampling position itself should also be included in r_j. Actually, a response region (rectangle with dotted line in Fig. 4*a*) can be regarded as the area covered by patches nearby the codewords. These nearby patches, which have codeword's size, use points from r_j as their top left corners. We include $l * l$ points in r_j, in which l is the side length of response points square as illustrated in Fig. 4*a*.

(f) We combine all codewords into the SED $D_{\text{SED}} = \left\{(d_1, r_1), (d_2, r_2), (d_3, r_3), \ldots, \left(d_{N_{D_{\text{SED}}}}, r_{N_{D_{\text{SED}}}}\right)\right\}$, in which $N_{D_{\text{SED}}}$ is the dictionary size and $r_j \left(j = 1, 2, \ldots, N_{D_{\text{SED}}}\right)$ is the corresponding response region for codeword d_j.

Assuming N_{train} images are contained in the training set, N_{one} codewords are collected from one clustering centre, and k_{c_i} is set to be k uniformly for all c_i, then the number of codewords in the final SED is $N_{D_{\text{SED}}} = N_{\text{train}} \times N_{\text{one}} \times (1/k)$. The procedure of generating one codeword is given in Fig. 4*a*.

4.2 Coding and pooling

Based on SED, coding can be performed locally spatially according to codewords' reserved response regions, which alleviates computation burden and retains discrimination power at the same time.

Given a local descriptor ϕ extracted from position (x_ϕ, y_ϕ) of one image, only entries, whose corresponding codewords' response regions r_j contain point (x_ϕ, y_ϕ), should be coded. The other entries are set to be zeros directly thus reducing computation time. Localised soft assignment [7] is chosen for its effectiveness and efficiency. Assuming the accountable codewords [whose response regions cover point

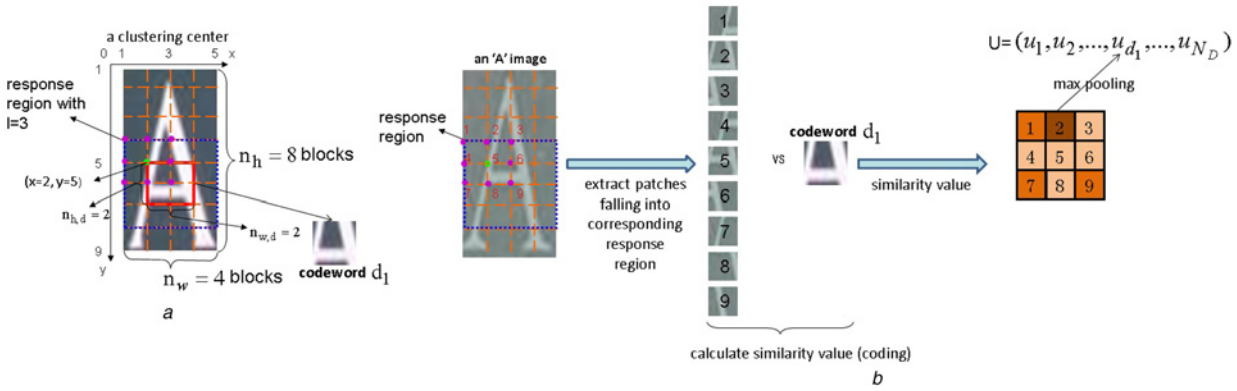

Fig. 4 *Illustration of SED*

a Partitioned blocks (dotted lines), codeword (solid rectangle), sampling location (point) and response region (dotted rectangle) are shown on a clustering centre. The dimension of codeword d_1 is $2 * 2 * n_{\text{hog}}$. We give out a visible image in this figure but clustering centres are virtual 3D matrices actually

b Physical explanation of SED coding and pooling from the view of codeword, note that darker patch means higher similarity between codeword and descriptor. For better viewing, please see the online version

(x_ϕ, y_ϕ)] set is $D_\phi = \{d_{\phi,1}, d_{\phi,2}, \ldots, d_{\phi,n_\phi}\} \subset D_{\text{SED}}$ for ϕ, localised soft coding is incorporated with spatial information and modified as (5). $NN_{(K)}(\phi)$ is the K nearest neighbours of D_ϕ in Euclidean space for descriptor ϕ. K is set to be 5 and β is set to be 0.1 in the experiments.

After coding, max pooling is performed over the whole image to obtain code vector $\boldsymbol{U}$. Max pooling is chosen rather than average pooling because of the implied physical meaning of coding. For every descriptor, each entry u_j represents the possibility of codeword d_j appearing in the descriptor's position (also around codeword d_j original collecting position). Actually, codeword d_j corresponds to a special character stroke. Prior knowledge is that one stroke structure d_j always appears only once around one position of a character image so it is reasonable to consider only the most likely appearing location.

The coding and pooling pipeline can be reconsidered from the view of codeword rather than the view of descriptor. For example, in Fig. 4*b*, in order to calculate the d_1th entry for one 'A' image, image patches which fall into response region of d_1 are extracted. Then similarity values between extracted image patches and d_1 are calculated (coding). Finally, the maximal similarity value is reserved, which represents the possibility of d_1 appearing around the sampling position on this new image.

4.3 Learning a sparse dictionary to model co-occurrence of strokes

4.3.1 Motivation: If code vectors $\boldsymbol{U}$ are directly fed into multi-class linear SVMs [16] for training, we are only able to model single stroke appearing in one position not co-occurrence of several strokes in different locations. Selecting non-linear SVMs rather than linear SVMs may seem to be able to model co-occurrence of strokes. However, we find in experiments choosing non-linear SVMs works even worst. It is perhaps because non-linear SVMs can introduce overfitting more easily than linear SVMs. Besides, classification based on non-linear SVMs is time-consuming. Intuitively, modelling co-occurrence of different stroke structures can introduce high-level semantic information and may restrain noise. Therefore it is appealing to find an effective way to model co-occurrence of strokes.

To model co-occurrence of several strokes, we propose to learn a sparse dictionary $D_{\text{SPR}} = \left[d_{s,1}, d_{s,2}, \ldots, d_{s,N_{D_{\text{SPR}}}}\right] \in R^{N_{D_{\text{SED}}} \times N_{D_{\text{SPR}}}}$ based on the code vectors $\boldsymbol{U}$, where $N_{D_{\text{SED}}}$ is the size of SED D_{SED} and $N_{D_{\text{SPR}}}$ are the number of atoms contained in sparse dictionary D_{SPR}. We use elastic net [17] as in [18] to learn D_{SPR}. The learned dictionary D_{SPR} should be sparse. It means most entries of $d_{s,i} \in R^{N_{D_{\text{SED}}}}$ are zeros. The non-zero entries of $d_{s,i}$ correspond to co-occurrence of some strokes, which will be proved in the experiment section. Thus, reconstruction coefficient w for code vector $\boldsymbol{U}$ based on D_{SPR} can represent appearance of co-occurrence of strokes for one character image.

4.3.2 Learning method: Given a set of training images represented as code vectors $\boldsymbol{U}_{\text{set}} = \{U_1, U_2, U_3, \ldots, U_i, \ldots, U_{N_{\text{train}}}\}$ ($U_i \in R^{N_{D_{\text{SED}}}}$ and N_{train} is the number of training images), sparse reconstruction coefficient w_i is learned simultaneously when learning sparse dictionary D_{SPR} as in (6). Then given an image represented as code vector $\boldsymbol{U}$, sparse coefficient w is computed so that $\boldsymbol{U}$ can be reconstructed from D_{SPR} like (7)

$$\min_{D_{\text{SPR}} \in C, W \in R} \sum_{i=1}^{N} \left(\frac{1}{2} \left\| \boldsymbol{U}_i - D_{\text{SPR}} w_i \right\|_2^2 + \lambda_w \left\| w_i \right\|_1 \right) \quad (6)$$

$$\min_{w \in R^{N_{D_{\text{SPR}}}}} \frac{1}{2} \left\| \boldsymbol{U} - D_{\text{SPR}} w \right\|_2^2 + \lambda_w \left\| w \right\|_1 \quad (7)$$

where $W = [w_1, w_2, \ldots, w_{N_{\text{train}}}] \in R^{N_{D_{\text{SPR}}} \times N_{\text{train}}}$, λ_w is a regularisation parameter and C is the convex set which D_{SPR} belongs to. The convex set C can be constructed as follows

$$C = \left\{ D_{\text{SPR}} \in R^{N_{D_{\text{SED}}} \times N_{D_{\text{SPR}}}}, \text{ s.t. } \forall i, \left\| d_{s,i} \right\|_1 + \frac{\gamma}{2} \left\| d_{s,i} \right\|_2^2 \leq 1 \right\} \quad (8)$$

The sparsity requirement of dictionary D_{SPR} is achieved by enforcing l_1-norm and l_2-norm on convex set C. This is called the elastic-net [17]. In the two optimisation problems, (7) is convex and (6) is convex with respect to each of the two variables D_{SPR} and W when the other one is fixed. We use SPASM toolbox [19] to solve the two optimisation problems. In the experiments, λ_w in (6) and (7) is set to be 0.1 and γ in (8) is set to be 0.3 empirically.

4.4 Training character detectors

Coefficients W computed from character training samples in (6) are used to train multi-class linear SVMs [16]. The regularisation parameter is set to the best by cross-validation on the training set.

In the test stage of character recognition, given a testing image (scaled to $H * W$ already), code vector $\boldsymbol{U}$ is calculated using localised soft coding and max pooling based on D_{SED} as in Section 4.2. Afterwards, w are calculated using D_{SPR} as in (7). Finally, w is fed into pre-trained multi-class SVMs to obtain a class label.

4.5 Word recognition model using CRF

Given a scene word image, the character recognition block provides us lots of windows containing characters within them, but at the same time it also produces lots of false positives. In this section, our aim is to generate whole-word recognition results from piles of detection windows. Thus, we build a CRF model as in [4, 5] to combine character detection results and linguistic models. Then word recognition can be performed by solving an energy minimisation problem.

$$u_j = \begin{cases} \dfrac{\exp\left(-\beta \left\| \phi - d_j \right\|^2\right)}{\sum_{a=1}^{K} \exp\left(-\beta \left\| \phi - d_a \right\|^2\right)}, & d_j \in D_\phi \text{ and } d_a \in NN_{(K)}(\phi) \\ 0, & \text{otherwise} \end{cases} \quad (5)$$

doi: 10.1049/iet-cvi.2014.0022

In general, the word recognition procedure is as follows: (i) we first perform character detection based on co-occurrence of strokes using SED at multi-scales; (ii) then we use these detection windows to decide the potential character locations on which the CRF model is defined; (iii) compute the unary and pairwise cost function based on the detection scores, the spatial constraints and the language model; and (iv) finally infer the most likely word using the tree-reweighted passing (TRW-S) algorithm. Details about the CRF language model are given below.

4.5.1 Word model: In a scene text image, there are several potential character positions, which usually have several character detection candidates. Each position is represented by a random variable X_i. Let n be the total number of potential locations. Since some potential locations might have no characters, a non-character label ϵ is introduced to represent these false positives. Thus, each random variable X_i takes a label $x_i \in C_\epsilon = C \cup \{\epsilon\}$. We use C_ε^n to represent the set of all possible labelling assignments to the random variables. Then we define a cost function $E{:}C_\epsilon^n \rightarrow \mathbb{R}$, to map any labelling to a real number $E(\cdot)$ which represents energy. $E(\cdot)$ is defined as a sum of unary and pairwise terms as follows

$$E(\boldsymbol{x}) = \sum_{i=1}^{n} E_i(x_i) + \lambda_{\mathrm{p}} \sum_{\{i,j\} \in N_g} E_{ij}(x_i, x_j) \tag{9}$$

where $x = \{x_1, x_2, \ldots, x_n\}$ represents the set of all the random variables, $E_i(x_i)$ is the unary cost function, $E_{ij}(x_i, x_j)$ denotes the pairwise cost and Ng represents the set of pairs of neighbouring detection windows. Ng is determined by the structure of the graphical model defined on them. λ_{p} is a tradeoff parameter between the unary and pairwise cost and is set to be 0.6 by cross-validation.

4.5.2 Graph construction: After applying non-maximal suppression (NMS) on the original character detection results, the left detection windows constitute the potential locations. We set the overlap parameter for NMS to 0.35 in the experiment. Then, for each location, we choose those detection windows which are close to this location as the candidate characters for this location. We add one node for each potential location sequentially from left to right. The nodes are connected by edges. Since nodes which are spatially distant from each other would not be directly related, we only connect nodes which are close to each other. Fig. 5 shows the process.

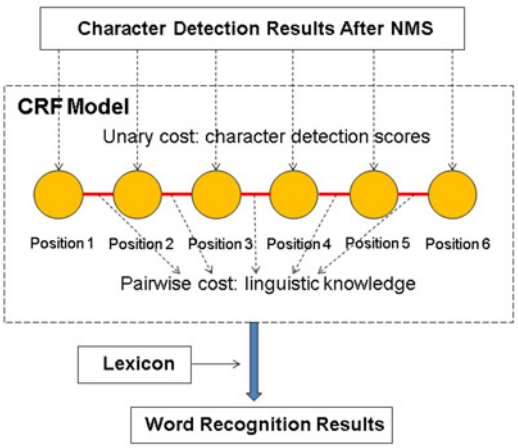

Fig. 5 *Word recognition using CRF model*

4.5.3 Cost function: The unary cost $E_i(x_i = c_j)$ represents the penalty of assigning label c_j to node x_i. In this case, if the detection score for a certain type of character model c_j is very high, the cost of labelling the node c_j should be small and vice versa. If the scores of all the candidate detections are very low, it is likely for the node to take a null label ε. To this end, we define the unary cost as follows

$$E_i(x_i = c_j) = \begin{cases} 1 - p(c_j|x_i), & \text{if } c_j \neq \epsilon \\ \max_j p(c_j|x_i), & \text{otherwise} \end{cases} \tag{10}$$

where $p(c_j|x_i)$ is the probability for node x_i to take label c_j. Here we use the detection scores (SVM score) to reflect the confidence for the class. For a true window, the cost of assigning a null label is high. On the other hand, false positive windows with poor SVM scores are more likely to take the null label ϵ.

We use the pairwise cost function $E(x_i, x_j)$ to incorporate linguistic knowledge and spatial constraints. The pairwise cost of two neighbouring nodes (x_i, x_j) taking labels (c_i, c_j) is defined as

$$E_{ij}(x_i, x_j) = \begin{cases} 1 - P(c_i, c_j), & \text{if } c_i \neq \epsilon \wedge c_j \neq \epsilon \\ D_{ij} + \mu S_i, & \text{if } c_i = \epsilon \wedge c_j \neq \epsilon \\ D_{ij} + \mu S_j, & \text{if } c_i \neq \epsilon \wedge c_j = \epsilon \\ D_{ij} + \mu S_{i,j}, & \text{if } c_i = \epsilon \wedge c_j = \epsilon \end{cases} \tag{11}$$

where $P(c_i, c_j)$ refers to the bi-gram language model learnt from the lexicon, D_{ij} is the relative distance of the two nodes, S_i and S_j represent the maximum character detection scores at the corresponding locations, $S_{i,j}$ is the larger one of S_i and S_j and μ is set to be 1.5 in the experiment. We use the SRI language modelling toolkit [20] to learn the probability of joint occurrence of characters in large English dictionary with around 0.5 million words provided by Mishra *et al.* [4]. The pairwise cost function means that if the probability of joint occurrence of a character pair (c_i, c_j) is large, the cost of nodes (x_i, x_j) taking labels (c_i, c_j) should be small. Moreover, if the relative distance of the two nodes is small, and the maximum score of the node is low, the cost of the node taking a null label should be small.

4.5.4 Inference: After computing the unary and pairwise cost, we use the sequential TRW-S algorithm [21] to minimise the cost function in (9), because of its efficiency and accuracy on our recognition problem. The TRW-S algorithm maximises a concave lower bound on the energy. It begins by considering a set of trees from the random field and computes probability distributions over each tree, which are then used to reweight the messages being passed during loopy back propagation (BP) [22] on each tree. The algorithm terminates when the lower bound cannot be increased further or the maximum number of iterations has reached.

5 Experiment

5.1 Datasets and tasks

5.1.1 Character recognition: We employ two public scene character datasets: ICDAR2003 [23] and CHARS74 K [24]. Both of these two datasets contain 62 character classes, namely digits 0–9, upper English letters A–Z and lower

Fig. 6 *Some scene character images from ICDAR2003 testing set*

English letters a–z. All character samples are extracted from scene images manually by datasets organisers. ICDAR2003 dataset contains 6185 training patches and 5430 testing patches cropped from 509 scene images. It is originally designed for robust reading competition of scene text detection and recognition so the contained samples can cover different conditions in natural scenes, such as heavy occlusions, different illuminations and complex backgrounds. Some scene character images from ICDAR2003 testing set are shown in Fig. 6. Similarly, CHARS74 K dataset has totally 12 503 scene character images cropped from various natural scenes and these images are not split into training and testing datasets.

For ICDAR2003 dataset, we use image patches from its own training set to train character classifiers and then test these classifiers on images from testing set. In CHARS74 K dataset, we perform CHARS74 K-15 evaluation and split training and testing sets referring to [25]. For both datasets, we use character recognition accuracy to evaluate performance as in [3, 13, 24, 26] for fair comparison.

5.1.2 Word recognition: We use the challenging public datasets SVT [27], ICDAR2003 robust word recognition datasets [23] and ICDAR2011 word recognition datasets [28] to evaluate the performance of the overall word recognition method. The SVT dataset contains images taken from Google View Street. Since we focus on the word recognition task, we use the SVT-WORD dataset following the experiment protocol of [3, 27]. Totally, we obtain 647

Fig. 7 *Some scene word images from ICDAR2003 dataset*

Fig. 8 *Most discriminative strokes (in rectangles) for every character can be localised according to the largest weights of linear SVMs. ($n_{h,d}$, $n_{w,d}$) is set to be (5,5)*

word images in SVT dataset. For ICDAR2003 dataset, we ignore words with less than two characters or with non-alphanumeric characters as in [3], which results in 829 word images overall. Some scene word images from ICDAR2003 dataset are shown in Fig. 7. We use ICDAR2011 and ICDAR2013 datasets in the same way as ICDAR2003 dataset.

For word recognition, we use whole-word recognition accuracy to evaluate the performance. The proposed word recognition method is performed with or without lexicons. Word recognition with lexicons is known as word spotting [27], which has been drawing lots of attention from the research community in recent years [3–5]. It should be noted character recognition plays a very important role in word recognition, which means word recognition accuracy heavily relies on character recognition accuracy. When sliding window technique is used as in this paper, character recognition is also referred as character detection.

5.2 Character recognition

5.2.1 Settings: All of the image patches are normalised to $W = 32$ and $H = 64$ and partitioned into blocks with $n_w = 8$ and $n_h = 16$. HOG features [15] are extracted within every block with bin number 9, cell size of 2×2 pixels, block size of 2×2 cells. $n_{h,\mathrm{d}}$, $n_{w,\mathrm{d}}$ are set to be one of 5, 6, 7.

Larger k can generate bigger dictionary which may be beneficial for classification, but at the same time it may result in heavier computation burden. According to our experiments, when k rises up to a peak point for both

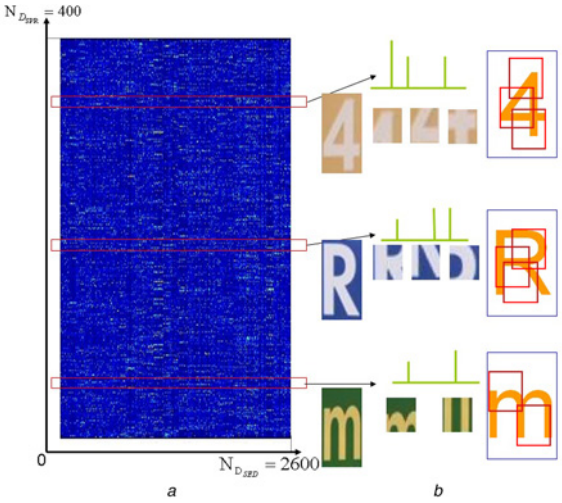

Fig. 9 *Learned sparse dictionary D_{SPR}^{T} with size of 400 for ICDAR2003 when $n_{h,d}$, $n_{w,d}$ = (6, 5)*

a $D_{\mathrm{SPR}}^{\mathrm{T}}$, red points indicate large magnitudes, whereas blue colour represents small magnitudes

b Large non-zero entries of each row often correspond to co-occurrence of several strokes from the same character

See online version for colour

doi: 10.1049/iet-cvi.2014.0022

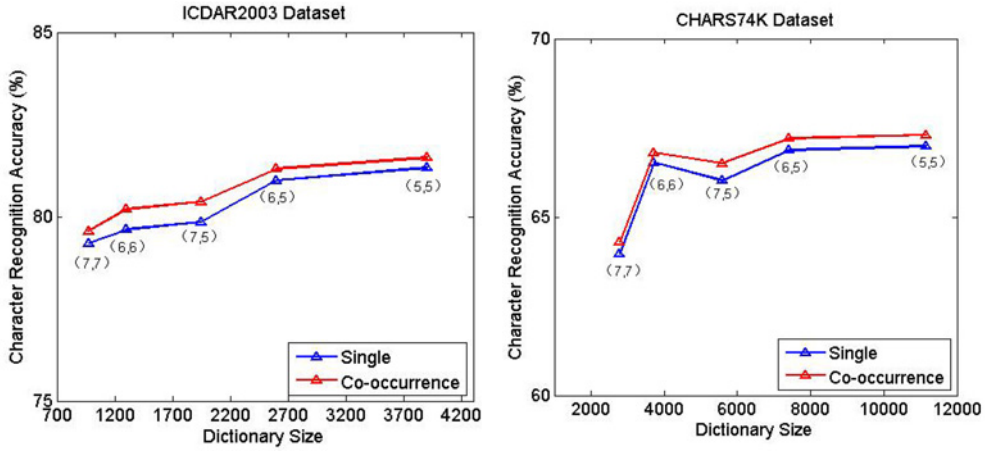

Fig. 10 *Superiority of co-occurrence of strokes over single stroke: left figure is for ICDAR2003 dataset and right figure is for CHARS74 K. Choices of* $(n_{h,d}, n_{w,d})$ *are labelled in both figures*

datasets, the classification accuracy seems to be still. To balance classification performance and computation time, k is set to be 20 for ICDAR2003 dataset and 1 for CHARS74 K dataset. For parameter l, it is ideal to set different l for different codewords as position range of different strokes may be various intuitively. However, it is difficult and labour-intensive to identify different l for different codewords. Therefore in the experiments, l is set to be 3 for both datasets directly and empirically.

When building SED, different sizes of codewords including $(n_{h,d}, n_{w,d}) = (7, 7), (6, 6), (7, 5), (6, 5), (5, 5)$ are chosen to build different sizes of dictionaries both for ICDAR2003 and CHARS74 K datasets. As for parameter $N_{D_{SPR}}$, we find its value does not have big influence on the classification performance in a range. In the experiments, $N_{D_{SPR}}$ is set to be 400 for both ICDAR2003 and CHARS74 K empirically as in [18].

5.2.2 Results and discussion: If we use SED code vectors $\boldsymbol{U}$ to train multi-class linear SVMs directly, the classification performance is shown in Fig. 10. Left figure is for ICDAR2003 dataset and right figure is for CHARS74 K dataset. It can be seen that bigger SED brings about better performance perhaps because bigger SED includes more discriminative character strokes. However when SED size reaches a level, classification performance seems to be still. When $(n_{h,d}, n_{w,d})$ is set to be (5, 5), the most discriminative codewords for every character can be localised according to the largest weights of trained SVMs. These discriminative codewords correspond to character discriminative strokes as shown in Fig. 8. It can be seen from Fig. 8 that these strokes are intuitively discriminative for character classification.

When for $(n_{h,d}, n_{w,d})$ is set to be (6, 5) and co-occurrence of different strokes is incorporated, the learned sparse dictionary D_{SPR} from ICDAR2003 training set with size of 400 is shown in Fig. 9. It can be seen that D_{SPR} is very sparse as desired. Large non-zero entries of rows in D_{SPR}^{T} often correspond to different strokes of one kind character and zero entries refer to irrelevant strokes. Owing to the implied high-level semantic information contained in D_{SPR}, reconstruction coefficient w based on D_{SPR} in (7) can represent co-occurrence of strokes on an image. Thus, if reconstruction coefficient w is used to train multi-class linear SVMs, we are able to introduce co-occurrence of strokes for scene character recognition as stated in Section 4.3.

Superiority of modelling co-occurrence of strokes over using SED for direct classification is shown in Fig. 10. It should be noted that sizes of SED depend on codewords' size $(n_{h,d}, n_{w,d})$ as stated in step (f) of Section 4.1. Sizes of all sparse dictionaries D_{SPR} are set to be 400. From Fig. 10, we can see that modelling co-occurrence of strokes by learning a sparse dictionary always works better than only modelling single stroke's appearance based on SED. That proves the usefulness of high-level semantic information. With the growth of SED size in a range, the performance of modelling co-occurrence of strokes becomes better. That is probably because larger SED contains more potential strokes which may co-occur.

5.2.3 Comparison with other algorithms: In recent years, researchers have mainly focused their attention on feature representation in the field of scene character recognition. For instance, Campos *et al.* [24] compare different features for scene character recognition. Newell and Griffin [29] propose two extensions of the HOG descriptor to include features at multiple scales and demonstrate superiority of these new features over HOG for robust character recognition. Coates *et al.* [26] introduce unsupervised feature learning using a variant K-means clustering and report promising character recognition results. Yi *et al.* [13] generate global HOG (GHOG) by computing HOG descriptor from global sampling for scene character recognition and they experimentally prove superiority of GHOG over coding/pooling method. However, in [13], they do not find a way to incorporate spatial information into the coding/pooling pipeline.

To include strokes of different aspect ratios meanwhile, different sizes of codewords $[(n_{h,d}, n_{w,d}) = (7, 7), (6, 6), (7, 5), (6, 5), (5, 5)]$ are incorporated in SED for both ICDAR2003 and CHARS74 K datasets. A sparse dictionary D_{SPR} with size of 400 is further learned for both datasets. The classification performance of our system is shown in Table 1. Experimental results outperform state-of-the-art methods. Especially for ICDAR2003 dataset, we only use training samples from ICDAR2003 training set rather than introducing other training samples to avoid overfitting as in [26, 30].

It can also be seen from Table 1 that by incorporating co-occurrence of strokes rather than simply using single stroke, 0.7 and 0.4% classification improvement are realised on ICDAR2003 and CHARS74 K, respectively. The

Table 1 Character recognition results on ICDAR2003 and CHARS74 K datasets, %

Algorithms	ICDAR2003	CHARS74 K-15
HOG + NN [3]	51.5	58
SYNTH + FERNS [3]	52	47
NATIVE + FERNS [3]	64	54
MSER [31]	67	–
global HOG [13]	76	62
co-HOG [30]	79.4	–
coates method [26]	81.7	–
geometrical blur + SVM [24]	–	53
multiple Kernel learning [24]	–	55
HOG columns [29]	–	66.5
our method (single stroke)	82.0	67.1
our method (co-occurrence of strokes)	82.7	67.5

improvements are not very significant perhaps because representation based on our SED is already strong. Compared with [13] which uses HOG and non-linear SVMs, we use more simpler linear SVMs and obtain inspiring 6.7% improvement on ICDAR2003 testing dataset, which demonstrates the representation power of co-occurrence of strokes based on SED. However, performance of our system on CHARS74 K dataset is still not satisfying perhaps because of the large font variations.

Generally, when recognising samples from ICDAR2003 testing dataset, the proposed method takes about 0.3 s on average to classify a character image on personal computer with Intel (R) Core (TM) i5-3210M central processing unit 2.50 GHz if ($n_{h,\mathrm{d}}$, $n_{w,\mathrm{d}}$) is set to be (7, 7) ($N_{D_{\mathrm{SED}}}$ is equal to 795) and $N_{D_{\mathrm{SPR}}}$ is set to be 400.

5.3 Word recognition

5.3.1 Settings: Different from character recognition, the performance of word recognition is evaluated using whole-word recognition accuracy. In this paper, the word recognition procedure can be regarded as a combination of character recognition, sliding window technique and language model. The processing settings and details are as follows.

First, when training character detectors for word recognition, we only use training samples from ICDAR2003 dataset to build the SED. To accelerate speed of the sliding window classification procedure, only codeword size $n_{w,\mathrm{d}} = 5$, $n_{h,\mathrm{d}} = 5$ is used. k is set to be 20. Totally, we obtain $N_{D_{\mathrm{SED}}} = 3900$ codewords in this SED. Then based on SED, a sparse dictionary D_{SPR} with size of $N_{D_{\mathrm{SPR}}} = 400$ is learned as in (6). Construction coefficients of samples from ICDAR2003 training set are calculated as in (7) and used to train character detectors.

Then given a scene text image, the width is scaled to 64 and height is calculated according to the aspect ratio. We extract sliding windows sequentially with the following six sizes: (i) height = 64, width = 64; (ii) height = 64, width = 48; (iii) height = 64, width = 80; (iv) height = 48, width = 48; (v) height = 48, width = 32; and (vi) height = 48, width = 64. Extracting steps are set to be 1/4 × height for vertical and 1/4 × width for horizontal. All sliding windows are then scaled to height = 64, width = 32. Within each window, SED code vector is computed referring to Section 4.2. Construction coefficients are calculated based on the learned sparse dictionary D_{SPR} as in (7) and then fed into the pre-trained linear SVMs to obtain a classification score.

Finally, after performing character detection at multi-scales, a CRF model is built on the character detection results. NMS is performed to select potential locations. For each potential location, at most five candidate characters are reserved according to SVM scores. We add one node for each position and connect the neighbouring ones from left to right. Then word recognition is performed by solving the energy minimisation problem in (9).

5.3.2 Results and discussion: We evaluate the proposed system on ICDAR2003, ICDAR2011, ICDAR2013 and SVT datasets. Bi-gram language model learnt from the lexicon with 0.5 million words is adopted on all four datasets. Word spotting [27], which refers to spotting word in a small lexicon, is conducted on ICDAR2003, ICDAR11 and SVT datasets. For ICDAR dataset, we evaluate performance using a lexicon created from all the words in the test set [ICDAR03(full), ICDAR11(full)], and with lexicon consisting of the ground truth words plus 50 random words from the test set [ICDAR03(50), ICDAR11(50)]. For SVT dataset, we use the lexicon provided by Wang *et al.* [3]. Similar to [3, 4], we use the inferred result to retrieve the word with the smallest edit distance in the lexicon. Results of our method are compared with state-of-the-art methods as shown in Table 2. Besides, we also report word recognition results without lexicons on all four datasets in Table 3.

Table 3 Word recognition rates of the proposed method on ICDAR2003, ICDAR2011, ICDAR2013 and SVT without lexicons

Method	ICDAR03	ICDAR11	ICDAR13	SVT
our method	41.98	43.09	43.17	26.12

Table 2 Word recognition rates of the proposed method and recent state-of-the-art methods on ICDAR2003, ICDAR2011 and SVT. The results on ICDAR03(50), ICDAR11(50) and SVT are acquired by retrieving the ones with the smallest edit distance in the lexicon of 50 words, whereas for ICDAR03(FULL) and ICDAR11(FULL), the lexicon contains all the ground truth words in the test set

Methods	ICDAR03(FULL)	ICDAR03(50)	ICDAR11(FULL)	ICDAR11(50)	SVT
ABBYY9.0 [32]	55	56	–	–	35
SYNTH + PLEX [3]	62	76	–	–	57
method in [4]	–	81.78	–	–	73.26
method in [33]	67.79	81.78	–	–	73.26
deep CNN [34]	84	90	–	–	70
TSM [5]	79.30	87.44	82.87	87.04	73.51
method in [35]	–	89.69	–	–	77.28
photo OCR [36]	–	–	–	–	90.39
our method	80.46	88.06	82.19	87.52	75.89

doi: 10.1049/iet-cvi.2014.0022

Fig. 11 *Some samples which our system fails to recognise even with lexicons*

It can be seen from Table 2 that our word recognition results can be compared with the latest published algorithms. As we use the similar word model as in [4, 5], the results are quite convincing. This good performance of word recognition relies on character detection accuracy. It further demonstrates the effectiveness of learning co-occurrence of strokes based on SED for robust character recognition. However, our method is still inferior to deep neural network [36] probably because of the limited training samples. Results from Table 3 are not satisfactory as expected. That is perhaps because our word recognition model is still simple. In our opinion, that is also the reason why most published methods mainly focus their attention on word spotting not open vocabulary recognition.

Some images which our system cannot recognise with lexicons are shown in Fig. 11. According to our observation, that is usually because of poor resolution, severe contaminations and heavy occlusions. Under these conditions, pre-trained character detectors often do not work well. Thus, when combined with the language model, the word recognition results are not satisfying.

6 Conclusion

This paper proposes to learn co-occurrence of strokes for scene text recognition. We build a new dictionary named SED, in which each codeword represents a local stroke. Based on SED, a sparse dictionary is further learned to model co-occurrence of local strokes for scene character classification. Experimental results of robust character recognition outperform state-of-the-art algorithms. When combining with a CRF word recognition model, the word recognition results can be compared with the latest published methods.

7 Acknowledgments

This work is supported by the National Natural Science Foundation of China under grant nos. 61172103 and 60933010, State 863 Projects under grant no. 2012AA041312. We also would like to thank the anonymous reviewers for their valuable comments.

8 References

1 Chen, X., Yuille, A.L.: 'Detecting and reading text in natural scenes'. IEEE Conf. Computer Vision and Pattern Recognition (CVPR), 2004, pp. 366–373
2 Neumann, L., Matas, J.: 'Real-time scene text localization and recognition'. IEEE Conf. Computer Vision and Pattern Recognition (CVPR), 2012, pp. 3538–3545
3 Wang, K., Babenko, B., Belongie, S.: 'End-to-end scene text recognition'. IEEE Int. Conf. Computer Vision (ICCV), 2011, pp. 1457–1464
4 Mishra, A., Alahari, K., Jawahar, C.: 'Top-down and bottom-up cues for scene text recognition'. IEEE Conf. Computer Vision and Pattern Recognition (CVPR), 2012, pp. 2687–2694
5 Shi, C., Wang, C., Xiao, B., Zhang, Y., Gao, S., Zhang, Z.: 'Scene text recognition using part-based tree-structured character detection'. IEEE Conf. Computer Vision and Pattern Recognition (CVPR), 2013, pp. 2961–2968
6 Gemert, J., Veenman, C., Smeulders, W., Geusebroek, J.: 'Visual word ambiguity', *IEEE Trans. Pattern Anal. Mach. Intell.*, 2010, **32**, (7), pp. 1271–1283
7 Liu, L., Wang, L., Liu, X.: 'In defense of soft-assignment coding'. , IEEE Int. Conf. Computer Vision (ICCV), 2011, pp. 2486–2493
8 Yang, J., Yu, K., Gong, Y., Huang, T.: 'Linear spatial pyramid matching using sparse coding for image classification'. IEEE Conf. Computer Vision and Pattern Recognition (CVPR), 2009, pp. 1794–1801
9 Boureau, Y., Bach, F., LeCun, Y., Ponce, J.: 'Learning mid-level features for recognition'. IEEE Conf. Computer Vision and Pattern Recognition (CVPR), 2010, pp. 2559–2566
10 Boureau, Y., Ponce, J., LeCun, Y.: 'A theoretical analysis of feature pooling in visual recognition'. Int. Conf. Machine Learning, 2010, pp. 111–118
11 Lazebnik, S., Schmid, C., Ponce, J.: 'Beyond bags of features: spatial pyramid matching for recognizing natural scene categories'. IEEE Conf. Computer Vision and Pattern Recognition (CVPR), 2006, vol. 2, pp. 2169–2178
12 Viitaniemi, V., Laaksonen, J.: 'Spatial extensions to bag of visual words'. ACM Int. Conf. Image and Video Retrieval, 2009, p. 37
13 Yi, C., Yang, X., Tian, Y.: 'Feature representation for scene text character recognition: a comparative study'. Int. Conf. Document Analysis and Recognition (ICDAR), 2013, pp. 907–911
14 Wang, J., Yang, J., Yu, K., Lv, F., Huang, Y., Gong, T.: 'Locality-constrained linear coding for image classification'. IEEE Conf. Computer Vision and Pattern Recognition (CVPR), 2010, pp. 3360–3367
15 Dalal, N., Triggs, B.: 'Histograms of oriented gradients for human detection'. IEEE Conf. Computer Vision and Pattern Recognition (CVPR), 2005, pp. 886–893
16 Fan, R., Chang, K., Hsieh, C., Wang, X., Lin, C.: 'Liblinear: a library for large linear classification', *J. Mach. Learn. Res.*, 2008, **9**, pp. 1871–1874
17 Zou, H., Hastie, T.: 'Regularization and variable selection via the elastic net', *J. R. Stat. Soc., B (Stat. Methodol.)*, 2005, **67**, (2), pp. 301–320
18 Yao, B., Jiang, X., Khosla, A., Lin, A., Guibas, L., Li, F.: 'Human action recognition by learning bases of action attributes and parts'. IEEE Int. Conf. Computer Vision (ICCV), 2011, pp. 1331–1338
19 Mairal, J., Bach, F., Ponce, J., Sapiro, G.: 'Online learning for matrix factorization and sparse coding', *J. Mach. Learn. Res.*, 2010, **11**, pp. 19–60
20 Stolcke, A.: 'SRILM-an extensible language modeling toolkit'. INTERSPEECH, 2002
21 Kolmogorov, V.: 'Convergent tree-reweighted message passing for energy minimization', *IEEE Trans. Pattern Anal. Mach. Intell.*, 2006, **28**, (10), pp. 1568–1583
22 Pearl, J.: 'Probabilistic reasoning in intelligent systems: networks of plausible inference' (Morgan Kaufmann, San Francisco, 1988)
23 Lucas, S., Panaretos, A., Sosa, L., Tang, A., Wong, S., Young, R.: 'ICDAR 2003 robust reading competitions'. Int. Conf. Document Analysis and Recognition (ICDAR), 2003, pp. 682–687
24 Campos, T., Babu, B., Varma, M.: 'Character recognition in natural images'. Computer Vision Theory and Applications, 2009, pp. 273–280
25 'Chars74 k'. Available at http://www.ee.surrey.ac.uk/CVSSP/demos/chars74k/
26 Coates, A., Carpenter, B., Case, C., *et al.*: 'Text detection and character recognition in scene images with unsupervised feature learning'. Int. Conf. Document Analysis and Recognition (ICDAR), 2011, pp. 440–445
27 Wang, K., Belongie, S.: 'Word spotting in the wild'. European Conf. Computer Vision, 2010, pp. 591–604
28 Shahab, A., Shafait, F., Dengel, A.: 'ICDAR 2011 robust reading competition challenge 2: reading text in scene images'. Int. Conf. Document Analysis and Recognition (ICDAR), 2011, pp. 1491–1496
29 Newell, A., Griffin, L.: 'Multiscale histogram of oriented gradient descriptors for robust character recognition'. Int. Conf. Document Analysis and Recognition (ICDAR), 2011, pp. 1085–1089

30 Tian, S., Lu, S., Su, B., Tan, C.: 'Scene text recognition using co-occurrence of histogram of oriented gradients'. Int. Conf. Document Analysis and Recognition (ICDAR), 2013, pp. 912–916
31 Neumann, L., Matas, J.: 'A method for text localization and recognition in real-world images'. Asian Conf. Computer Vision, 2010, pp. 770–783
32 'Abbyy finereader 9.0'. Available at http://www.abbyy.com
33 Mishra, A., Alahari, K., Jawahar, C.V.: 'Scene text recognition using higher order language priors'. British Machine Vision Conf. (BMVC), 2012, pp. 1–11
34 Wang, T., Wu, D., Coates, A., Ng, A.: 'End-to-end text recognition with convolutional neural networks'. Int. Conf. Pattern Recognition (ICPR), 2012, pp. 3304–3308
35 Goel, V., Mishra, A., Alahari, K., Jawahar, C.V.: 'Whole is greater than sum of parts: recognizing scene text words'. Int. Conf. Document Analysis and Recognition (ICDAR), 2013, pp. 398–402
36 Bissacco, A., Cummins, M., Netzer, Y., Neven, H.: 'PhotoOCR: reading text in uncontrolled conditions'. IEEE Int. Conf. Computer Vision (ICCV), 2013, pp. 785–792

doi: 10.1049/iet-cvi.2014.0022