Automatic Discrimination of Text and Non-Text Natural Images

Chengquan Zhang, Cong Yao, Baoguang Shi, Xiang Bai
School of Electronic Information and Communications
Huazhong University of Science and Technology
Wuhan, 430074, P.R. China.
Email: {zchengquan, yaocong2010, shibaoguang}@gmail.com, xbai@hust.edu.cn

Abstract—With the rapid growth of image and video data, there comes an interesting yet challenging problem: How to organize and utilize such large volume of data? Textual content in images and videos is an important source of information, which can be of great usefulness and assistance. Therefore, we investigate in this paper the problem of text image discrimination, which aims at distinguishing natural images with text from those without text. To tackle this problem, we propose a method that combines three mature techniques in this area, namely: MSER, CNN and BoW. To better evaluate the proposed algorithm, we also construct a large benchmark for text image discrimination, which includes natural images in a variety of scenarios. This algorithm has proven to be both effective and efficient, thus it can serve as a tool for mining valuable textual information from huge amount of image and video data.

I. INTRODUCTION

Text present in images and videos, either protogenous or superimposed, carries rich and precise information, thus it can serve as an important cue for a wide range of applications, such as image search [1], human-computer interaction [2] and assistive system for the blind [3]. Different from conventional approaches related to textual information understanding, which are usually concerned with text detection or recognition, we tackle in this paper the problem of text image discrimination. Given a collection of natural images, the goal of text image discrimination is to discriminate the images that contain text from those that do not contain any text, without considering the location or content of text, as shown in Fig. 1. This problem is obviously of great importance and practicality, but has been neglected by the community for a long period of time.

The difficulties in text image discrimination mainly stem from three aspects: (1) Diversity of texts. In stark contrast to characters in document images, which are usually with regular fonts, similar color and horizontal orientation, texts in natural images are far more diverse, i.e., they may exhibit significant variations in font, color, orientation, or even language type. Moreover, the portion of texts in an image can vary tremendously, for instance, a single character in a corner or multiple lines of characters occupying the whole image; (2) Complexity of backgrounds. The backgrounds may contain a lot of non-text elements (such as branch, grass, fence and sign) that are very alike true text, which might cause confusion; (3) Interference factors. Various factors, like noise, blur and non-uniform illumination, may give rise to failures.

To address these challenges, we propose an effective and robust algorithm, which takes the advantages of three widely used techniques in this field: MSER, CNN and BoW. Character candidates are extracted using MSER [4]. CNN act as a discriminative codebook, which compute a bank of responses for each candidate. For each image, a feature vector is generated with BoW, by aggregating and pooling the responses of all the candidates in the image. The final prediction (text or non-text image) is given by a trained SVM classifier.

Moreover, we generate and release a large image database for text image discrimination, since there is no public benchmark in this field. This dataset contains a collection of natural images with and without text, from various scenarios. The texts in the images are in different languages, colors, scales, fonts, orientations, and layouts. Due to its diversity and representativeness, we believe this dataset can serve as a standard benchmark for text image discrimination.

We have evaluated the effectiveness of the proposed algorithm on the proposed dataset. The experiments demonstrate that the proposed method achieves superior performance, compared with conventional approaches. With GPU, this algorithm runs reasonably fast, so it can be used in large-scale non-text image filtration.

In summary, the contributions of this paper are two-fold:

• We emphasize the significance of text image discrimination and propose an effective algorithm for it.
• We establish a large benchmark for algorithm development, assessment and comparison. It includes 15302 images, exhibiting variations in scene, style, illumination, text appearance and layout, etc.

II. RELATED WORK

Text has been the core research object in the document analysis and recognition community. However, majority of the existing works have focused on document images [5], [6], while those concerned with natural images mainly tackle tasks like text detection and recognition [7]–[9], rather the task of text/non-text distinction.

There are a fraction of works that address text/non-text distinction. Indermuhle et al. [10] and Vidya et al. [11] proposed a system that can distinguish between text and non-text regions in handwritten documents, respectively. However, these methods were aimed at classifying text/non-text at region level, rather than image level.

Most related to our work, the algorithm of Alessi et al. [12] is able to detect if a digital image contains text or not, it is only applicable to document images or simple video frames. In contrast, the proposed approach is capable of distinguishing between natural images with text and those without text. In this sense, this work is the first that formerly tackle the problem of text image discrimination for real-world natural images.

III. TEXT IMAGE DISCRIMINATION METHOD

In this section, we describe in detail the proposed method for text image discrimination, which is an ingenious combination of Maximally Stable Extremal Region (MSER) [4], Convolutional Neural Network (CNN) [13] and Bag of Words (BoW) [14]. Recently, a rich body of works [15]–[17] have demonstrated that CNN is effective for scene text detection and recognition. The proposed algorithm also utilizes the ability of CNN to learn discriminative features for accurate text image discrimination. In training phase, K-means [18] are applied to text regions extracted by MSER, before training a CNN model. This converts the binary-class classification problem (text vs. non-text) to multi-class classification problem. According to the response of each regions, a histogram-like descriptor, which represents the characteristics of a natural image, is formed using Bag of Words (BoW).

A. Region Extraction and Clustering

Maximally Stable Extremal Regions [4] (MSER) is widely used in the fields of text detection and recognition, since it is able to discover candidate characters effectively and efficiently. Neumann et al. [19] have also proved that MSER can capture over 95% true text regions with using multiple channels. In our method, we employ MSER to extract regions which can support the discrimination of text and non-text images. However, different from text detection or recognition, text image discrimination does not require the localization and recognition characters. Moreover, the proposed method allows text regions with few connection characters. Multi-channels (RGB-Gray) and loose settings are used in our method.

Before training the CNN model, we compute HOG [20] feature for each text region from foreground (text component), and K-means is performed to cluster the text regions into multiple categories. The softmax layer in CNN model will produce a response with (K+1) dimension for each region. Naturally, the dimension of the final feature is (K+1). Note that, clustering is not applied to regions from background.

B. Multi-Class CNN Model Training

The whole CNN Structure is composed of 4 convolutional layers and 2 full connection layers. Normally, each convolutional layer is followed by maxpooling and rectified units. The setting of the CNN architecture used in this paper is showed in TABLE I.

<table>
<thead>
<tr>
<th>Layer</th>
<th>Settings</th>
</tr>
</thead>
<tbody>
<tr>
<td>Conv-1</td>
<td>ks=3, ps=1, ss=1, nMap=64</td>
</tr>
<tr>
<td>Conv-2</td>
<td>ks=3, ps=2, ss=1, nMap=128</td>
</tr>
<tr>
<td>Conv-3</td>
<td>ks=3, ps=1, ss=1, nMap=384</td>
</tr>
<tr>
<td>Full-1</td>
<td>nNode=1024</td>
</tr>
<tr>
<td>Full-2</td>
<td>nNode=(K+1)</td>
</tr>
</tbody>
</table>

The input image region is rescaled to 32 × 32, and this would generate 1×1 size maps through the first 4 convolutional layers under special design. The kernel numbers we used are 64, 128, 384 and 512, respectively. After 4 convolution steps, 2 full connection layers with 1024 and (K+1) perception units are closely followed. Since softmax are used in the last layer, we can obtain a vector of (K+1) dimension, where each dimension denotes the probability of one category.

In the training phase, we use Stochastic Gradient Descent and back-propagation to jointly optimize all parameters, which can minimize the classification error as much as possible. Two important measures are taken into consideration in the training process: one is penalty factor, the other is boosting. Since clustering methods belong to unsupervised learning, which can’t divide the samples into accuracy classes. Thus, we can use different penalty factors for regions from foreground and background, by using the Negative Log Likelihood criterion as loss function. Here, we design low penalty factors for K text
region clusters, which allows for CNN model to adjust the suitable clustering results. However, high value is assigned to regions from background, because whether the model is good at filtering non-text regions is important for the feature vector generation step.

In order to make the CNN model be good at filtering non-text regions, we use boosting [21] to optimize our model. At first, we divide the regions into two groups: regions from foreground (text regions) and regions from background (non-text regions). K-means is adopted to divide the text regions into multiple clusters, while keep one class for non-text regions. Because of the complexity and diversity of background, we randomly select some samples from the background. When the first time of the CNN model achieving an optimum state, we add hard non-text examples in next training process with the use of this model, which procedure repeats for 3~5 times.

C. Feature Vector Generation and Linear SVM Training

Given an image, N regions are extracted with MSER in multi-channels, which is followed by color channel changing and scale regularization. Each region will generate a response with (K+1) dimension. A feature matrix with the size of $N \times (K + 1)$ can be used to represent the whole image. Then, a feature vector is formed with BoW, by aggregating and pooling the feature matrix, which can describe the characteristics of the image. We call this operation CNN Coding, since the softmax output of CNN model acts as the codebook, which is similar to sparse coding used in word retrieval [22]. The coding output can be expressed as follow:

$$\Phi(I) = \sum_{i=1}^{K+1} \alpha_i \phi_i,$$

where $I$ denotes an image and $\phi_i$ is the i-th class response. $\alpha_i$ denotes the weight of i-th class. Thus, $\Phi(I)$ stands for the CNN Coding result for a whole image.

A feasible way that can improve upon CNN Coding is to perform region filtering before coding, since regions from background are complex and diverse. Moreover, given an text image, the number of non-text regions extracted by MSER is usually several order of magnitude larger than that of the text regions. In order to generate an reliable and effective descriptor for an image, some measures should be adopted to control the ratio between text and non-text regions. The trained CNN model has proven to be excellent at filtering non-text regions, so we not only utilize the CNN model to generate a feature vector of an image, but also use the model as a filter. As shown in Fig. 3, the CNN model is able to effectively eliminate non-text regions, which makes the final feature more discriminative. We will demonstrate the advantage of this strategy in Sec. V.

Finally, the coding results are fed to a SVM classifier. In our method, we used LIBLINEAR [23] as the final classifier, which leads to excellent performance.

IV. DATASET AND EVALUATION PROTOCOL

In this section, we introduce the proposed dataset for text image discrimination and the corresponding evaluation method.

A. Dataset

In order to evaluate the proposed algorithm and compare it with other methods, we have collected a large dataset from internet, which includes 7302 text images and 8000 non-text images. Majority of the images are natural image, while a small number of them are born-digital images and scanned document images. The dataset is challenging and diverse, due to the variations in scene type, style, illumination, language type and text layout, as well as the complexity of backgrounds. As shown in Fig. 4, the text images are manually labelled with bounding boxes for visible text regions. We randomly select 2000 samples from each class to build the testing dataset, all the rest images are used as the training set.

B. Evaluation Protocol

Alike to evaluation methods for general image classification, precision($P$), recall($R$) and F-Measure($F$) are used as the metrics to measure the performance of algorithms for text image discrimination. These three metrics are defined as follows:

$$P = \frac{TP}{TP + FP}$$
$$R = \frac{TP}{TP + FN}$$

Fig. 3: Regions filtered by the CNN model. (a) Text image, (b) Non-text image. Input images are showed in the first row. The red bounding boxes presented in the second row are the regions extracted by MSER, while the green bounding boxes presented in the third row are the survived regions after filtering.
Fig. 4: Examples from the proposed dataset. (a) Text images along with manually labelled bounding boxes. (b) Non-text images.

\[ F = \frac{2 \times P \times R}{P + R} \]  
\[ (2) \]

where, \( TP \) is the number of text images that are correctly classified and \( FP \) is the number of non-text images that are falsely classified. \( FN \) is the number of text images that are falsely classified. In our experiments, precision means the ratio between the number of text images over that of those predicted as positive, and recall means the ratio between the number of correctly classified text images over all text images in the test set.

V. EXPERIMENTS AND DISCUSSIONS

In this section, we evaluate our algorithm on the proposed benchmark dataset, using the standard evaluation protocol. In the following experiments, we will discuss the effects of different numbers of clusters on text regions, and different percentages of regions reserved after filtering. Moreover, we will compare our method with other traditional algorithms for image classification, such as Locality-constrained Linear Coding (LLC) [24], CNN, and improved LLC with using MSER.

Effect of cluster number: As shown in TABLE II, we compare 6 different numbers of clusters used in training the CNN model. The precision and recall are measured when the F-Measure achieving optimum value. Comparing different values of \( K \), we find out the best performance is achieved at the value of \( K = 100 \), over which there will be no obvious improvement.

TABLE II: Effects of different values of \( K \)

<table>
<thead>
<tr>
<th>((K+1))</th>
<th>Precision</th>
<th>Recall</th>
<th>F-Measure</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>0.889</td>
<td>0.878</td>
<td>0.883</td>
</tr>
<tr>
<td>51</td>
<td><strong>0.906</strong></td>
<td>0.903</td>
<td><strong>0.901</strong></td>
</tr>
<tr>
<td>101</td>
<td>0.898</td>
<td>0.892</td>
<td>0.890</td>
</tr>
<tr>
<td>201</td>
<td>0.892</td>
<td>0.892</td>
<td>0.892</td>
</tr>
<tr>
<td>301</td>
<td>0.881</td>
<td>0.902</td>
<td>0.891</td>
</tr>
<tr>
<td>401</td>
<td>0.894</td>
<td>0.884</td>
<td>0.888</td>
</tr>
<tr>
<td>501</td>
<td>0.879</td>
<td><strong>0.908</strong></td>
<td>0.892</td>
</tr>
</tbody>
</table>

Effect of reserved region ratio: As aforementioned, we found that too many non-text regions will have a negative effect on the final performance. In our experiments, we have experimented with different percentages of regions reserved. As can be observed from TABLE III, the best performance is obtained when 2% top-ranked regions are reserved, which strongly proves that too many regions from background would hurt the classification performance. In this experiment, the precision and recall are also measured when F-Measure achieves optimum value.

TABLE III: Effects of different percentages of regions reserved.

<table>
<thead>
<tr>
<th>Percentage</th>
<th>Precision</th>
<th>Recall</th>
<th>F-Measure</th>
</tr>
</thead>
<tbody>
<tr>
<td>1%</td>
<td>0.900</td>
<td>0.884</td>
<td>0.892</td>
</tr>
<tr>
<td>2%</td>
<td>0.898</td>
<td><strong>0.903</strong></td>
<td><strong>0.901</strong></td>
</tr>
<tr>
<td>5%</td>
<td>0.906</td>
<td>0.866</td>
<td>0.885</td>
</tr>
<tr>
<td>10%</td>
<td>0.897</td>
<td>0.853</td>
<td>0.874</td>
</tr>
<tr>
<td>20%</td>
<td>0.916</td>
<td>0.819</td>
<td>0.864</td>
</tr>
<tr>
<td>50%</td>
<td>0.903</td>
<td>0.816</td>
<td>0.857</td>
</tr>
<tr>
<td>100%</td>
<td>0.901</td>
<td>0.812</td>
<td>0.854</td>
</tr>
</tbody>
</table>

Comparison with other methods: (1) Locality-constrained Linear Coding [24] is our first baseline method, which extracts dense sift features with 3 different scales. The codebook size in our experiment is 2048. Besides, we replace SPM [25] with global max-pooling, since we find the SPM hardly bring any improvement with spending more time on coding. (2) CNN is considered as the second baseline method, whose structure is similar to the model used in our proposed method, except that the input scale is 224 × 224 and a global max-pooling is performed before the last 2 full connection layers. (3) Taking the bounding box information into account, we also design an improved coding scheme for the traditional LLC. For each image HOG, LBP, gradient histogram, and gradient orientation histogram are extracted for the stable regions after filtering, and integrated with the LLC method. We call this method MSER+Adaboost+LLC. The P-R curves in Fig. 5 show that with the bounding box information, the proposed method achieves the significantly enhanced performance, compared with the baseline methods. Note that the
comparison between the proposed method (CNN Coding) and MSER+Adaboost+LLC is fair, since both methods used bounding box labels in the training phase.

**Time Cost:** We also measured the time cost of our algorithm on a regular PC (CPU: Intel(R) Xeon(R) CPU E3-1230 V2 @3.30 GHz; GPU: Tesla K40c; RAM: 8GB). As shown in TABLE IV, our method takes around 0.43~0.49s to accomplish the classification task on a single CPU and a single GPU. We analyse the average time taken for each stage of our pipeline on our dataset, which has an average image size of $720 \times 620$. The step of MSER is executed on C++ with OpenCV-2.4.8. The CNN Coding and linear SVM are executed on the platform of Torch7 under Linux.

The proposed system achieves high classification accuracy and runs reasonably fast, thus it can be used as a powerful tool in large-scale textual information mining tasks.

**TABLE IV: Time cost for each stage.**

<table>
<thead>
<tr>
<th>Stage</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>MSER Extraction</td>
<td>0.18~0.23s</td>
</tr>
<tr>
<td>CNN Coding</td>
<td>0.25~0.26s</td>
</tr>
<tr>
<td>SVM Classification</td>
<td>0.24ms</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td>0.43~0.49s</td>
</tr>
</tbody>
</table>

**VI. CONCLUSION**

We have presented an effective and efficient algorithm for text image discrimination, an important problem in textual information understanding. Moreover, a large image dataset, which includes diverse natural images with and without text, is generated and released. This dataset can serve as a standard benchmark in this field. The experiments on the proposed benchmark show the advantages of the proposed algorithm over conventional methods.

In this paper, we used the bounding boxes as well as the image level labels, for training the model for text image discrimination. In the future, we will investigate methods that only utilize image level labels.

**REFERENCES**


