SCENE TEXT RECOGNITION WITH HIGH PERFORMANCE CNN CLASSIFIER AND EFFICIENT WORD INFERENCE

Xinhao Liu *Takahito Kawanishi* *Xiaomeng Wu* *Kunio Kashino*

NTT Corporation, Kanagawa, Japan 243-0198
{liu.xinhao, kawanishi.takahito, wu.xiaomeng, kashino.kunio} @lab.ntt.co.jp

ABSTRACT

The recognition of text in natural scene images is a practical yet challenging task due to the large variations in backgrounds, textures, fonts, and illumination conditions. In this paper, we propose a highly accurate character recognition model by utilizing the representational power of a specially designed Convolutional Neural Network (CNN). Based on the recognition model, we also develop an efficient post processing approach for error correction and hypothesis re-verification. Character and word image recognition experiments on two public datasets, namely the ICDAR 2003 Robust Reading dataset and the Street View Text (SVT) dataset both show that the proposed approach provides superior or comparable results to the state-of-the-art techniques.

***Index Terms*—** Scene Text Recognition, Convolutional Neural Network, Feature Learning, Character Classification

1. INTRODUCTION AND RELATED WORK

Text in natural scene images usually contains a lot of semantic value such as product names, shop names and traffic signs. Recognizing these texts is an important step for understanding the images and it has attracted increasing attention from the research community. Unlike the characters in printed documents, natural scene texts are more difficult to recognize due to the large variations in backgrounds, textures, fonts, and illumination conditions. Thus developing a robust text detection and recognition algorithm has become an interesting yet challenging research topic.

An end-to-end text recognition system can be decomposed into two tasks: text localization and word recognition. Text localization involves detecting the text locations from a natural scene image. Word recognition consists of identifying the characters and recognizing these characters as a word from a cropped image patch. In this work, we focus on the second task, which is essentially a classification problem. Many algorithms have been proposed as solutions to this problem. For example, Wang et. al.[1] extracted Histogram of Oriented Gradients (HOG) features and used a random ferns sliding window classifier to detect characters in an image, and then grouped them using a pictorial structures model with a fixed lexicon to recognize them. However, the performance was unsatisfactory. Tian et. al. [2] proposed using the Co-occurrence of Histogram of Oriented Gradients (CoHOG) with linear SVM to recognize the characters. Improved as it is, the discriminative power of the CoHOG feature is still limited. Coates et. al. [3] and Wang et. al.[4] proposed using the unsupervised learned Convolutional Neural Network (CNN) features to detect and recognize the texts. Thanks to the high performance of CNN, many algorithms [5, 6, 7] with different network structures have been proposed for character recognition tasks, which greatly improve the accuracy. For word recognition, an optimal solution for character segmentation combined with the lexicon-driven language model is derived by using beam search [5], or a bi-gram classification model [7]. The disadvantage of these algorithms is that the exhaustive search of character location for optimal recognition score and language model score makes the system quite complex and inefficiency.

In this paper, we propose a CNN-based character recognition model and a hypothesis re-verification method for highly efficient and accurate word recognition. The recognition model is based on the CNN structure known as Network in Network (NIN) [8], which was originally designed for natural image classification. Our model learns a softmax function for each character class on top of the fully-connected layer and NIN layer. On the basis of this model, sliding window based character detection and Non-Maximum Suppression (NMS) are employed to find a word candidate. To handle errors in classifier output, we propose a re-verification method to score the hypotheses derived from the words with top-k smallest edit distances in a pre-defined lexicon. In summary, our contributions are: 1) an accurate CNN model for natural scene character recognition and 2) a robust and efficient word inference approach based on hypothesis re-verification. In the experiments, compared with the state-of-the-art, our approach show superior performance for character recognition and comparable performance for word recognition on public datasets.

The following of the paper is organized as: the proposed approach is described in Section 2, The implementation and experimental results are described in detail in Section 3. Conclusions and future direction are given in Section 4.

2. PROPOSED ALGORITHM

2.1. CNN for Character Classification

The most important part of the text recognition system is the character classifier, which greatly affect the performance. Let C denote a character set that has 62 classes (26 upper case, 26 lower case letters and 10 digits). To classify an image patch x into one of the possible character c, instead of using the raw pixels directly, we consider a mapping function $\mathbf{f} : X \mapsto \mathbb{R}^d$, which encodes $x \in X$ as a d-dimensional vectorial representation of $\mathbf{f}(x)$. Then a classifier ϕ is trained to compute the characters' posterior probability distribution

$$p(c|x) = \phi(\mathbf{f}(x)), \tag{1}$$

and the character can be recognized by maximizing the posterior probability,

$$\hat{c} = \arg\max p(c|x). \tag{2}$$

Previously, hand-crafted features such as HOG [1, 2] were used for $\mathbf{f}(\cdot)$. More recently, the learned feature from deep structure CNN has been widely used because of its high performance in many image recognition tasks [9]. In this work we also employ the CNN model for our character classification.

The proposed network has 3 convolutional layers and 1 fully connected layer as shown in Fig. 1. The 32×32 gray scale input image is firstly convolved with 92 kernels of size $5 \times 5 \times 1$ with a stride of 1 pixel. The ReLU [10] normalized output is fed into a 2-layer multilayer perception (MLP) with neurons 160 and 96, and then is max-pooled with a 3×3 kernel. This structure is based on NIN [8] in which, instead of conventional convolution operator, small multilayer perception networks are used to map the input to the next convolutional layer. This structure has achieved state-of-the-art classification results in many image recognition tasks such as CIFAR-10 and inspired by this here we also adopt this structure to improve the character classification accuracy. After that, dropout [11] with a rate of 0.5 is applied to reduce the over-fitting. The second and third convolution layers have similar structure except for the parameters as shown in Fig. 1. The fully connected layer has 128 neurons and the final output is the 62-dim vector with each element represents the confidence for each class. All the parameters of the model are jointly optimized to minimize the classification loss over a training set using Stochastic Gradient Descent (SGD) with a linearly decreasing learning rate 0.01 and momentum 0.9. The details of the training data and training results are shown in Section 3.

2.2. Word Inference

Once the character classifier is trained, the next step is to recognize the cropped image as a whole word. Given a cropped word image, we firstly resize it to a fixed height of 32 pixels while keeping its aspect ratio. Then we evaluate the classifier response for each 32×32 window in a sliding window manner. Sliding window detection has been proven to be effective and is widely used in many vision tasks such as pedestrian detection and face recognition. From this step we can obtain a $62 \times T$ score matrix $\mathbf{M}$, where T is the number of sliding windows. The element $\mathbf{M}(i, t)$ indicates the confidence of t-th window belonging to the i-th character in set C. To make the score matrix case insensitive, we set the score for a character at the larger value between its upper and lower case. To eliminate the repeated detections, we perform Non Maximum Suppression (NMS) over the score matrix $\mathbf{M}$ as introduced in [12, 4]. For each column of $\mathbf{M}$, a response score $R(t)$ is computed as the difference between the best and second best scores in that column.Then the NMS response score $\tilde{R}(t)$ can be calculated as:

$$\tilde{R}(t) = \begin{cases} R(t) & \text{if} \quad R(t) \geqq R(t'), \forall t' \, |t - t'| < \delta \\ 0 & \text{otherwise,} \end{cases} \tag{3}$$

where δ is a width parameter, for which we set as 5 pixels in this work. The columns with response 0 are regarded as the non-text. The most probable labeling of the word image is derived as the characters with the maximum response score according to $\tilde{R}(t)$. The classifier raw output at this point is usually error-prone due to the unsegmented characters and should be further corrected using a language model or a lexicon. We use the edit distance as the metric for correction as described in the next section.

2.3. Hypothesis Re-verification

The lexicon-driven recognition approach has been proved effective, for example, in [7] each lexicon word is considered to be a hypothesis and its recognition score is evaluated. However, evaluating the CNN recognition score of each word in the lexicon becomes very inefficient especially when the lexicon is large. Unlike the lexicon-driven score approach, we use the sliding-window classifier output described in the previous step and correct the output to words with top-k (e.g. top-3) smallest edit distances in the given lexicon. This process does not involve evaluating the CNN score for each lexicon word and the corrections are usually a few best matches, which will greatly reduce the number of hypothesis and improve efficiency. Then we perform second round verification operations on the reduced hypothesis list to obtain the final recognition result.

Given the recognition score matrix $\mathbf{M}$, the score for the word hypothesis $w = \{c_1, c_2, ..., c_L\}$ can be derived as:

$$S(w, \mathbf{M}) = \frac{1}{L} \sum_{i=1}^{L} s(c_i, p_i, \mathbf{M}), \tag{4}$$

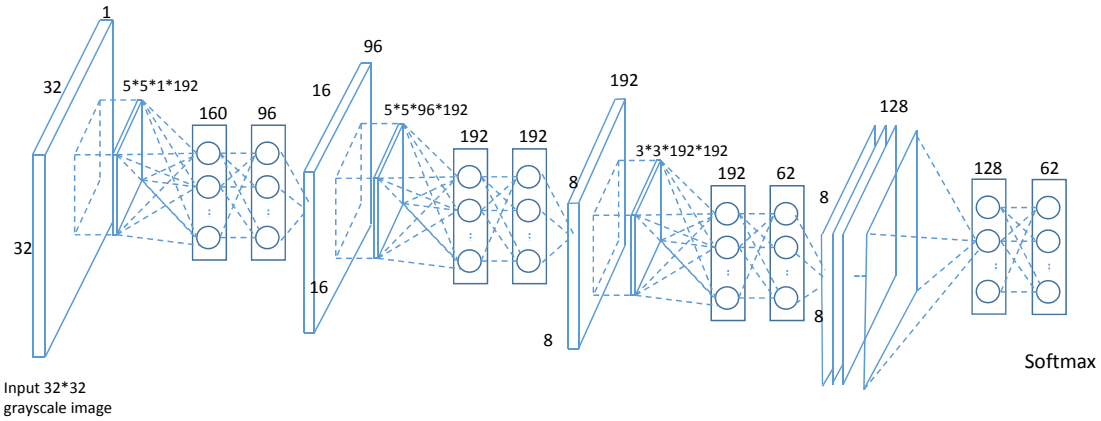

Fig. 1. Proposed CNN structure for character classification.

where $s(c_i, p_i, \mathbf{M})$ is the recognition score for the i-th character c_i, which is centered at position p_i and L is the length of the word. To compute p_i for a word image, we assume each character is equally distributed within the image width. In the experiments it turns out that this assumption works well even for the SVT dataset in which the word bounding boxes are not tightly labeled. The word with the best score is recognized as the final result. Despite its advantage in terms of efficiency, the proposed hypothesis re-verification method also alleviate the negative influence of errors in CNN-based character recognition and NMS step.

3. EXPERIMENTAL RESULTS

In this section we present the experimental setting and recognition results for our algorithm. We use the open source deep learning framework Caffe [13] with GPU to train our CNN character classifier. The training data are collected from several datasets with character level bounding box annotations including ICDAR 2003, 2005, 2013 training set, KAIST [14], natural scene images from Chars74k [15] which give a total of 107k training samples [7]. We resize each sample to the size of 32×32 regardless its original aspect ratio and subtract the mean pixel value.

3.1. Character Recognition Results

For the ICDAR 2003 test set there are 5430 characters (5379 for alphabets and numbers) and for the SVT dataset, only the test set is annotated with a character level bounding box by [16] and it contains 3796 samples. The experimental results in Table 1 show the character classification performance of different algorithms. We can see that the proposed algorithm outperforms all existing methods both for case-sensitive and case-insensitive classification. Note that the testing dataset in [4, 6, 7] consists of only 5198 rather than 5379 characters because in [3] the characters are re-cropped with a proper window size that fits the classifier, and some of the out-of-box characters were removed. This procedure corrects some of the annotation errors of the original test set but also excludes some difficult test cases, which makes the classification task easier. To compare the results with different algorithms fairly we evaluate these two cases separately and report both results.

3.2. Cropped Word Recognition Results

We also evaluate the proposed algorithm on word images from the ICDAR 2003 and SVT-WORD datasets. Words with less than two characters are ignored thus leaving 862 test images for the ICDAR 2003 test set and 647 for the SVT-WORD. The recognizer is tested with a lexicon of different size. The lexicon is formed by the ground truth word plus a number of distractors. For the ICDAR 2003 dataset, the distractors are all words in ICDAR 2003 test set (ICDAR03-full) or 50 randomly selected words (ICDAR03-50) [19] and for SVT-WORD, there are 50 selected words [1].

Table 2 show the cropped word recognition results for different algorithms. Our proposed method achieves 93.5%, 87.2% and 87.5% accuracy on the ICDAR03-50, ICDAR03-full and SVT-WORD datasets respectively. In [5] additional large-scale training data and many post-processing steps are employed, which make the system very complex and in [7] unary and bi-gram character classifiers are trained to improve the accuracy, which is difficult to extend other languages with a much larger character set than alphabets and digits such as Chinese or Japanese. In contrast, thanks to the high performance of the character classifier, the proposed method achieves comparable or superior results to state-of-the-art algorithms even without a complex language model and post-processing, especially for the more difficult dataset SVT-WORD. Fig. 2 (a) shows some challenging examples that are correctly recognized by the proposed algorithm and

Table 1. Character classification accuracy (%) on the ICDAR 2003 and SVT dataset (the numbers in brackets indicate the number of test data)

	Case sensitive classifier			Case insensitive classifier		
Method	ICDAR03 (5379)	ICDAR03 (5198)	SVT (3796)	ICDAR03 (5379)	ICDAR03 (5198)	SVT (3796)
ABBYY FineReader [17]	26.6	-	15.4	-	-	-
T. Wang et al. [4]	-	83.9	-	-	-	-
Co-HOG + SVM [2]	79.4	-	**75.4**	83.6	-	80.6
Alsharif et al. [6]	-	86.0	-	-	89.8	-
Conv Co-HOG [18]	81.0	-	75.0	85.3	-	81.1
Jaderberg et al. [7]	-	86.8	-	-	91.0	80.3
Proposed	**83.0**	**89.5**	**75.4**	**90.1**	**93.1**	**84.4**

Table 2. Cropped word recognition accuracy (%) on the ICDAR 2003 and SVT-WORD dataset.

Method	ICDAR03-50	ICDAR03-full	SVT-WORD
K. Wang et al. [19]	76.0	62.0	57
Mishar et al. [20]	81.8	67.8	73.2
T. Wang et al. [4]	90.0	84.0	70.0
Alsharif et al. [6]	93.1	88.6	74.3
Jaderberg et al. [7]	**96.2**	**91.5**	86.1
Proposed (w/o re-verification)	89.2	78.2	85.3
Proposed	93.5	87.2	**87.5**

Fig. 2 (b) show some failures. We can see that the proposed algorithm is capable of recognizing many difficult cases with noisy low resolution and complex background images and some failures are even difficult for humans to read.

4. CONCLUSION

In this paper, we propose a convolutional neural network based method for the task of natural scene character and word recognition. The features learned from the deep structure are more robust to background variations of the natural scene images. Base on the high performance character classifier model, we can easily recognize the word image with a more simple and efficient word inference approach. Experimental results on different tasks and datasets show that the performance of the proposed algorithm is superior or comparable to the state-of-the-art both for character classification and word recognition. In future work, we will continue to improve the text recognition accuracy by incorporating with a high level language model while maintaining its efficiency.

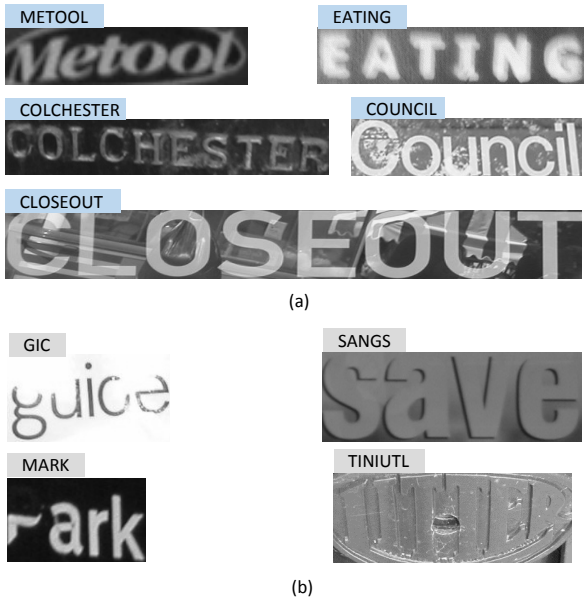

Fig. 2. Examples of recognition results from the ICDAR dataset: (a) correctly recognized cases, (b) failures.

5. REFERENCES

[1] Kai Wang and Serge Belongie, "Word spotting in the wild," in *ECCV*, pp. 591–604. Springer, 2010.

[2] Shangxuan Tian, Shijian Lu, Bolan Su, and Chew Lim Tan, "Scene text recognition using Co-occurrence of Histogram of Oriented Gradients," in *Document Analysis and Recognition (ICDAR), 2013 12th International*

Conference on. IEEE, 2013, pp. 912–916.

[3] Adam Coates, Blake Carpenter, Carl Case, Sanjeev Satheesh, Bipin Suresh, Tao Wang, David J Wu, and Andrew Y Ng, "Text detection and character recognition in scene images with unsupervised feature learning," in *Document Analysis and Recognition (ICDAR), 2011 International Conference on*. IEEE, 2011, pp. 440–445.

[4] Tao Wang, David J Wu, Andrew Coates, and Andrew Y Ng, "End-to-end text recognition with convolutional neural networks," in *Pattern Recognition (ICPR), 2012 21st International Conference on*. IEEE, 2012, pp. 3304–3308.

[5] Alessandro Bissacco, Mark Cummins, Yuval Netzer, and Hartmut Neven, "PhotoOCR: Reading text in uncontrolled conditions," in *Computer Vision (ICCV), 2013 IEEE International Conference on*. IEEE, 2013, pp. 785–792.

[6] Ouais Alsharif and Joelle Pineau, "End-to-end text recognition with hybrid HMM maxout models," in *ICLR*, 2014.

[7] Max Jaderberg, Andrea Vedaldi, and Andrew Zisserman, "Deep features for text spotting," in *Computer Vision–ECCV 2014*, pp. 512–528. Springer, 2014.

[8] Chen Q. Lin, M. and Yan, "Network in network," in *ICLR*, 2014.

[9] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton, "Imagenet classification with deep convolutional neural networks," in *NIPS*, 2012, pp. 1097–1105.

[10] Vinod Nair and Geoffrey E Hinton, "Rectified linear units improve restricted Boltzmann machines," in *Proceedings of the 27th International Conference on Machine Learning*, 2010, pp. 807–814.

[11] Geoffrey E Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R Salakhutdinov, "Improving neural networks by preventing co-adaptation of feature detectors," *arXiv preprint arXiv:1207.0580*, 2012.

[12] Alexander Neubeck and Luc Van Gool, "Efficient non-maximum suppression," in *Pattern Recognition, 18th International Conference on*. IEEE, 2006, vol. 3, pp. 850–855.

[13] Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell, "Caffe: Convolutional architecture for fast feature embedding," *arXiv preprint arXiv:1408.5093*, 2014.

[14] "Kaist Scene Text Database," `http://www.iapr-tc11.org/mediawiki/index.php/KAIST_Scene_Text_Database`, 2011.

[15] T. E. de Campos, B. R. Babu, and M. Varma, "Character recognition in natural images," in *Proceedings of the International Conference on Computer Vision Theory and Applications, Lisbon, Portugal*, February 2009.

[16] Anand Mishra, Karteek Alahari, and CV Jawahar, "Top-down and bottom-up cues for scene text recognition," in *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*. IEEE, 2012, pp. 2687–2694.

[17] "ABBYY FineReader 10," `http://www.abbyy.com/`.

[18] Bolan Su, Shijian Lu, Shangxuan Tian, Joo Hwee Lim, and Chew Lim Tan, "Character recognition in natural scenes using convolutional co-occurrence HOG," in *Pattern Recognition (ICPR), 2014 22nd International Conference on*. IEEE, 2014, pp. 2926–2931.

[19] Kai Wang, Boris Babenko, and Serge Belongie, "End-to-end scene text recognition," in *ICCV*. IEEE, 2011, pp. 1457–1464.

[20] Anand Mishra, Karteek Alahari, and CV Jawahar, "Scene text recognition using higher order language priors," in *BMVC*, 2012.