Text Detection and Character Recognition in Scene Images with Unsupervised Feature Learning

Adam Coates, Blake Carpenter, Carl Case, Sanjeev Satheesh, Bipin Suresh, Tao Wang, David J. Wu, Andrew Y. Ng
Computer Science Department
Stanford University
353 Serra Mall
Stanford, CA 94305 USA
{*acoates,blakec,cbcase,ssanjeev,bipins,twangcat,dwu4,ang*}*@cs.stanford.edu*

***Abstract*—Reading text from photographs is a challenging problem that has received a significant amount of attention. Two key components of most systems are (i) text detection from images and (ii) character recognition, and many recent methods have been proposed to design better feature representations and models for both. In this paper, we apply methods recently developed in machine learning–specifically, large-scale algorithms for *learning* the features automatically from unlabeled data–and show that they allow us to construct highly effective classifiers for both detection and recognition to be used in a high accuracy end-to-end system.**

***Keywords*-Robust reading, character recognition, feature learning, photo OCR**

I. Introduction

Detection of text and identification of characters in scene images is a challenging visual recognition problem. As in much of computer vision, the challenges posed by the complexity of these images have been combated with hand-designed features [1], [2], [3] and models that incorporate various pieces of high-level prior knowledge [4], [5]. In this paper, we produce results from a system that attempts to learn the necessary features directly from the data as an alternative to using purpose-built, text-specific features or models. Among our results, we achieve performance among the best known on the ICDAR 2003 character recognition dataset.

In contrast to more classical OCR problems, where the characters are typically monotone on fixed backgrounds, character recognition in scene images is potentially far more complicated due to the many possible variations in background, lighting, texture and font. As a result, building complete systems for these scenarios requires us to invent representations that account for all of these types of variations. Indeed, significant effort has gone into creating such systems, with top performers integrating dozens of cleverly combined features and processing stages [5]. Recent work in machine learning, however, has sought to create algorithms that can learn higher level representations of data automatically for many tasks. Such systems might be particularly valuable where specialized features are needed but not easily created by hand. Another potential strength of these approaches is that we can easily generate large numbers of features that enable higher performance to be achieved by classification algorithms. In this paper, we'll apply one such feature learning system to determine to what extent these algorithms may be useful in scene text detection and character recognition.

Feature learning algorithms have enjoyed a string of successes in other fields (for instance, achieving high performance in visual recognition [6] and audio recognition [7]). Unfortunately, one caveat is that these systems have often been too computationally expensive, especially for application to large images. To apply these algorithms to scene text applications, we will thus use a more scalable feature learning system. Specifically, we use a variant of K-means clustering to train a bank of features, similarly to the system in [8]. Armed with this tool, we will produce results showing the effect on recognition performance as we increase the number of learned features. Our results will show that it's possible to do quite well simply by learning many features from the data. Our approach contrasts with much prior work in scene text applications, as none of the features used here have been explicitly built for the application at hand. Indeed, the system follows closely the one proposed in [8].

This paper is organized as follows. We will first survey some related work in scene text recognition, as well as the machine learning and vision results that inform our basic approach in Section II. We'll then describe the learning architecture used in our experiments in Section III, and present our experimental results in Section IV followed by our conclusions.

II. Related Work

Scene text recognition has generated significant interest from many branches of research. While it is now possible to achieve extremely high performance on tasks such as digit recognition in controlled settings [9], the task of detecting and labeling characters in complex scenes remains an active research topic. However, many of the methods used for scene text detection and character recognition are

predicated on cleverly engineered systems specific to the new task. For text detection, for instance, solutions have ranged from simple off-the-shelf classifiers trained on hand-coded features [10] to multi-stage pipelines combining many different algorithms [11], [5]. Common features include edge features, texture descriptors, and shape contexts [1]. Meanwhile, various flavors of probabilistic model have also been applied [4], [12], [13], folding many forms of prior knowledge into the detection and recognition system.

On the other hand, some systems with highly flexible learning schemes attempt to learn all necessary information from labeled data with minimal prior knowledge. For instance, multi-layered neural network architectures have been applied to character recognition and are competitive with other leading methods [14]. This mirrors the success of such approaches in more traditional document and hand-written text recognition systems [15]. Indeed, the method used in our system is related to convolutional neural networks. The primary difference is that the training method used here is unsupervised, and uses a much more scalable training algorithm that can rapidly train many features.

Feature learning methods in general are currently the focus of much research, particularly applied to computer vision problems. As a result, a wide variety of algorithms are now available to learn features from unlabeled data [16], [17], [18], [19], [20]. Many results obtained with feature learning systems have also shown that higher performance in recognition tasks could be achieved through larger scale representations, such as could be generated by a scalable feature learning system. For instance, Van Gemert et al. [21] showed that performance can grow with larger numbers of low-level features, and Li et al. [22] have provided evidence of a similar phenomenon for high-level features like objects and parts. In this work, we focus on training low-level features, but more sophisticated feature learning methods are capable of learning higher level constructs that might be even more effective [23], [7], [17], [6].

III. Learning Architecture

We now describe the architecture used to learn the feature representations and train the classifiers used for our detection and character recognition systems. The basic setup is closely related to a convolutional neural network [15], but due to its training method can be used to rapidly construct extremely large sets of features with minimal tuning.

Our system proceeds in several stages:

1) Apply an unsupervised feature learning algorithm to a set of image patches harvested from the training data to learn a bank of image features.
2) Evaluate the features convolutionally over the training images. Reduce the number of features using spatial pooling [15].
3) Train a linear classifier for either text detection or character recognition.

We will now describe each of these stages in more detail.

A. Feature learning

The key component of our system is the application of an unsupervised learning algorithm to generate the features used for classification. Many choices of unsupervised learning algorithm are available for this purpose, such as auto-encoders [19], RBMs [16], and sparse coding [24]. Here, however, we use a variant of K-means clustering that has been shown to yield results comparable to other methods while also being much simpler and faster.

Like many feature learning schemes, our system works by applying a common recipe:

1) Collect a set of small image patches, $\tilde{x}^{(i)}$ from training data. In our case, we use 8x8 grayscale[1] patches, so $\tilde{x}^{(i)} \in \mathbb{R}^{64}$.
2) Apply simple statistical pre-processing (e.g., whitening) to the patches of the input to yield a new dataset $x^{(i)}$.
3) Run an unsupervised learning algorithm on the $x^{(i)}$ to build a mapping from input patches to a feature vector, $z^{(i)} = f(x^{(i)})$.

The particular system we employ is similar to the one presented in [8]. First, given a set of training images, we extract a set of m 8-by-8 pixel patches to yield vectors of pixels $\tilde{x}^{(i)} \in \mathbb{R}^{64}, i \in \{1, \ldots, m\}$. Each vector is brightness and contrast normalized.[2] We then whiten the $\tilde{x}^{(i)}$ using ZCA[3] whitening [25] to yield $x^{(i)}$. Given this whitened bank of input vectors, we are now ready to learn a set of features that can be evaluated on such patches.

For the unsupervised learning stage, we use a variant of K-means clustering. K-means can be modified so that it yields a dictionary $D \in \mathbb{R}^{64 \times d}$ of normalized basis vectors. Specifically, instead of learning "centroids" based on Euclidean distance, we learn a set of normalized vectors $D^{(j)}, j \in \{1, \ldots, d\}$ to form the columns of D, using inner products as the similarity metric. That is, we solve

$$\min_{D, s^{(i)}} \sum_i ||Ds^{(i)} - x^{(i)}||^2 \tag{1}$$

$$s.t. \quad ||s^{(i)}||_1 = ||s^{(i)}||_\infty, \forall i \tag{2}$$

$$||D^{(j)}||_2 = 1, \forall j \tag{3}$$

where $x^{(i)}$ are the input examples and $s^{(i)}$ are the corresponding "one hot" encodings[4] of the examples. Like K-means, the optimization is done by alternating minimization over D and the $s^{(i)}$. Here, the optimal solution for $s^{(i)}$ given

[1] All of our experiments use grayscale images, though the methods here are equally applicable to color patches.

[2] We subtract out the mean and divide by the standard deviation of all the pixel values.

[3] ZCA whitening is like PCA whitening, except that it rotates the data back to the same axes as the original input.

[4] The constraint $||s^{(i)}||_1 = ||s^{(i)}||_\infty$ means that $s^{(i)}$ may have only 1 non-zero value, though its magnitude is unconstrained.

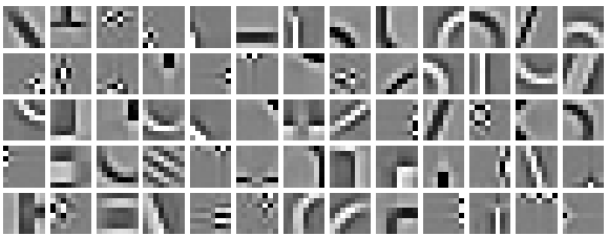


Figure 1. A small subset of the dictionary elements learned from grayscale, 8-by-8 pixel image patches extracted from the ICDAR 2003 dataset.

D is to set $s_k^{(i)} = D^{(k)\top} x^{(i)}$ for $k = \arg\max_j D^{(j)\top} x^{(i)}$, and set $s_j^{(i)} = 0$ for all other $j \neq k$. Then, holding all $s^{(i)}$ fixed, it is easy to solve for D (in closed-form for each column) followed by renormalizing the columns.

Shown in Figure 1 are a set of dictionary elements (columns of D) resulting from this algorithm when applied to whitened patches extracted from small images of characters. These are visibly similar to filters learned by other algorithms (e.g., [24], [25], [16]), even though the method we use is quite simple and very fast. Note that the features are specialized to the data—some elements correspond to short, curved strokes rather than simply to edges.

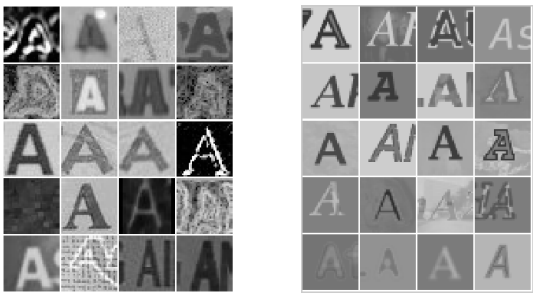
Once we have our trained dictionary, D, we can then define the feature representation for a single new 8-by-8 patch. Given a new input patch $\tilde{x}$, we first apply the normalization and whitening transform used above to yield x, then map it to a new representation $z \in \mathbb{R}^d$ by taking the inner product with each dictionary element (column of D) and applying a scalar nonlinear function. In this work, we use the following mapping, which we have found to work well in other applications: $z = \max\{0, |Dx| - \alpha\}$ where α is a hyper-parameter to be chosen. (We typically use $\alpha = 0.5$.)

B. Feature extraction

Both our detector and character classifier consider 32-by-32 pixel images. To compute the feature representation of the 32-by-32 image, we compute the representation described above for every 8-by-8 sub-patch of the input, yielding a 25-by-25-by-d representation. Formally, we will let $z^{(ij)} \in \mathbb{R}^d$ be the representation of the 8-by-8 patch located at position i, j within the input image. At this stage, it is necessary to reduce the dimensionality of the representation before classification. A common way to do this is with spatial pooling [26] where we combine the responses of a feature at multiple locations into a single feature. In our system, we use average pooling: we sum up the vectors $z^{(ij)}$ over 9 blocks in a 3-by-3 grid over the image, yielding a final feature vector with $9d$ features for this image.

C. Text detector training

For text detection, we train a binary classifier that aims to distinguish 32-by-32 windows that contain text from windows that do not. We build a training set for this classifier by extracting 32-by-32 windows from the ICDAR 2003 training dataset, using the word bounding boxes to decide whether a window is text or non-text.[5] With this procedure, we harvest a set of 60000 32-by-32 windows for training (30000 positive, 30000 negative). We then use the feature extraction method described above to convert each image into a $9d$-dimensional feature vector. These feature vectors and the ground-truth "text" and "not text" labels acquired from the bounding boxes are then used to train a linear SVM. We will later use our feature extractor and the trained classifier for detection in the usual "sliding window" fashion.



(a) Distorted ICDAR examples

(b) Synthetic examples

Figure 2. Augmented training examples.

D. Character classifier training

For character classification, we also use a fixed-sized input image of 32-by-32 pixels, which is applied to the character images in a set of labeled train and test datasets.[6]

However, since we can produce large numbers of features using the feature learning approach above, over-fitting becomes a serious problem when training from the (relatively) small character datasets currently in use. To help mitigate this problem, we have combined data from multiple sources. In particular, we have compiled our training data from the ICDAR 2003 training images [27], Weinman et al.'s sign reading dataset [4], and the English subset of the Chars74k dataset [1]. Our combined training set contains approximately 12400 labeled character images.

With large numbers of features, it is useful to have even more data. To satisfy these needs, we have also experimented with synthetic augmentations of these datasets. In particular, we have added synthetic examples that are copies of the ICDAR training samples with random distortions and image filters applied (see Figure 2(a)), as well as artificial examples of rendered characters blended with random scenery images

[5] We define a window as "text" if 80% of the window's area is within a text region, and the window's width or height is within 30% of the width or height (respectively) of the ground-truth region. The latter condition ensures that the detector tends to detector characters of size similar to the window.

[6] Typically, input images from public datasets are already cropped to the boundaries of the character. Since our classifier uses a fixed-sized window, we re-cropped characters from the original images using an enclosing window of the proper size.

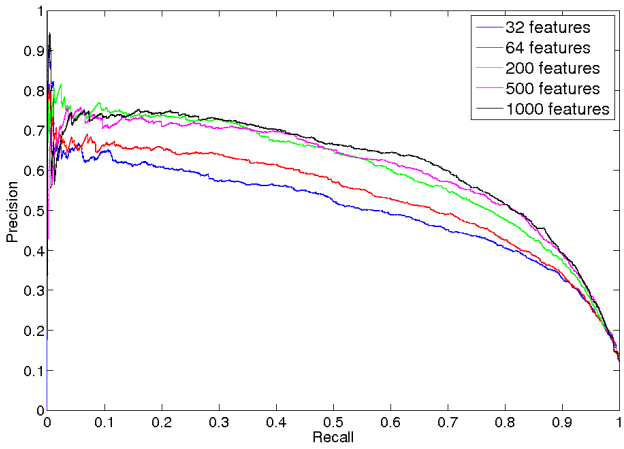


Figure 3. Precision-Recall curves for detectors with varying numbers of features.

(Figure 2(b)). With these examples included, our dataset includes a total of 49200 images.

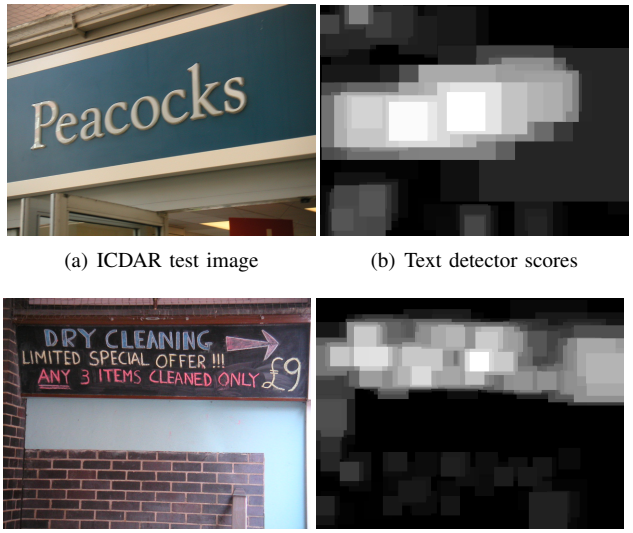
IV. Experiments

We now present experimental results achieved with the system described above, demonstrating the impact of being able to train increasing numbers of features. Specifically, for detection and character recognition, we trained our classifiers with increasing numbers of learned features and in each case evaluated the results on the ICDAR 2003 test sets for text detection and character recognition.

A. Detection

To evaluate our detector over a large input image, we take the classifier trained as in Section III-C and compute the features and classifier output for each 32-by-32 window of the image. We perform this process at multiple scales and then, for each location in the original image assign it a score equal to the maximum classifier output achieved at any scale. By this mechanism, we label each pixel with a score according to whether that pixel is part of a block of text. These scores are then thresholded to yield binary decisions at each pixel. By varying the threshold and using the ICDAR bounding boxes as per-pixel labels, we sweep out a precision-recall curve for the detector and report the area under this curve (AUC) as our final performance measure.

Figure 3 plots the precision-recall curves for our detector for varying numbers of features. It is seen there that performance improves consistently as we increase the number of features. Our detector's performance (area under each curve) improves from 0.5 AUC, to 0.62 AUC simply by including more features. While our performance is not yet comparable to top performing systems it is notable that our approach included virtually no prior knowledge. In contrast, Pan et al.'s recent state-of-the-art system [5] involves multiple highly tuned processing stages incorporating several sets of expert-chosen features.



(a) ICDAR test image (b) Text detector scores

(c) ICDAR test image (d) Text detector scores

Figure 4. Example text detection classifier outputs.

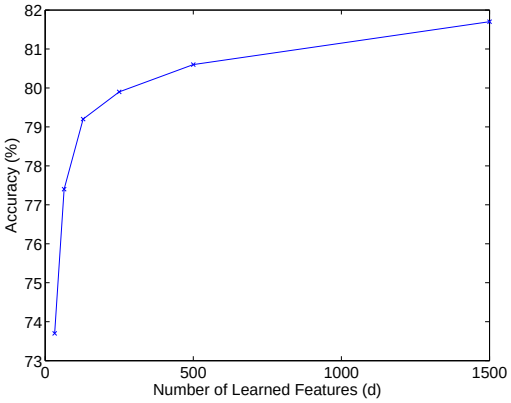


Figure 5. Character classification accuracy (62-way) on ICDAR 2003 test set as a function of the number of learned features.

Note that these numbers are per-pixel accuracies (i.e., the performance of the detector in identifying, for a single window, whether it is text or non-text). In practice, the predicted labels of adjacent windows are highly correlated and thus the outputs include large contiguous "clumps" of positively and negatively labeled windows that could be passed on for more processing. A typical result generated by our detector is shown in Figure 4.

B. Character Recognition

As with the detectors, we trained our character classifiers with varying numbers of features on the combined training set described in Section III. We then tested this classifier on the ICDAR 2003 test set, which contains 5198 test characters

[7] Achieved without pre-segmented characters.

Table I
TEST RECOGNITION ACCURACY ON ICDAR 2003 CHARACTER SETS. (DATASET-CLASSES)

Algorithm	Test-62	Sample-62	Sample-36
Neumann and Matas, 2010 [28]	67.0%[7]	-	-
Yokobayashi et al., 2006 [2]	-	81.4%	-
Saidane and Garcia, 2007 [14]	-	-	84.5%
This paper	81.7%	81.4%	85.5%

from 62 classes (10 digits, 26 upper- and 26 lower-case letters). The average classification accuracy on the ICDAR test set for increasing numbers of features is plotted in Figure 5. Again, we see that accuracy climbs as a function of the number of features. Note that the accuracy for the largest system (1500 features) is the highest, at 81.7% for the 62-way classification problem. This is comparable or superior to other (purpose-built) systems tested on the same problem. For instance, the system in [2], achieves 81.4% on the smaller ICDAR "sample" set where we, too, achieve 81.4%. The authors of [14], employing a supervised convolutional network, achieve 84.5% on this dataset when it is collapsed to a 36-way problem (removing case sensitivity). In that scenario, our system achieves 85.5% with 1500 features. These results are summarized in comparison to other work in Table I.

V. CONCLUSION

In this paper we have produced a text detection and recognition system based on a scalable feature learning algorithm and applied it to images of text in natural scenes. We demonstrated that with larger banks of features we are able to achieve increasing accuracy with top performance comparable to other systems, similar to results observed in other areas of computer vision and machine learning. Thus, while much research has focused on developing by hand the models and features used in scene-text applications, our results point out that it may be possible to achieve high performance using a more automated and scalable solution. With more scalable and sophisticated feature learning algorithms currently being developed by machine learning researchers, it is possible that the approaches pursued here might achieve performance well beyond what is possible through other methods that rely heavily on hand-coded prior knowledge.

ACKNOWLEDGMENT

Adam Coates is supported by a Stanford Graduate Fellowship.

REFERENCES

[1] 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.

[2] M. Yokobayashi and T. Wakahara, "Binarization and recognition of degraded characters using a maximum separability axis in color space and gat correlation," in *International Conference on Pattern Recognition*, vol. 2, 2006, pp. 885–888.

[3] J. J. Weinman, "Typographical features for scene text recognition," in *Proc. IAPR International Conference on Pattern Recognition*, Aug. 2010, pp. 3987–3990.

[4] J. Weinman, E. Learned-Miller, and A. R. Hanson, "Scene text recognition using similarity and a lexicon with sparse belief propagation," in *Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 10, 2009.

[5] Y. Pan, X. Hou, and C. Liu, "Text localization in natural scene images based on conditional random field," in *International Conference on Document Analysis and Recognition*, 2009.

[6] J. Yang, K. Yu, Y. Gong, and T. S. Huang, "Linear spatial pyramid matching using sparse coding for image classification." in *Computer Vision and Pattern Recognition*, 2009.

[7] H. Lee, R. Grosse, R. Ranganath, and A. Y. Ng, "Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations," in *International Conference on Machine Learning*, 2009.

[8] A. Coates, H. Lee, and A. Y. Ng, "An analysis of single-layer networks in unsupervised feature learning," in *International Conference on Artificial Intelligence and Statistics*, 2011.

[9] M. Ranzato, Y. Boureau, and Y. LeCun, "Sparse feature learning for deep belief networks," in *Neural Information Processing Systems*, 2007.

[10] X. Chen and A. Yuille, "Detecting and reading text in natural scenes," in *Computer Vision and Pattern Recognition*, vol. 2, 2004.

[11] Y. Pan, X. Hou, and C. Liu, "A robust system to detect and localize texts in natural scene images," in *International Workshop on Document Analysis Systems*, 2008.

[12] J. J. Weinman, E. Learned-Miller, and A. R. Hanson, "A discriminative semi-markov model for robust scene text recognition," in *Proc. IAPR International Conference on Pattern Recognition*, Dec. 2008.

[13] X. Fan and G. Fan, "Graphical Models for Joint Segmentation and Recognition of License Plate Characters," *IEEE Signal Processing Letters*, vol. 16, no. 1, 2009.

[14] Z. Saidane and C. Garcia, "Automatic scene text recognition using a convolutional neural network," in *Workshop on Camera-Based Document Analysis and Recognition*, 2007.

[15] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel, "Backpropagation applied to handwritten zip code recognition," *Neural Computation*, vol. 1, pp. 541–551, 1989.

[16] G. Hinton, S. Osindero, and Y. Teh, "A fast learning algorithm for deep belief nets," *Neural Computation*, vol. 18, no. 7, pp. 1527–1554, 2006.

[17] R. Salakhutdinov and G. E. Hinton, "Deep Boltzmann Machines," in *12th International Conference on AI and Statistics*, 2009.

[18] M. Ranzato, A. Krizhevsky, and G. E. Hinton, "Factored 3-way Restricted Boltzmann Machines for Modeling Natural Images," in *13th International Conference on AI and Statistics*, 2010.

[19] Y. Bengio, P. Lamblin, D. Popovici, and H. Larochelle, "Greedy layer-wise training of deep networks," in *Neural Information Processing Systems*, 2006.

[20] R. Raina, A. Battle, H. Lee, B. Packer, and A. Ng, "Self-taught learning: transfer learning from unlabeled data," in *24th International Conference on Machine learning*, 2007.

[21] J. C. van Gemert, J. M. Geusebroek, C. J. Veenman, and A. W. M. Smeulders, "Kernel codebooks for scene categorization," in *European Conference on Computer Vision*, 2008.

[22] L.-J. Li, H. Su, E. Xing, and L. Fei-Fei, "Object bank: A high-level image representation for scene classification and semantic feature sparsification," in *Advances in Neural Information Processing Systems*, 2010.

[23] K. Kavukcuoglu, P. Sermanet, Y. Boureau, K. Gregor, M. Mathieu, and Y. LeCun, "Learning convolutional feature hierarchies for visual recognition," in *Advances in Neural Information Processing Systems*, 2010.

[24] B. A. Olshausen and D. J. Field, "Emergence of simple-cell receptive field properties by learning a sparse code for natural images," *Nature*, vol. 381, no. 6583, pp. 607–609, 1996.

[25] A. Hyvarinen and E. Oja, "Independent component analysis: algorithms and applications," *Neural networks*, vol. 13, no. 4-5, pp. 411–430, 2000.

[26] Y. Boureau, F. Bach, Y. LeCun, and J. Ponce, "Learning mid-level features for recognition," in *Computer Vision and Pattern Recognition*, 2010.

[27] S. Lucas, A. Panaretos, L. Sosa, A. Tang, S. Wong, and R. Young, "ICDAR 2003 robust reading competitions," *International Conference on Document Analysis and Recognition*, 2003.

[28] L. Neumann and J. Matas, "A method for text localization and recognition in real-world images," in *Asian Conference on Computer Vision*, 2010.