A study of holistic strategies for the recognition of characters in natural scene images

2016

Muhammad Ali
University of Central Florida

A STUDY OF HOLISTIC STRATEGIES FOR THE RECOGNITION OF CHARACTERS IN NATURAL SCENE IMAGES

by

MUHAMMAD ALI
B.Sc. University of Engineering & Technology Lahore, 2000
M.Sc. University of the Punjab, 2004
M.S. University of Central Florida, 2014

A dissertation submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy
in the Department of Electrical Engineering and Computer Science
in the College of Engineering and Computer Science
at the University of Central Florida
Orlando, Florida

Summer Term
2016

Major Professor: Hassan Foroosh

ABSTRACT

Recognition and understanding of text in scene images is an important and challenging task. The importance can be seen in the context of tasks such as assisted navigation for the blind, providing directions to driverless cars, e.g. Google car, etc. Other applications include automated document archival services, mining text from images, and so on. The challenge comes from a variety of factors, like variable typefaces, uncontrolled imaging conditions, and various sources of noise corrupting the captured images. In this work, we study and address the fundamental problem of recognition of characters extracted from natural scene images, and contribute three holistic strategies to deal with this challenging task.

Scene text recognition (STR) has been a known problem in computer vision and pattern recognition community for over two decades, and is still an active area of research owing to the fact that the recognition performance has still got a lot of room for improvement. Recognition of characters lies at the heart of STR and is a crucial component for a reliable STR system. Most of the current methods heavily rely on discriminative power of local features, such as histograms of oriented gradient (HoG), scale invariant feature transform (SIFT), shape contexts (SC), geometric blur (GB), etc. One of the problems with such methods is that the local features are rasterized in an ad hoc manner to get a single vector for subsequent use in recognition. This rearrangement of features clearly perturbs the spatial correlations that may carry crucial information vis-à-vis recognition. Moreover, such approaches, in general, do not take into account the rotational invariance property that often leads to failed recognition in cases where characters in scene images

do not occur in upright position. To eliminate this local feature dependency and the associated problems, we propose the following three holistic solutions:

The first one is based on modelling character images of a class as a 3-mode tensor and then factoring it into a set of rank-1 matrices and the associated mixing coefficients. Each set of rank-1 matrices spans the solution subspace of a specific image class and enables us to capture the required holistic signature for each character class along with the mixing coefficients associated with each character image. During recognition, we project each test image onto the candidate subspaces to derive its mixing coefficients, which are eventually used for final classification.

The second approach we study in this work lets us form a novel holistic feature for character recognition based on active contour model, also known as snakes. Our feature vector is based on two variables, direction and distance, cumulatively traversed by each point as the initial circular contour evolves under the force field induced by the character image. The initial contour design in conjunction with cross-correlation based similarity metric enables us to account for rotational variance in the character image.

Our third approach is based on modelling a 3-mode tensor via rotation of a single image. This is different from our tensor based approach described above in that we form the tensor using a single image instead of collecting a specific number of samples of a particular class. In this case, to generate a 3D image cube, we rotate an image through a predefined range of angles. This enables us to explicitly capture rotational variance and leads to better performance than various local approaches.

Finally, as an application, we use our holistic model to recognize word images extracted from natural scenes. Here we first use our novel word segmentation method based on image seam analysis to split a scene word into individual character images. We then apply our holistic model to recognize individual letters and use a spell-checker module to get the final word prediction.

Throughout our work, we employ popular scene text datasets, like Chars74K-Font, Chars74K-Image, SVT, and ICDAR03, which include synthetic and natural image sets, to test the performance of our strategies. We compare results of our recognition models with several baseline methods and show comparable or better performance than several local feature-based methods justifying thus the importance of holistic strategies.

I dedicate this work to my daughter Fatima, my wife Rida,

and my parents Asad Ali and Shafqat Batool. Thank you all for your patience and support!

ACKNOWLEDGMENTS

I would like to thank my adviser, Dr. Hassan Foroosh, for his continued support throughout my PhD program. I would also like to thank all my colleagues at the Computational Imaging Laboratory, especially Sun Chuan, for being there whenever I needed them. Special thanks to my advisory committee; Dr. Hughes, Dr. Sukthankar, Dr. Wiegand, and Dr. Yun, for their invaluable comments and suggestions to improve this work. Last but not the least, I would like to extend my gratitude to the Fulbright Scholarship Program for providing me with the necessary funding to pursue my PhD program and to the Institute of International Education (IIE) for their continued support throughout my stay in the US. I would also thank United States Educational Foundation in Pakistan (USEFP) for taking care of my travel and logistics.

TABLE OF CONTENTS

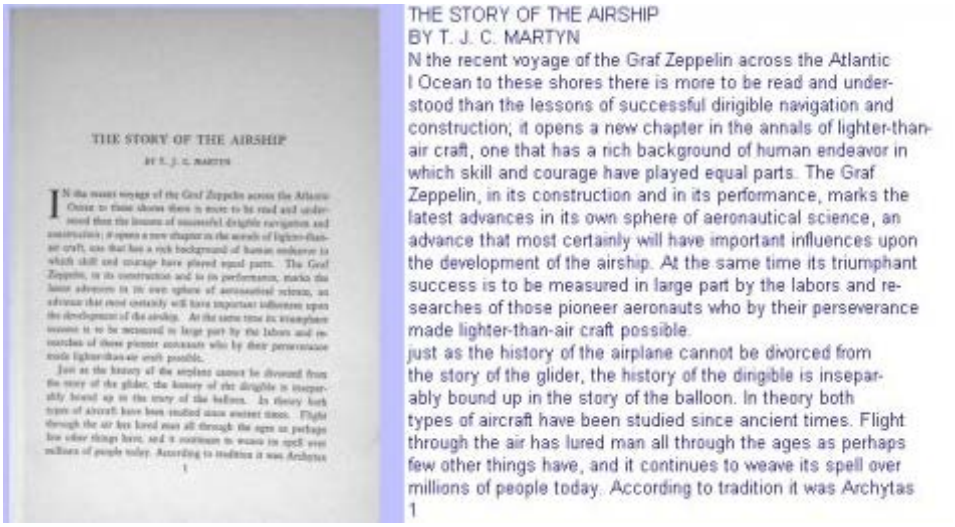
LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

Scene text recognition (STR) is a natural extension to the well-established domain of document optical character recognition (OCR). While document OCR is almost a solved problem, its generalization, i.e. STR, is hard problem to solve. It has been an active area of research since early 2000s and still remains a challenge for pattern recognition community. Figures 1.1 and 1.2 depict the difficulty associated with STR compared with document text recognition.



THE STORY OF THE AIRSHIP
BY T. J. C. MARTYN

N the recent voyage of the Graf Zeppelin across the Atlantic
I Ocean to these shores there is more to be read and understood than the lessons of successful dirigible navigation and construction; it opens a new chapter in the annals of lighter-than-air craft, one that has a rich background of human endeavor in which skill and courage have played equal parts. The Graf Zeppelin, in its construction and in its performance, marks the latest advances in its own sphere of aeronautical science, an advance that most certainly will have important influences upon the development of the airship. At the same time its triumphant success is to be measured in large part by the labors and researches of those pioneer aeronauts who by their perseverance made lighter-than-air craft possible.

just as the history of the airplane cannot be divorced from the story of the glider, the history of the dirigible is inseparably bound up in the story of the balloon. In theory both types of aircraft have been studied since ancient times. Flight through the air has lured man all through the ages as perhaps few other things have, and it continues to weave its spell over millions of people today. According to tradition it was Archytas

1

Figure 1.1: A typical document OCR system

Although there are some partial solutions proposed by researchers over the years, the complex nature of the problem itself and the gap in performance compared with the human's ability to recognize text, still propels interest in this domain.

Figure 1.2: Scene text recognition system
Source: End-to-end Scene Text Recognition (Wang et al., 2011)

1.1 Problem Categories

Natural scene text recognition is a challenging problem in computer vision, machine learning and image processing. Being a broad challenge and owing to the unique set of perspectives it can offer in terms of solutions, the problem has been conventionally broken up by the concerned research community into the following four sub problems:

1. Cropped Character Recognition

2. Cropped Word Recognition

3. Scene Text Detection

4. Full-image Scene Text Recognition

There is another related problem known as ‘word spotting’ proposed by (Wang et al., 2011) to recognize words in an image given a short lexicon pertaining to the image. Since the above mentioned problem areas present a broad scenario of research effort, in this dissertation we will explore the first two sub problems.

We discuss our solutions to the cropped character recognition in Chapters 3 to 5. In Chapter 7, we discuss our solution to the word recognition problem along with the aforementioned .word spotting problem.

1.2 Motivation

Our motivation to study character recognition in natural scene images comes from the following facts:

1. STR is still an unsolved problem that has important applications

2. Most current methods in STR rely on discriminative power of local features that are collected using complex processing and later rasterized in ad hoc manner. Moreover, rotations that commonly occur in scene characters are not generally taken care of.

1.2.1 Importance of STR

The importance of scene text recognition cannot be emphasized more given the application areas it can be utilized in. With the ubiquitous availability of digital cameras on mobile (handheld) devices, e.g. smartphones, there is a deluge of images that are taken by users on a daily basis. The image data floods online repositories. The question as to what can be done with such a humungous

information data arises naturally. One commercial aspect is the automated mining of such data for textual content. Other applications could be efficient image archival and image retrieval.

Computer vision applications running on wearable devices, like glasses, are of tremendous interest. In the important domain of assisted navigation for visually impaired people, there are devices, e.g. OrCam[†] camera mounted on glasses etc., which could directly benefit from STR research. Other utilities of natural scene text recognition include and automatic reading of informational signs for automobile drivers or driverless cars, e.g. Google Car.

1.2.2 Holistic Scene Character Recognition

While STR community has mostly focused on solutions that are based on local feature descriptors, e.g. histogram of gradients (HoG), scale invariant feature transform (SIFT), shape contexts (SC), etc., we present a contrasting point of view: holistic features. The need for holistic solution arises from the fact that local features are collected via complex and computationally intensive means, e.g. sliding windows etc. Later on, these features are cast into a single vector through ad hoc rasterization that often perturbs local spatial correlations crucial for recognition task. Moreover, rotations in character images are generally ignored, thereby reducing the recognition performance.

[†] http://www.orcam.com

Our holistic solutions address these issues by avoiding ad hoc rasterization and taking care of image rotations. In Chapter 6, we discuss experiments that support our claim that holistic solutions are better than local feature based methods

1.3 Scene Text Datasets

The introduction of ICDAR03 reading competition (Lucas et al., 2003) and the associated dataset stirred up research interest in document recognition community and subsequently other researchers came forward and proposed their methods and/or datasets. For example (Weinman et al., 2009) used their sign reading dataset (WLM dataset), (de Campos et al., 2009) proposed Chars74K, and (Wang et al., 2011) came up with their Street View Text (SVT) dataset. More recently (Nagy et al., 2011) put forth NEOCR dataset and (Mishra et al, 2012a) proposed IIT5K dataset.

As depicted in the Figures 1.3 and 1.4, which show sample images from popular natural scene text character datasets like Chars74K and ICDAR, the challenge for recognition is obvious. The images suffer from noise and exhibit low resolution, low contrast, variable typefaces, illumination effects, perspective distortions, and rotations.

Figure 1.3 Sample images from different ICDAR & SVT datasets

Figure 1.4 Synthetic characters (top most row) from Chars74K-Font dataset, along with some random samples from Chars74K-Image

In the following we discuss the sources and some attributes of the popular STR datasets we used in our work.

1.3.1 ICDAR03 Character Dataset

The ICDAR03[‡] robust character dataset contains 11,615 images of cropped scene characters and the dataset comes split into training and testing subsets. Characters have mostly been cropped from images of book titles, storefronts and signs and exhibit great variability in terms of resolution, contrast, illumination, color, etc. The test set has 5,340 images in total but those belonging to 62 classes (A-Z, a-z, and 0-9) are just 5,379.

1.3.2 Chars74K Dataset

As the name suggests, the Chars74K[§] dataset originally has about seventy four thousand characters in two languages, English and Kannada. The English subset of Chars74K dataset, also known as Chars74-K-Img, consists of 12,503 characters. Characters have been cropped from 1,922 images of advertisement signs and products from stores etc. This dataset doesn't come split into training and testing sets, rather the authors give their proposed training and testing splits for comparison with their results. There is, however, a split between '*GoodImg*' and '*BadImg*' and as obvious from the names, the respective splits contain 'good' and less noisy (7,705 images) as well as 'bad' more noisy images (4,798 images) for a total of 12,503 images.

[‡] http://www.iapr-tc11.org/mediawiki/index.php/ICDAR_2003_Robust_Reading_Competitions

[§] http://www.ee.surrey.ac.uk/CVSSP/demos/chars74k/

The synthetic version of this dataset is known as Chars74K-Font consists of English alphabet generated in various typefaces for 62 character classes: ‘A’ to ‘Z’, ‘a’ to ‘z’, and digits ‘0’ to ‘9’. The dataset consists of 62,992 images with 1,016 images per class.

1.3.3 SVT-CHAR Dataset

The SVT-CHAR dataset consists of 3,796 character images cut out and annotated by (Mishra et al., 2012) from cropped word images of the Street View Text (SVT) dataset. There is no training portion of SVT-CHAR and results are usually reported only using it as a test set.

1.3.4 ICDAR03-Word Dataset

The ICDAR03 word recognition test dataset contains 1,111 cropped images taken from a variety of sources: from indoor scenes, hallway signage, book covers, etc., to the outdoor signs and billboards. Researchers have used this dataset for both lexicon driven (word spotting) and general word recognition problems (sometimes called as open vocabulary word recognition). It is a challenging dataset and contains images with lot of noise, character distortions, occlusions, and low contrast.

1.3.5 SVT Dataset

The SVT[**] dataset consists of 647 words cropped from 250 scene images. The images are obtained from Google Street View and contain pictures of storefronts and other outdoor business

[**] http://vision.ucsd.edu/~kai/svt/

signage. This, again, is a very challenging dataset and in additions to the usual noise and distortions in natural scene imagery, it also contains a lot of cropping artifacts which makes it harder to segment and recognize.

Figure 1.5: Sample images from SVT dataset for word recognition

Figure 1.6: Samples images from ICDAR03-Word test dataset

1.4 Organization of the Dissertation

The dissertation is organized as follows: Chapter 2 mentions the literature review relating to STR in general and scene character recognition in particular. We mention where our work fits

into the greater picture. Then in Chapter 3, we discuss our first strategy of holistic scene character recognition that is based on getting rank-1 tensor subspaces. Chapter 4 is devoted to our second holistic technique, which is based on evolving a circular active contour to get a novel feature vector. Chapter 5 is about making use of rank-1 decomposition of our image tensor which we form in a novel way from an individual character image. In Chapter 6, we compare our holistic approach with a popular local feature based approach, namely histogram of oriented gradients (HoG). We demonstrate through experiments that our method is better than HoG, especially when characters may appear in rotated positions.

As an application of our holistic recognition strategy, in Chapter 7 we experiment with the problem of word recognition in natural scene images. In this context, we propose a novel image word segmentation method based on image seam analysis. Our results show promise of better performance and justify our holistic character based recognition.

Finally, in Chapter 8 we conclude discussion of our contribution to the STR research and present prospects of future extensions and improvements.

CHAPTER 2

RELATED WORK

Since the introduction of ICDAR 2003 Robust Reading Competition and the associated challenge datasets (Lucas et al. 2003), the area of scene text recognition has seen an increase in research efforts to solve the various facets of the problem. Different solutions have been proposed for the sub-problems mentioned in Chapter 1. In the following, we take a chronological perspective of the development of several works in STR community, related specifically to scene character recognition (aka robust character recognition) and scene word recognition.

Initially, some researchers integrated off-the-shelf OCR solutions to recognize characters segmented from natural scene images. (Chen and Yuille, 2004) used an adaptive version of Niblack's binarization algorithm (Niblack, 1985) on the detected textual regions and then employed commercial OCRs for final recognition. Their reported results with a commercial OCR engine ABBYY (www.abbyy.com) were good for the dataset they had, which was especially collected from cameras mounted on blind people. However, later performance of ABBYY reported by (Wang and Belongie, 2010) and (de Campos et al., 2009) showed its limited utility on more challenging ICDAR and Chars74K datasets.

Overall, the literature in natural scene character recognition is dominated by local feature-based methods: These methods mainly focus on extracting a feature vector, e.g. a Histogram of oriented Gradients (HoG) (Dalal and Triggs, 2005) or some variant of it, from a character image and then using some classifier, e.g. Nearest Neighbor, SVM, Random Forests etc., to recognize the character. (de Campos et al., 2009) used various feature descriptors in addition to HoG, which

included Shape Contexts (SC), Scale Invariant Feature Transform (SIFT), Geometric Blur (GB), etc. in combination with bag-of-visual-words model. The results, however, showed a lot of room for improvement.

(Donoser et al., 2008) used MSERs in conjunction with simple template matching to get initial character recognition results which were subsequently improved by exploiting web search engines to get final recognition results. (Weinman et al., 2009) used a probabilistic framework wherein they utilized Gabor filters in their similarity model to recognize characters in their dataset. (Smith et al, 2011) also used similarity constraints like (Wienman et al, 2009) but used a different approach to enforce the constraints via integer programming. (Wang and Belongie, 2010) showed better performance than (de Campos et al., 2009) by incorporating HoG features in conjunction with Nearest Neighbor classification.

(Neumann and Matas, 2011) used maximally stable extremal regions (MSER) to create MSER mask and then got features along its boundary which they subsequently used in SVM for classification. (Mishra et al, 2012) used language cues in their hierarchical recognition scheme. (Yi et al, 2013) gave a thorough application of HoGs, both locally and globally, in conjunction with soft-assignment coding, max pooling, and SVMs to give better recognition performance.

In addition to the above, unsupervised feature learning system has been proposed by (Coates et al., 2011) that utilizes a variant of K-means clustering to first build a dictionary then map all character images to a new representation in the dictionary. Unlike the methods mentioned above, their technique relied on using a lot of synthetic data to train their model.

The aforementioned methods rely, one way or the other, on ad hoc rasterization of feature vectors. We believe that using holistic methods would preserve the structural information of a character image which might be lost in rasterizing descriptors in a vector. In this context we put forth three holistic strategies.

The first one is based on modelling character images as a 3-mode tensor and then factoring it into a set of rank-1 matrices and the associated mixing coefficients. Work on rank-1 approximations of tensor has been extensively studied (Kolda and Bader, 2009) and has been applied to recognition tasks (face, action etc.) before, e.g. (Sun et al, 2011). However, its application to scene character recognition is novel and we draw upon the rank-1 tensor decomposition discussed in (Shashua and Levin, 2001) and modify it to adapt to the situation of scene character recognition.

The second approach we discuss in this work lets us form a novel holistic feature based on active contour model, also known as snakes. Active contour was proposed by (Kass et al, 1987) and mostly its application involved image segmentation and recognition. The use of active contour models in shape recognition is common but we are not aware of its application on scene text recognition. Motivated by our holistic feature paradigm, we constrain the shape of the initial contour to a circular design, which in conjunction with cross-correlation based similarity metric, enables us to account for rotational variance in the character image. The closest application to our method is (Yi and Tian, 2014) where the authors establish boundary points using discrete contour evolution during the process of finding character polygons as a first step in getting stroke configurations. Other than this, their approach quite different from ours.

Our third strategy is based on rank-1 tensor decomposition of a 3-mode image tensor. Compared with our first approach, this methods differs in the way we form tensor from a single image via rotation. The closest application to this is in (Tariq and Foroosh, 2015) where the authors use rank-1 decomposition for their image annotation problem. However, in their application, they discard the two spatial components of the factorization and retain just the temporal one. In our case, we retain the spatial components and use the holistic feature in image to class distance metric learning (I2CDML) framework for classification.

Word recognition in natural scene images is a difficult problem. As mentioned in Chapter 1, the problem has two aspects: word spotting (where a lexicon is provided for lookup) and the more general word recognition (also known as lexicon free or open vocabulary word recognition) For four to five years, researchers have come up with their ideas to solve both the versions of the problem by segmenting the input word image into individual characters. (Neumann and Matas, 2011) used vertical projection profiles to get cues for possible character segments in their MSER based framework for word recognition.

(Mishra et al., 2011) proposed an MRF model in an iterative graph cut framework to segment the foreground (text) from the background. (Field and Learned-Miller, 2012) proposed bilateral regression to segment and recognize words. Later, (Mishra et al., 2012a) used bottom up technique to segment word images along with top down cues from language models etc. to do the recognition task.

We utilize and build upon the idea of image seams from content aware image resizing proposed in (Avidan and Shamir, 2007) to bring forth a novel segmentation technique for scene

word images. As part of the segmentation process, we recognize individual characters along the way. The final word recognition is then based on predicting the most probable word using the available spell checking system. Compared to more traditional approaches to segment characters, our method is quite simple yet efficient in that it doesn't rely on sliding windows or other computational intensive modelling techniques, e.g. MRF model in (Mishra et al, 2011).

In Chapters 3, 4, and 5, we discuss our holistic strategies in detail and provide its application to scene word recognition in Chapter 7.

CHAPTER 3

RECOGNITION USING TENSOR SUBSPACE PROJECTION

In this chapter we discuss our first holistic strategy based on getting rank-1 approximations of the character images stacked up in a 3-mode tensor. The rank-1 approximation of a 3-way tensor, e.g. an image cube or an action video, has been shown to be effective in image coding and face recognition in (Shashua and Levin, 2001). (Sun et al., 2011) have used it for action recognition. This chapter is based on our work that was published in the proceedings of the International Conference on Computer Vision Theory & Applications (VISAPP[††] 2015). In the following sections, we start off with a brief discussion of tensor representation and rank-1 approximation of a 3-mode tensor and then give its novel application to our scene character recognition problem.

3.1 Tensor Rank Problem

Consider a set of character images $\{A_i\}$, where $i = 1, \dots, d_3$ and the dimensions of images be $d_1 \times d_2$. Let the images be stacked together as slices of a tensor T whose elements are $T_{g,h,i}$, where $g = 1, \dots, d_1$ & $h = 1, \dots, d_2$.

The following expresses tensor T as the sum of k rank-1 tensors:

$$T = \sum_{m=1}^{k} \lambda^m \otimes u^m \otimes v^m \tag{3.1}$$

[††] http://www.visapp.visigrapp.com

where u, v are the basis vectors and λ represents the mixing coefficients. The problem of determining the smallest k for which Equation 3.1 holds is called the tensor rank problem. Figure 3.1 illustrates the concept of rank-1 decomposition of *T*. For two-image tensor, polynomial time algorithms are available for low rank factorization. However, when the number of slices (or images in our case) of *T* are more than '2' the problem of finding such a superposition of low rank tensors is NP-hard (Hazan et al., 2005). Various algorithms have been proposed to get the rank-1 factors of a multi-image (number of images >2) tensor depending upon how the solution space is constrained. For example, High-Order SVD (HOSVD) (Xianqian and Sidiropoulos, 2001) enforces orthogonality constraints among the basis vectors to get the higher order spectral decomposition. (Shashua and Levin, 2001) give algorithms to the effect of getting desirable SVD like extension to the multi-image tensor decomposition.

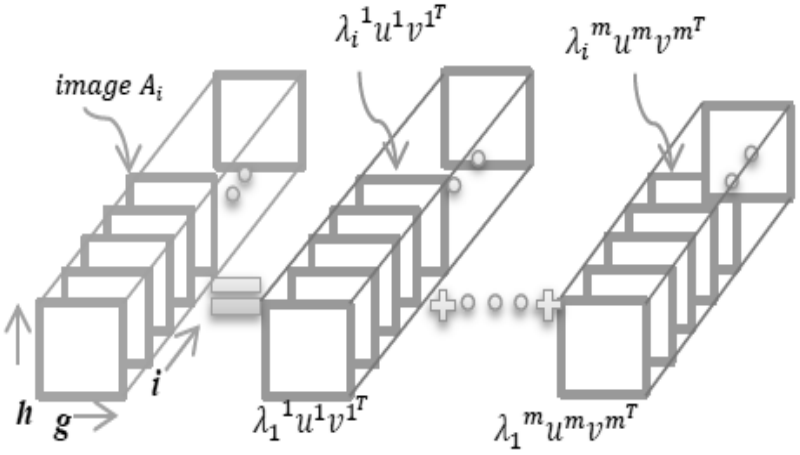


Figure 3.1: An image cube expressed as the sum of 'k' rank-1 matrices

3.2 Getting Rank-1 Matrices for an Image Tensor

We modify the greedy algorithm given in (Shashua and Levin, 2001) to get the rank-1 matrices of scene character image tensors. The iterative algorithm solves the following minimization problem to get the desired unit vectors (rank-1 elements) uv^T and the mixing scalar vector [$\lambda_1, \ldots\ldots, \lambda_p$] associated with each image.

$$\sum_{i=1}^{p} \|A_i - \lambda_i uv^T\|_F^2 \tag{3.2}$$

where $A_1, A_2, \ldots, A_p$ is the given set of images. The steps are summarized below:

1. Create the matrix $S = \sum_{i=1}^{p} A_i A_i^T$ and find the eigenvector corresponding to the largest eigenvalue. This becomes the unit vector u and captures the spatial redundancy in the image set

2. Using u from above, get the eigenvector v corresponding to the largest eigenvalue of the matrix MM^T, where the columns of M are $A_i^T u$. Hence v captures the temporal aspect of character images, e.g. font variations etc.

3. Next, find the scalar λ_i associated with each image as the inner product: $v^T A_i^T u$

4. Compute the residual image as:

$A_i = A_i - \lambda_i uv^T$, replace it with the original image in the set and repeat the above steps until stopping criterion is met

(Shashua and Levin, 2001) do iterative refinement of the vectors u and v in steps 1 and 2 respectively, around the initially estimated location before computing the mixing scalar

coefficients. We avoid this because we empirically found that it exacerbates noise in scene character images and results in performance reduction.

The stopping criteria in step 4 could be the residual falling below a specified threshold or pre-specifying the number of rank-1 elements. We use the latter one and the impact of it on performance is further discussed in Section 3.5.3.

3.3 Recognition Framework for Rank-1 Tensor Subspace Projection

Our framework for scene character recognition, as depicted in Figure 3.2, starts with preprocessing images, followed by tensor decomposition to extract 'k' rank-1 basis matrices that span the training images, projecting test images in each character class (subspace), and finally classifying the test image using inner product of mixing scalar vectors as a similarity measure.

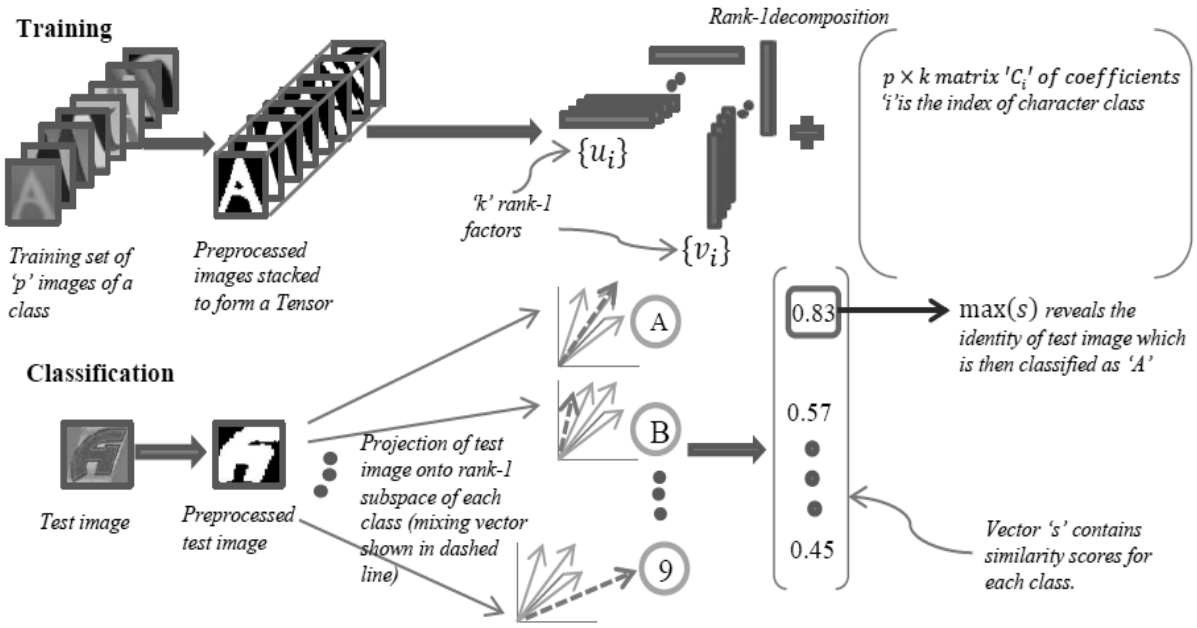


Figure 3.2: Recognition framework based on rank-1 tensor subspace projection

3.3.1 Image Segmentation

The cropped character images from natural scene datasets contain a lot of non-character structures and imperfect cropping artefacts that makes it difficult to effectively capture typeface and shape variations. Since the focus of our work is to demonstrate effectiveness of holistic recognition framework based on rank-1 tensor decomposition, we preprocess each image in training and testing sets to keep the images as noise free as possible. To this end we adopt binarization for image segmentation to reduce noise and extract, possibly only, character structures. This somehow lets us isolate the classification problem from the binarization problem.

Binarization has been used before to segment textual information from natural scene images, e.g. see (Chen and Yuille, 2004), (Mishra et al. 2011), (Kita and Wakahara, 2010), (Field and Learned-Miller, 2013). Binarization of natural scene character images is a challenging problem in its own right. Therefore, for the purpose of this work, we employ a simple and novel combination of the methods of (Yokobayashi and Wakahara, 2005) and (Otsu, 1979) in an effort to segment each image to get textual foreground (in white). Using these methods, we get both the binary image and its inverted version (four images in total). For these binary images, we then perform a connected component analysis based on the observation that cropped characters mostly fall in the middle of the image. We consider any small pixel group as noise if its size is less than a small fraction (<5%) of the size of the largest central connected component.

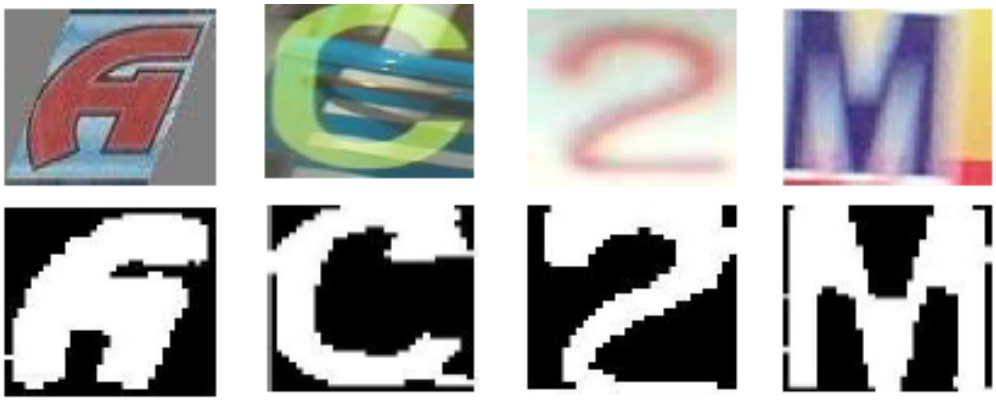


Figure 3.3: Sample segmentation of scene character images

3.3.2 Size Normalization and Image Selection

We then normalize each image following a convex-hull based method given by D'Errico[‡‡]. The intuition is the fact that convex hull contains an edge of a rectangle that bounds an image. We find the one that has the least perimeter. We then try to make the image upright by rotating the bounding rectangle so that its major axis is vertical. This somehow corrects the slant in the input image. Following this, we resize the image to 32×32 pixels.

The four binary images from the above steps contain a potential candidate that we select as a correct binary image. In the case of training, we use a reference image of a class to decide on the correct binary image. The reference image we use is a character (for each class) in Arial font (more on this choice in section 3.4.3) centered in the image using ImageMagick[§§]. To get the 'correctly' binarized image, we subtract each binary image from the reference image and pick the

[‡‡] http://www.mathworks.com.au/matlabcentral/fileexchange/34767. Last visited: 10 December 2014
[§§] http://www.imagemagick.org

one with the least Frobenius norm. For testing we simply check segmentation for all reference images and select the one with the least Frobenius norm. Some results of preprocessing steps are shown in Figure 3.3.

3.3.3 Training

For training, we stack the preprocessed images belonging to each character class to form a mode-3 tensor. For each tensor we then apply rank-1 decomposition with a specified number of rank-1 matrices and we keep this number same across all the classes. Hence, the output of training is the specified number of rank-1 matrices that form the subspace basis for each class along with a set of scalar coefficients yielding the mixing vector for each image in that class. Figure 3.2 top part illustrates the training process. The sensitivity of number of rank-1 elements to the accuracy on test data is discussed in Section 3.4.3

3.3.4 Classification

To classify a test image, we first preprocess it and then project it onto the rank-1 subspace of each character class. The projection here means to get inner product between the rank-1 factors and the test image to get the mixing coefficients by the expression: $\lambda_i = v^T A_i^T u$; where λ_i is the i^{th} mixing coefficient and A_i is the corresponding residual image.

When $i = 1$, A_i = given test image. For $i \geq 2$, $A_i = A_{i-1} - \lambda_{i-1} uv^T$. In this way, we get 62 vectors for each test image (one per each class). We then measure similarity of each test vector using the inner product with the training vectors of each class and record the maximum. The final

classification is given by taking the maximum over all classes. The process is illustrated in the bottom part of Figure 3.2.

3.3.5 Computational Complexity

At each step of the algorithm, we are dealing with S samples of an image class, where each image is normalized to be $n \times n$ pixels. Creating self-similarity matrix $A_i A_i^T$ is the most computationally extensive step in the algorithm that requires $O(n^3)$ operations, where $A_i \in S$.The algorithm then iteratively computes K rank-1 elements for each class, which, depending upon the underlying implementation of the Eigen decomposition, would need $O(n^3)$ in the worst case. Being greedy, the algorithm is guaranteed to converge in linear time to a local minimum (Shashua and Levin, 2001). The overall computational complexity turns out to be $O(Cn^3)$ for generating K rank-1 subspaces from S samples of a class, where $C = SK$ is a constant in our case.

3.4 Experimental Evaluation

We evaluated our approach on three popular scene character datasets Chars74K, ICDAR, and SVT-CHAR. We used various experimental settings to report our results on these datasets. We also compare our method with several baseline methods in scene character recognition.

3.4.1 Datasets and Parameters

We use ICDAR robust character dataset (ICDAR03), English subset of Chars74K-Img (Chars74K) and characters extracted from street view text dataset (SVT-CHAR) in the following experiments.

For all experiments, we report results using 500 rank-1 factors for tensor decomposition. The impact of this particular choice of rank-1 factors on accuracy is further discussed in Section 3.4.3.

3.4.2 Results

In our first experiment, we used the whole ICDAR03 training set to get the rank-1 factors for each class of characters. The accuracy on the test set was 69% (see Table 3.1). In Figure 3.4, the lines parallel to the main diagonal of the confusion matrix reflect ambiguities due to character case, e.g. small case 'c' confused with 'C', etc. (Wang et al., 2012) reported accuracy of 83.9% on a modified version of the ICDAR03 test set, but they re-cropped all images for their experiments and their set contains 5,198 images, which is less than those in ICDAR03 test set. In Table 3.1, we also report our results on the training and test splits proposed by (de Campos et al., 2009) for Chars74K, viz., Chars74K-15, where the suffix '15' specifies the number of training and test samples to be used for the experiment.

Table 3.1: Character recognition performance on ICDAR03 and Chars74K datasets

Method	**ICDAR03**	**Chars74K-15**
GB+NN (de Campos et al., 2009)	41%	47.1%
HoG+NN (Wang & Belongie 2010)	51.5%	58%
SYNTH+FERNS (Wang et al., 2011)	52%	47%
NATIVE+FERNS (Wang et al., 2011)	64%	54%

Method	ICDAR03	Chars74K-15
MSER(Neumann & Matas, 2011)	67%	-
Proposed RANK-1	69%	57.1%

Our second experiment was on SVT-CHAR. Since this is just a test set, therefore, we formed its training by combining the training sets of ICDAR03 and the '*GoodImg*' portion of the Chars74K dataset. We trained our factors on all 62 classes, to fairly compare our results with the reported ones, despite the fact that the SVT-CHAR does not contain any digit classes. We got 64% accuracy and the results are shown in second column of Table 3.2.

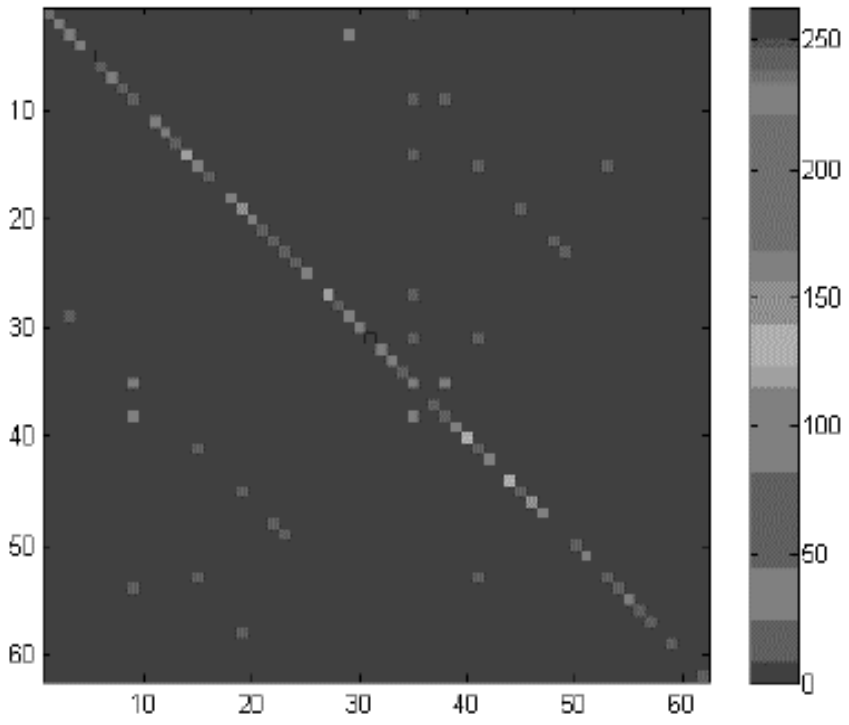


Figure 3.4: Confusion matrix for ICDAR03 test set. Numbers 1-62 show character classes A-Z, a-z,0-9. Lines parallel to the main diagonal show character confusions

Table 3.2: Recognition performance on Chars74K-15 Test Split.

Method	**Chars74K**	**SVT-CHAR**
ABBYY FineReader	31%	15.4%
GB+NN (de Campos et al., 2009)	54.3%	-
Method	**Chars74K**	**SVT-CHAR**
HoG+SVM (Mishra et. al., 2012)	-	61.9%
MSER (Neumann & Matas, 2011)	71.6%	-
Proposed RANK-1	68.5%	64%

In our third experiment, we used Chars74K-15 test split while training on all Chars74K but those images that are in the test set. Column 1 of Table 3.2 shows some improvement in accuracy as compared with other baseline methods and our earlier results given in Table 3.1.

The above results clearly show that our approach does better if given more training samples, which prompted us to do another experiment with leave-random-one-out cross-validation (CV) setting. We show results in Table 3.3 for both Chars74K and ICDAR03. For ICDAR03 we combined training and testing sets to get one big set for CV. The results show median accuracy over 100 trials. The accuracy of 72.5% is the best result we are aware of on the whole Chars74K dataset (including both *GoodImg* & *BadImg* sets). On the other hand the results on ICDAR03 under this setting show improvement and further propound our observation.

Table 3.3: Recognition performance using leave-random-one-out cross-validation (CV).

Method	ICDAR	Chars74K
Proposed RANK-1 + CV	76%	72.5%

In Figure 3.5, we show some test samples from different datasets that our approach correctly recognized. The Figure 3.6 shows the cases where our method failed. Some images here are not even easy human observers to recognize correctly due to low contrast, shape ambiguities, noise etc.

Figure 3.5: Letters correctly recognized by our method

Figure 3.6: Some of the characters that could not be correctly recognized due to shape ambiguities, low contrast, occlusion, imperfect cropping, large rotations etc.

3.4.3 Discussion

3.4.3.1 Case Sensitivity

Cropped characters from natural scene images exhibit extreme shape similarities for some classes. We identified at least ten classes from English alphabet and two digit classes 0 and 1 that can become ambiguous in the absence of contextual clues. Consider Figure 3.7. It is extremely hard even for humans to distinguish between the pair of characters due to case similarity. Digit ‘zero’ and ‘one’ are also sometimes confused with letter ‘O’, ‘I’, and ‘l (ell)’ due to shape similarities. We, however, don’t take into account digit and letter confusion and report only letter case insensitive accuracies in Table 3.4.

Moreover, from a recognition standpoint, the case distinction is immaterial even if eventually we are to recognize words from characters, unless we use case information to mark

word boundaries. Table 3.4 shows that we get accuracy boost over our corresponding results in Tables 3.1 through 3.3.

Figure 3.7: Case ambiguities in natural scene characters. Top row shows upper case while the bottom row shows lower case letters from ICDAR03 dataset

Table 3.4: Recognition performance after removing case sensitivity

Method	**ICDAR03**	**Chars74K-15**
Proposed RANK-1	80%	66%
Proposed RANK-1 + CV	84%	78.7%
Method	**Chars74K**	**SVT-CHAR**
Proposed RANK-1	75.1%	73%

3.4.3.2 Rank-1 Elements and Number of Samples

We give number of rank-1 elements as input to the decomposition algorithm. Figure 3.8 shows the effect of the choice of rank-1 elements on accuracy for ICDAR test set and Chars74K set when tested on the proposed test split by (de Campos et al., 2009). As noted in (Shashua and

Levin, 2001), addition of rank-1 elements helps capture temporal redundancies in the input image set that in our case occur due to font and shape changes. This eventually gets to increased accuracy. However, as shown in Figure 3.8, a point comes after which we get a kind of stagnation in accuracy. We, therefore, empirically fixed the number of rank-1 elements to '500'.

As mentioned in Section 3.4.2 above, the number of training samples per class also played an important role in boosting accuracy on the test set. We empirically observed that unless we add good (or less noisy) images to the training set, the decomposition process would be affected by the presence of even a small number of noisy images. This can be explained by the fact that as the number of less noisy images increases, the additive effect of noisy images is reduced and a good pattern of variation in character's font and shape becomes available and is effectively captured in the spatial and temporal components of the rank-1 decomposition. This is the reason why we used just the '*GoodImg*' part of Chars74K when we combined ICDAR03 and Chars74K to train for SVT-CHAR.

To further demonstrate this fact, we plot in Figure 3.9 the accuracy gain with increasing number of training images for the character class 'A' (this trend is also true for all other character classes). The accuracy here represents performance measured on the test samples of 'A' from (de Campos et al., 2009) test split after training over different number of available training samples of character 'A' from Chars74K. We varied the sample count from '15' (given in de Campos et al. training split) to '659' (all samples of 'A' excluding the '15' given in the test split). The plot validates our observation about gain in accuracy with the increase in number of samples for natural scene character recognition.

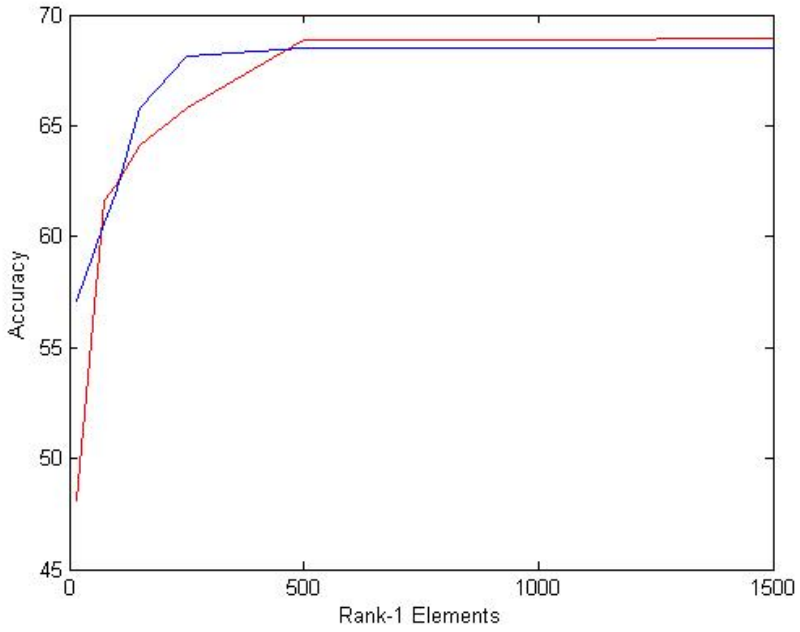


Figure 3.8: Number of rank-1 elements vs. accuracy on ICDAR (shown in red) and Chars74K (shown in blue)

3.4.3.3 Selection of Reference Image for Binarization

We mentioned in section 3.3.1 under image segmentation that we used a reference image for each class produced in Arial font using imageMagick software. However, we didn't elaborate further on our particular choice at that point and before we further discuss this, we should recall that the choice was made during segmentation process to select one binary image out of the available four. This situation of four binary images arose when we used two separate segmentation algorithms to produce a binary image and its complement.

As mentioned before, binarization of scene character images is a hard problem in itself and choosing a particular binary output is not an easy task either. Our workaround for this situation was a heuristic that was based on our observation that characters generally fall in the center of the

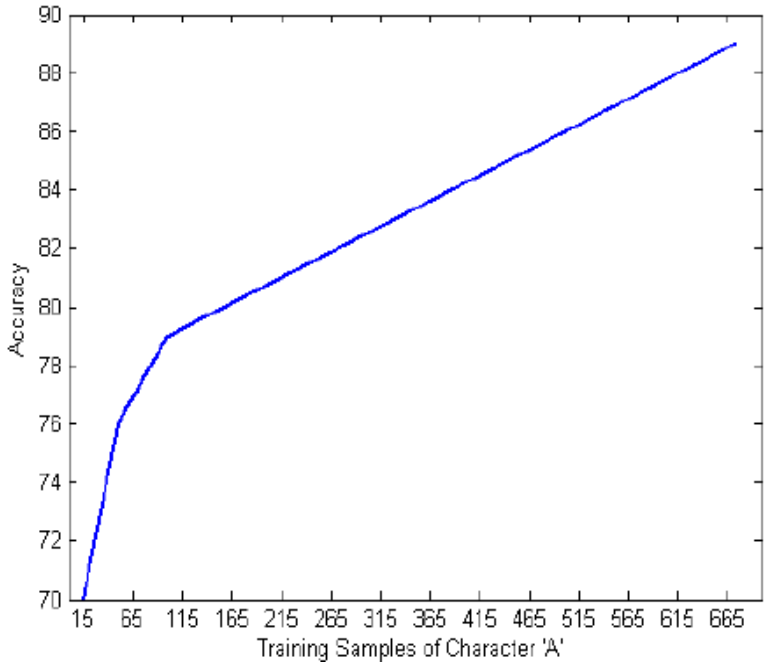


Figure 3.9: Accuracy vs number of training samples of ‘A’ from Chars74K dataset

cropped image region and, therefore, any correct binary output should mostly appear in that central region too.

Moreover, if we somehow put an occluding object, e.g. a synthetic character, in the central region of a candidate binary image, then probably the correct binarization would be the one that overlaps with the object the most. We measured the overlap by subtracting the object from a candidate binary output image and taking the Frobenius norm of the difference. Hence, we came up with the idea of a ‘reference’ object in the form a synthetic image belonging to the class of the training image. For test cases, we simply tried synthetic ‘reference’ images from all 62 character classes and made the selection based on the overall minimum norm value.

While selecting the synthetic reference image, we only considered the font that captured the ‘basic’ shape of a character and choose ‘Arial’ which is a typeface without serifs. However, considering the fact that the datasets we are dealing with already contain a lot of noise and cropping artifacts, it turns out that the contribution of serif structures is really insignificant while taking the norm of the difference in case of selection of binary images. Moreover, this lack of serif structure making some letters ambiguous, e.g. I (capital eye) and l (small ell), is the concern of the recognition model. As an example, consider Figure 3.10 that depicts an input image from ICDAR03, two reference images in Arial and Times New Roman fonts, along with the four candidate binary images. If we look closely, we find that both reference images would output the correct binary image and that the presence or absence of serifs would not make any difference vis-à-vis the selection of the correct binary image.

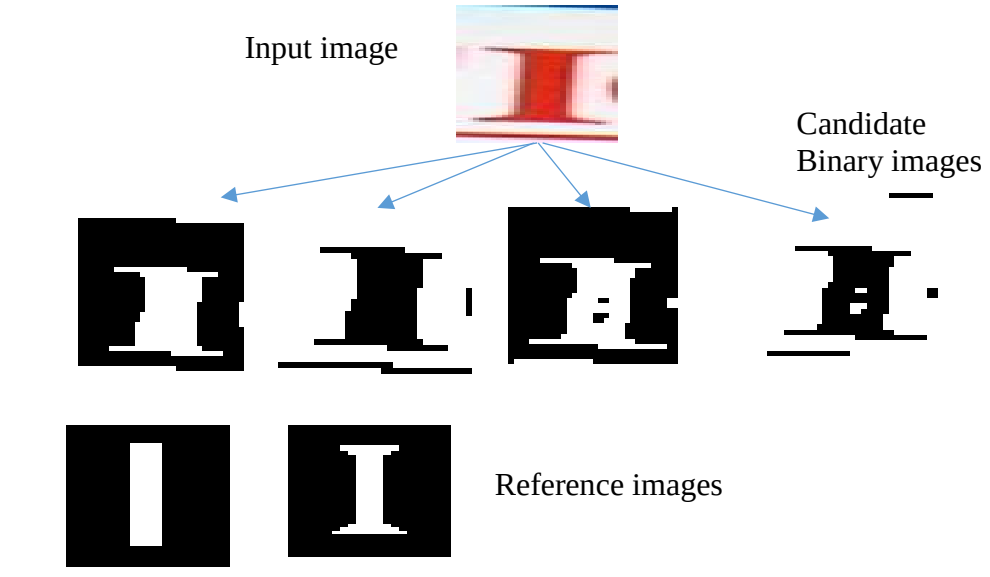


Figure 3.10: Correct binary image selection. (Top row) Input image. (Middle row) Candidate binary images. (Bottom row) Reference images. Both reference images select the correct binary image (first from the left)

Nevertheless, we re-ran experiments on our datasets to show the impact of serifs. For this experiment we used Times New Roman font to compare with our earlier results on Arial font in Table 3.1. The results of this comparison are shown in Table 3.5. Slight difference in accuracy may be attribute to noise in the datasets. Overall, the impact of a font having serifs is minimal.

Table 3.5: Impact of Reference Font on Accuracy

Method	**ICDAR03**	**Chars74K-15**
Proposed RANK-1 + Arial Font	69%	57.1%
Proposed RANK-1+ Times New Roman Font	68.47%	57.3%

3.4.3.4 Pros and Cons

Pros:

1. The strategy avoids dependency on local features and capture holistic signature that describes an image class very well.

2. The recognition framework is simple compared with the complex pipelines of the methods compared with in results.

Cons:

1. Images need to be somewhat aligned during pre-processing stage. We did it by making them upright before stacking them into the tensor.

2. Noise needs to be taken care of to improve accuracy on the scene datasets.

3.5 Conclusion

We proposed a holistic approach to solve natural scene character recognition that avoids dependency on specific features. Our method is based on multi-image tensor decomposition similar to (Shashua and Levin, 2001) with modification as to the way we get rank-1 matrices for natural scene images that contain a lot of typeface variations and noise. Through our results we showed the potential of using image tensor decomposition to better capture shape and font variations in scene character images. We got better results than several baseline methods and achieved improved recognition performance on the datasets using leave-random-one-out cross-validation, justifying thus our intuition of the importance of feature-independency and the benefit of preserving spatial correlations in recognition.

Given better segmentation methods, the accuracy of our method could further be improved. Moreover, we can employ other tensor decomposition methods so that we don't need to make a character upright before finding rank-1 matrices (see Chapter 5 for more details).

CHAPTER 4

RECOGNITION BASED ON ACTIVE CONTOUR MODEL

In the previous chapter, we discussed our first holistic strategy based on rank-1 matrices to recognize characters in the image patches cropped from natural scene images. However, in that strategy character rotation was handled in an ad hoc manner in the preprocessing step by making it upright based on a heuristic. In this chapter, we will discuss an approach based on active contour model that will be rotation invariant by its design and application. The contents in this chapter are based on our work that was published in the proceedings of International Conference on Computer Vision Theory and Applications (VISAPP[***] 2016).

Our contribution here is thus twofold:

1. A novel feature vector based on active contour model

2. Rotation invariance

The use of active contour models in shape recognition is not new but we are not aware of its application specifically for the scene text recognition. The closest application we found was in (Yi and Tian, 2014) where the authors establish boundary points using discrete contour evolution during the process of finding character polygons as a first step in getting stroke configurations. Other than this, their approach quite different from ours.

[***] http://www.visapp.visigrapp.com

The results we get show that the proposed method effectively captures character shape variations occurring in natural scene images in a holistic manner thus avoiding the problems associated with techniques based on local image features.

4.1 Motivation

An active contour is a dynamic object (curve), which evolves to wrap around an object boundary. This idea of capturing object shape motivates us to use it to extract holistic feature for our character recognition problem.

Active contour models have been used for image segmentation and shape description since long, e.g. see (Kass et al. 1987), (Xu and Prince, 1997), (Xu and Prince, 1998). In the following section, we first briefly recap the basics mostly following notations and derivations in (Ivins and Porrill, 2000) and then move on to give its novel application to deriving our holistic feature for characters synthetic or extracted from natural scene images.

4.1.1 Basics

Consider a 2D image. An active contour model can be described as a closed loop of points or pixels. It is also called a snake for its movement in image plane. Mathematically, a snake is a set of points in the image plane. It can be characterized by the following parametric curve:

$$u(s) = \big(x(s), y(s)\big) \tag{4.1}$$

where, x and y are the coordinates of pixels and s is the parameter. As described in (Ivins and Porrill, 2000), we can associate an energy functional E with this curve (note that the curve is a loop in this case).

$$E(u) = \oint P(u) + \alpha(s)|u'|^2 + \beta(s)|u''|^2 \, ds \tag{4.2}$$

where, $P(u)$ is the external image energy (derived mainly from image gradients) and α and β are the internal curve parameters for tension and stiffness respectively.

Assuming $\alpha(s)$= $\boldsymbol{\alpha}$, and $\beta(s)$ = $\boldsymbol{\beta}$ as constants, the minimization of Equation 4.2 can be done by satisfying two independent Euler equations in u :

$$\boldsymbol{\beta} u'''' - \boldsymbol{\alpha} u'' = -\frac{d(P)}{du} \tag{4.3}$$

Following (Ivins and Porrill, 2000), the derivatives in Equation 4.3 can be approximated by finite differences as follows:

$$\boldsymbol{\beta}(x_{s-2} - 4x_{s-1} + 6x_s - 4x_{s+1} + x_{s+2}) - \boldsymbol{\alpha}(x_{s+2} + x_{s-2} - 2x_s) = f_x(x, y) \tag{4.4}$$

$$\boldsymbol{\beta}(y_{s-2} - 4y_{s-1} + 6y_s - 4y_{s+1} + y_{s+2}) - \boldsymbol{\alpha}(y_{s+2} + y_{s-2} - 2y_s) = f_y(x, y) \tag{4.5}$$

where f_x and f_y are the components of image force (computed from gradients in 'x' and 'y' direction). To solve Equations 4.4 and 4.5, we use the semi-implicit method discussed in (Ivins and Porrill, 2000) where two sets of finite difference equations are formed to describe the *x* and *y* coordinates of the entire snake; these equations can be written in terms of a *cyclic symmetric pentadiagonal banded matrix* **M** incorporating the constants $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ as follows:

$$\mathbf{M}.\mathbf{x} = \mathbf{f}_x\,(\mathbf{x},\mathbf{y}) \;\; ; \; \mathbf{M}.\mathbf{y} = \mathbf{f}_y\,(\mathbf{x},\mathbf{y}) \tag{4.6}$$

where, **x** and **y** are vectors containing the x and y coordinates of all the snake elements; $\mathbf{f}_x$ and $\mathbf{f}_y$ are the corresponding vectors of image forces acting on contour points. We can solve these equations iteratively by using a discrete and small time step 'τ' (we set $\tau = 1$ for our experiments). The stiffness and tension constraints are applied at time *t*+1 after adjusting the snake according to the image forces at time *t*:

$$\mathbf{x}_{t+1} = (\mathbf{M} + \tau\mathbf{I})^{-1}.(\mathbf{x}^t + \tau\mathbf{f}_x(\mathbf{x}^t, \mathbf{y}^t) \tag{4.7}$$

$$\mathbf{y}_{t+1} = (\mathbf{M} + \tau\mathbf{I})^{-1}.(\mathbf{y}^t + \tau\mathbf{f}_y(\mathbf{x}^t, \mathbf{y}^t) \tag{4.8}$$

The matrix inversion in the above Equations 4.7 and 4. 8 is taken only once because they are composed of constant terms. Hence, the above iterative minimization provides for a fast way to solve the Equation (4.2).

We compute the external image energy, $P(u)$, by taking a weighted combination of three factors: lines, edges, and corners. Thereafter, we compute the effects of external forces on each point of the contour by interpolation. The new coordinates of contour points are computed from Equations 4.7 and 4.8 above. To further expand the reach of image forces and let the snake enter concave regions, we also utilize gradient vector flow as mentioned in (Xu and Prince 1998).

4.1.2 Feature Vector

As we evolve the snake around character images and compute new coordinates of each point at each time step of the above iterative minimization, we compute the following:

$$\Delta d_t = \sqrt{\Delta x_t{}^2 + \Delta y_t{}^2} \tag{4.9}$$

$$\Delta \theta_t = tan^{-1}\left(\frac{\Delta y_t}{\Delta x_t}\right) \tag{4.10}$$

where, $\Delta x_t = x_t - x_{t-1}$and $\Delta y_t = y_t - y_{t-1}$. The quantities $\Delta d_t \; and \; \Delta \theta_t$ represent the distance and angle increments respectively for each point (pixel) on the contour at time instant t. We accumulate the increments over the course of evolution for all points p and concatenate to form our descriptor as follows:

$$f(im) = \begin{pmatrix} (\sum_{t=1}^{n} \Delta d_t) p_1 \\ \vdots \\ (\sum_{t=1}^{n} \Delta d_t) p_m \\ (\sum_{t=1}^{n} \Delta \theta_t) p_1 \\ \vdots \\ (\sum_{t=1}^{n} \Delta \theta_t) p_m \end{pmatrix} \tag{4.11}$$

where, $f(im)$ is the final feature vector, and n is the number of iterations of the contour evolution. The size of the above feature vector is equal to twice the number of contour points m (i.e., $2m$ $=125 \times 2 = 250$ in our experiments).

The rotational invariance property of the above feature vector follows from its design (the initial contour is circular) and its use in conjunction with the cross-correlation similarity metric.

4.2 Snake Based Character Recognition Framework

Our framework for scene character recognition starts with pre-processing images, followed by training where we generate feature vector using active contour evolution , as depicted in Figure

4.1, for all training samples, and finally classification by getting the feature vector for each test image and computing similarity using cross-correlation.

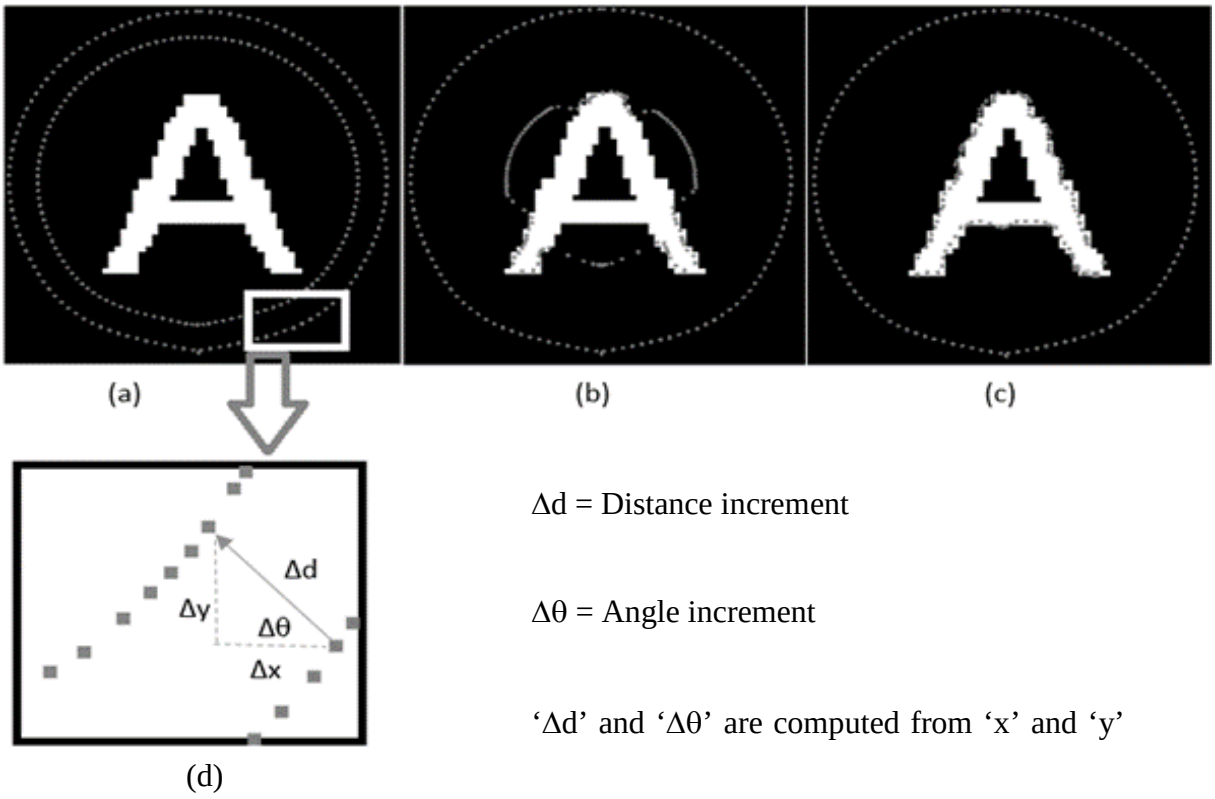


Figure 4.1: Illustration of our framework for feature vector extraction. Outer loop shows the location of pixels in the initial contour and inner loops show the evolution of the initial contour after some iterations. (a) Contour evolution after 50 iterations (b) After 125 iterations (note the wrapping of contour around the character (c) Final contour after 200 iterations (d) Enlarged view of the points and computation of distance and angle increments

4.2.1 Textual Foreground Extraction

Segmentation of textual information from natural scene images is a challenging problem due to noise and distortions introduced mainly by uncontrolled imaging conditions. Many researchers have attempted to tackle it, e.g. see (Chen and Yuille, 2004), (Mishra et al. 2011), (Kita

and Wakahara, 2010), (Field and Learned-Miller, 2013). Since we are primarily concerned with the problem of character recognition, we adopt a simple heuristic based method to get the correct textual foreground (in white).

To this end, we obtain two binary images: one from the output of Otsu's method and the other by inverting it. At this point, we do a simple analysis of the skeleton by counting number of pixels of both images to get the correct segmented image. We then perform a connected component analysis based on the observation that cropped characters mostly fall in the middle of the image. We consider any small pixel group as noise if its size is less than a small fraction (<5%) of the size of the largest central connected component. The process is shown as a flow chart in Figure 4.2.

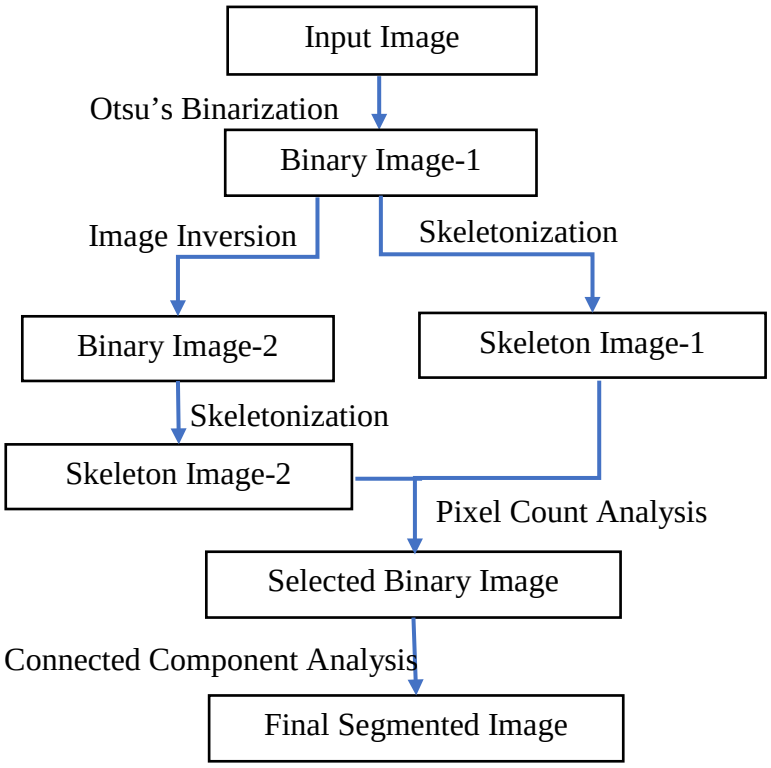


Figure 4.2: Segmentation of character images

After segmentation, we normalize each image to have the size of 32×32 pixels. To make sure that curve evolution doesn't start too close to the character boundary, we pad the image with a 16×16 frame of zeros. Hence the final size of the image becomes 64×64.

4.2.2 Getting Snake Features

For training, we take individual images of each class and pre-process them. For each image, we envelope the character with a circular contour (radius of contour is fixed at 30 pixels) centered on the image and sampled uniformly with 125 points. The contour is then evolved towards the character and for each point on it, we accumulate direction (angle) and distance, until the character boundary is reached. The two quantities (direction and distance) are then concatenated and normalized to form a feature vector containing 250 elements.

Figure 4.1 illustrates the process of extracting the feature vector for a random image in the training process. The sensitivity of results for different parameter settings is discussed in Section 4.3.3.

4.2.3 Classification

To classify a test image, we first pre-process it and then compute its feature vector by evolving a circular contour towards it. We then employ Nearest Neighbor classifier to find the class of the test image by measuring similarity of the test vector with each of the training vectors using the cross-correlation metric and recording the maximum. The final classification is given by taking the maximum over all classes.

4.2.4 Computational Complexity

The most important step in the feature extraction phase is the curve evolution, where we deal with N contour points laid out in a circle and each one is described with x and y coordinates. So we have $2N$ individual quantities to take care of while evolving the contour. Each movement of the contour depends on computing external and internal forces acting on each point. Luckily, the iterative optimization method we used in modelling snake evolution uses cyclic pentadiagonal matrix, which can be inverted in $O(N)$ steps instead of $O(N^3)$. Moreover, by assuming the contour parameters α and β to be constant, we need just one computation involving matrix inversion. Finally, we iterate over g evolution steps to get the snake to its terminal shape. Since in our implementation g is fixed beforehand, the final runtime complexity turns out to be $O(gN)$.

4.3 Experimental Evaluation

We evaluated our approach using various popular character datasets and performed experiments using different settings to report our results on the datasets. We also compare our method with several baseline methods.

4.3.1 Datasets and Parameters

We use Chars 74K-Font, Chars74K-Img (Chars74K), and the ICDAR (ICDAR03) datasets in the following experiments.

For all experiments, we fixed the parameters α and β to the value of 0.05, the number of iterations to 200, the radius of initial contour to 30 pixels. This parameter setting yield good results across all datasets. Further discussion on this setting is deferred until Section 4.3.3.

4.3.2 Results

In our first experiment, we randomly picked 15 samples from Chars74K-Font dataset, each for training and testing to make fair comparison with the results of de Campos et al. The results are shown in Table 4.1. The second column of Table 4.1 presents interesting results when training on synthetic fonts and testing on Chars74 test split proposed in (de Campos et al., 2009). Here also, we show better performance than the reported method.

Table 4.1: Character recognition performance on Chars74K-Font dataset & Chars74K-15 Test Split

Method	**Chars74K-Font**	**Chars74K-15 Test Split**
GB+NN (de Campos et al., 2009)	69.71%	47.16%
Proposed Active Contour	71%	56%

In our second experiment, we used the whole ICDAR03 training set to get features for each class of characters. The accuracy on the test set was 62% (see Table 4.2). Although, the accuracy is better than or as well as other baseline methods, yet it is lower than the rank-1 tensor method discussed in the previous chapter. The possible reasons for this lower accuracy will be discussed later in section 4.3.3. However, we get better results (about 2%) on Chars74K-15 than before. This may be attributed to presence of more rotated images in Chars74K dataset than ICDAR03 and our system being better dealing with it than the rank-1 method.

Table 4.2: Character recognition performance on ICDAR03 and Chars74K-15 datasets

Method	**ICDAR03**	**Chars74K-15**
GB+NN (de Campos et al., 2009)	41%	47.09%
HoG+NN (Wang & Belongie 2010)	51.5%	58%
SYNTH+FERNS (Wang et al., 2011)	52%	47%
NATIVE+FERNS (Wang et al., 2011)	64%	54%
Stroke Configuration(Yi and Tian, 2014)	62.8%	60%
Proposed Active Contour	62%	59%

Our third experiment was to test the performance of Chars74K-15 test split on a modified training set (using all training samples but those in the test split) as per de Campos et al. The results are reported in Table 4.3.

Table 4.3: Recognition performance on Chars74K-15 Test Split

Method	**Chars74K-15 Test Split**
ABBYYFineReader (www.abbyy.com)	31%
GB+NN(de Campos et al., 2009)	54.3%
Proposed Active Contour	61.5%

The difference in results of Table 4.2 (2nd column) and Table 4.3 clearly show, as observed before in Chapter 3, that the number of training samples in the former (15 in this case) can be increased to better capture the variation in test samples. Hence, as before, we did another

experiment with leave-random-one-out cross-validation (CV) setting. We show results in Table 4.4 for both Chars74K and ICDAR.

For ICDAR, we combined training and testing sets of ICDAR03 to get one big set for CV. For Chars74K, as mentioned before, the data already comes without training and testing splits. The results in Table 4.4 show median accuracy over 100 trials.

Table 4.4: Recognition performance using leave-random-one-out cross-validation (CV).

Method	**ICDAR**	**Chars74K**
Proposed Active Contour + CV	65%	62%

4.3.3 Discussion

4.3.3.1 Parameter Sensitivity

In Figure 4.3, we show how the accuracy changes with respect to the snake's parameter values. The parameters α and β are internal smoothness coefficients and control how the contour behaves as it evolves over the given generations. The parameter α represents tensile behavior of the contour while β represents stiffness. Here we assume $\alpha = \beta$.

To estimate the values of the parameters, we perform experiment on the Chars74K-Font dataset and compute accuracy over different values of α and β spread over an arithmetic scale from 0.001 to 0.01. We found that the best accuracy occurs at $\alpha = \beta = 0.05$. Hence, we used this value throughout our experiments.

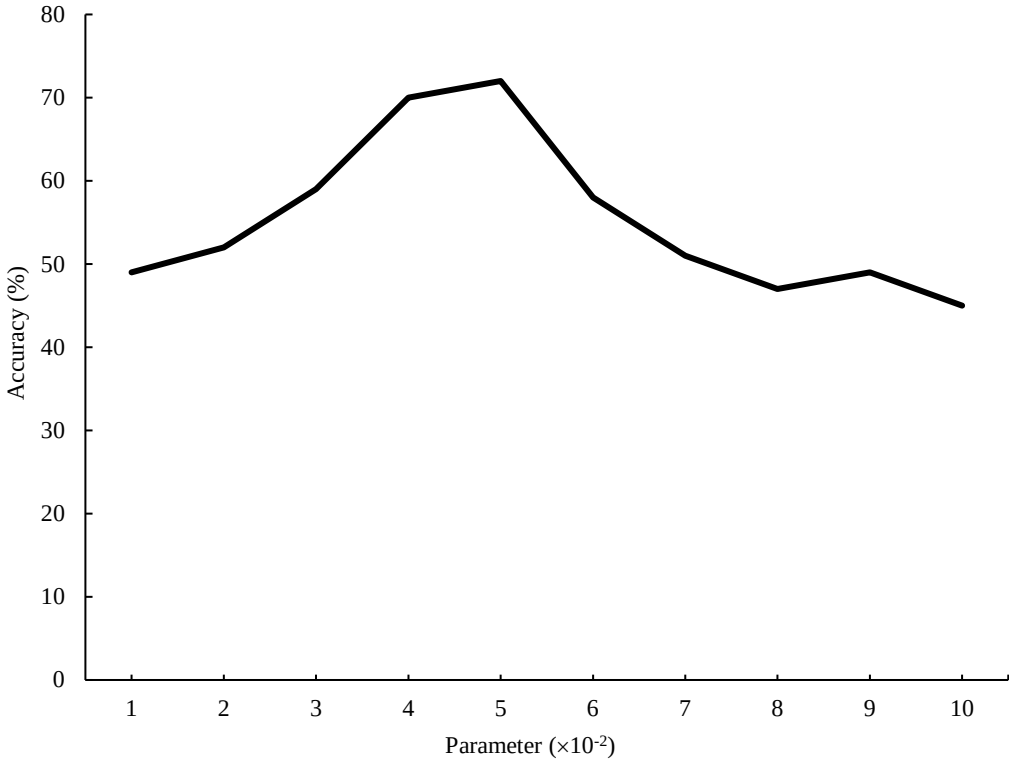


Figure 4.3: Snake internal parameters' sensitivity estimated on Chars74K-Font dataset. We assume parameters $\alpha=\beta$ in this case and the best value occurs at $\alpha = \beta = 0.05$

4.3.3.2 Accuracy

Our system performed better than several baseline methods. However, in comparison with our rank-1 tensor approach, the accuracy of the active contour based method is lower. Table 4.5 below shows the comparison of the accuracy values for the two methods on ICDAR and Chars74K datasets. The possible reasons are as below:

Table 4.5: Comparison of performance of our rank-1 and active contour based approaches

Method	**ICDAR03**	**Chars74K-15**
Proposed RANK-1	69%	57.1%
Proposed Active Contour	62%	59%
Method	**ICDAR**	**Chars74K**
Proposed RANK-1 + CV	76%	72.5%
Proposed Active Contour + CV	65%	62%

1. Noise

Among other factors, the noise in the images influenced the results the most. This is obvious from the results shown in Table 4.1 where our system performed reasonably well on the synthetic Chars74K-Font dataset because of the absence of many noisy artifacts from the images. This situation helped contour evolution get a good feature descriptor for an image. Although we did segment the text in pre-processing step, yet we find that that the performance improvement can be achieved only if a more elaborate algorithm is used for segmentation/noise removal.

2. Classifier

Choice of the classifier may also have played a role in lowering the accuracy. In the case of our rank-1 tensor method, we project a test image using the subspace of a class and then record the difference (similarity) between mixing coefficients of the test image and each members of that

class. While in active contour method, we used 1-Nearest Neighbor classifier. Hence to improve performance, more robust classifiers would help.

4.3.3.3 Pros and Cons

Pros:

1. Rotation invariant feature design help us avoid extra (sometimes erroneous) step of making a character upright in the preprocessing stage.

2. Our novel feature effectively captures intra-class variations in scene characters via simple and intuitive way compared with costly local feature based recognition pipelines, e.g. (Yi and Tian, 2014).

Cons:

1. Contour evolution is sensitive to the presence of noise and other cropping artifacts that are present in natural scene datasets. To minimize this, we used binary segmentation but it's still a performance bottleneck.

2. Some character classes in English alphabet are rotationally symmetric, e.g. 'M' and 'W'. This introduces ambiguity in classification. It should, however, be noted that any system that claims to take care of rotational invariance in this context, would suffer from this inherent problem.

4.4 Conclusion

In this chapter, we put forth a novel feature to holistically solve natural scene character recognition problem based on the active contour model. We managed to mitigate the issue of rotational variance in the previous rank-1 method with our direction and distance tracking of circular contour points. Through our results we showed the potential of using our feature to better capture rotation and font variations in scene character images than various local feature based methods (more on this in Chapter 6). We got better results than several baseline methods and achieved improved recognition performance on the datasets using leave-random-one-out cross-validation, and, thus, showed the importance of holistic features in the context of feature-independency and preservation of spatial correlations in recognition.

In the next chapter, we discuss another holistic feature that combines the strengths of our rank-1 and active contour based features.

CHAPTER 5

RECOGNITION USING RANK-1 TENSOR DECOMPOSITION

In Chapter 3, we discussed the factorization of a 3-mode image tensor by varying certain aspects of the algorithm proposed by (Shashua and Levin, 2001). Therein, we stacked preprocessed images of a class from the datasets into a tensor. Although it makes sense to put all training samples of a class into a single tensor and then decompose it, yet given the drastic variations in typeface and other aspects of character images from one slice of the tensor to the next, we observed that this introduces quite a bit noise in the temporal component of the rank-1 elements (vector '*v*') of a single class. This chapter is based on our work that was accepted in the IEEE International Conference on Image Processing (ICIP[†††] 2016).

5.1 Modifying the Way to Compose Image Tensors

To ameliorate the performance bottleneck for the aforementioned reason, we propose to compose an image tensor from a single image. For action recognition, as in (Sun et al, 2011), the third dimension is the temporal axis of the videos. In our case, we can 'create' temporal dimension by successively rotating a given image through predefined range of angles. This would not only make the frame to frame transitions smooth but also could take care of the rotation variance to some degree.

[†††] 2016.ieeeicip.org

In conjunction with the above, we intend to utilize a better variant of the nearest neighbor classifier to account for the accuracy slump discussed in Chapter 4.

The method in (Tariq and Foroosh, 2015) also uses rank-1 tensor decomposition based on Tucker for automatic image annotation problem but unlike our method, the authors only use one temporal vector for getting context of an image, discarding the vectors that encode spatial information of the image for their problem. Rank-1 decomposition based on Tucker method is also used by (Sun et al, 2011) for action recognition.

In this chapter, we discuss a novel approach to form the 3-mode tensor by rotating the given training/test image through a range of angles. The resulting tensor is then decomposed using rank-1 Tucker decomposition (Kolda and Bader, 2009) to get holistic feature descriptor for the character image. We use image-to-class distance metric learning (Wang et al., 2010) to train and then test the images in popular scene text datasets. The results we get show the effectiveness of the proposed method to capture character shape variations occurring in natural scene images in a holistic manner, thus avoiding the problems associated with ad hoc rasterisation of local image features

5.2 Tensor Representation and Rank-1 Decomposition

Our motivation to use holistic approach is to do away with the dependency on local features. Tensors afford a natural and holistic representation for videos, images, etc. and have successfully been used in the context of image and video understanding, e.g. see (Shashua and Levin, 2001), (Tariq and Foroosh, 2015), (Sun et al., 2011) etc. In the following, we first describe our novel tensorial representation for characters cropped from natural scene images following the

notations in (Kolda and Bader, 2009). We then discuss the distance metric learning algorithm used in our method.

As mentioned in Chapter 3, a tensor can be thought of as a multi-dimensional array. An N-way tensor (N^{th} order or mode) is an element of the tensor product of N vector spaces, each of which has its own coordinate system (Kolda and Bader, 2009). As we already know that an image stack (or cube) can also be considered a 3-mode tensor $\mathcal{T} \epsilon \mathbb{R}^{U \times V \times W}$, with U, V being spatial dimensions and W the temporal dimension. While the temporal dimension is well defined for videos, it is not so obvious for a single image.

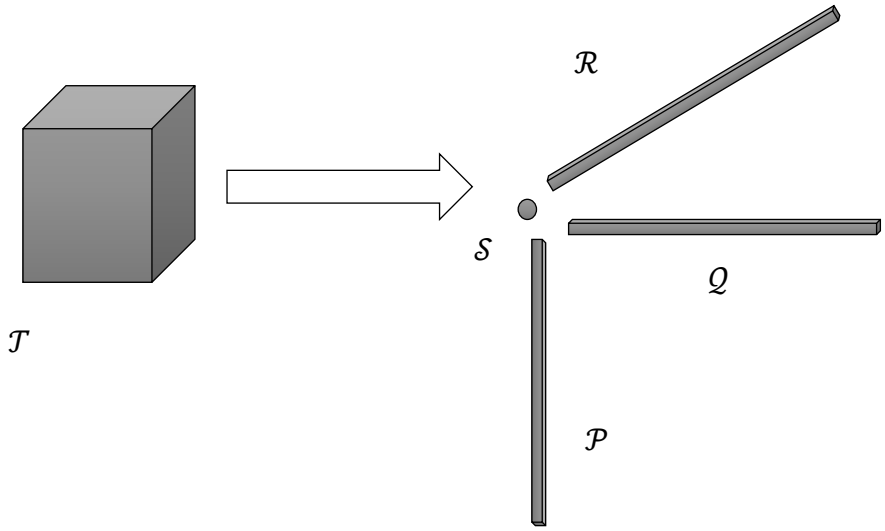


Figure 5.1: Illustration of Tucker decomposition of a 3-mode image tensor $\mathcal{T}$. $\mathcal{P}, \mathcal{Q}, \mathcal{R}$ are the resulting vectors and $\mathcal{S}$ is the scalar

To get around this issue, we propose a novel way to form a 3-mode tensor for an image by taking its rotations over a predefined range of angles ($1 \leq \theta \leq 180$ in our case) to create the 'temporal' or 3rd dimension. Before the formation of tensor for a train/test image, we first pre-

process it to segment its foreground textual content. To this end, we adopt the segmentation method in Chapter 4 and then resize the resulting image into 32×32 pixels.

Tucker decomposition (Kolda and Bader, 2009) is then applied to project the image tensor onto its modes. In general, for a tensor $\mathcal{T} \epsilon \mathbb{R}^{U \times V \times W}$, the Tucker decomposition approximates it as follows:

$$\mathcal{T} \approx \mathcal{S} \times_1 \mathcal{P} \times_2 \mathcal{Q} \times_3 \mathcal{R} = \sum_{x=1}^{X} \sum_{y=1}^{Y} \sum_{z=1}^{Z} s_{xyz} p_x \circ q_y \circ r_z \tag{5.1}$$

where, $\mathcal{P} \epsilon \mathbb{R}^{U \times X}$, $\mathcal{Q} \epsilon \mathbb{R}^{V \times Y}$, and $\mathcal{R} \epsilon \mathbb{R}^{W \times Z}$ are the orthogonal factor matrices, $\mathcal{S} \epsilon \mathbb{R}^{X \times Y \times Z}$is the core tensor and the operator $\times_i$ denotes multiplication between a tensor and a vector in mode-i of that tensor. p_x, q_y, r_z and s_{xyz} in the expression after equality in Equation 5.1 represent, respectively, the components (columns) of the factor matrices $\mathcal{P}, \mathcal{Q}, \mathcal{R}$ and the elements of core tensor $\mathcal{S}$.

When using the Tucker decomposition, if we let X=Y=Z= K, where $K < \min(U, V, W)$, then the tensor $\mathcal{S} \epsilon \mathbb{R}^{K \times K \times K}$ can be thought as a compressed version of $\mathcal{T}$ (Kolda and Bader, 2009). We set $K = 1$ to get rank-1 decomposition that yields three vectors $\mathcal{P}, \mathcal{Q}, \mathcal{R}$ and a scalar $\mathcal{S}$. We concatenate the three vectors to get our final feature descriptor for each train/test image.

5.3 Image-to-Class Distance Metric Learning

To classify the character images based on our holistic feature, we use a variant of image-to-class distance metric learning (I2CDML) proposed by (Wang et al., 2010). The image-to-class distance effectively takes into account large intra-class variations, like in the case of scene characters.

In the following, we use the notations and problem formulation sequence in (Wang et al., 2010) to make the description of the classifier clear.

Let an image X_i which is represented by a feature vector $f_i \epsilon \mathbb{R}^d$. To compute I2C distance to a candidate class c, we find the nearest neighbor (NN) of f_i in c, denoted as f_i^c. The original I2C distance is then defined as follows:

$$Dist(X_i, c) = \| f_i - f_i^c \|^2 \tag{5.2}$$

We then learn Mahalanobis metric $M_c \epsilon \mathbb{R}^{d \times d}$ for each class and replace the original I2C distance in Equation 5.2 by the following:

$$Dist(X_i, c) = (f_i - f_i^c)^T M_c (f_i - f_i^c) \tag{5.3}$$

Given a test image represented by its feature vector f_t, we find the NN in a candidate class c, denoted by f_t^c, and classify it using the following:

$$ClassLabel = \underset{c \epsilon \{1,2,\dots c\}}{\arg min} (f_t - f_t^c)^T M_c (f_t - f_t^c) \tag{5.4}$$

To learn the metrics for each class, the idea proposed in (Wang et al., 2010) is to make the distance of an image X_i from its true belonging class p should be smaller than to any other (non-belonging) class n:

$$Dist(X_i, n) - Dist(X_i, p) \geq 1 \tag{5.5}$$

The objective function is, therefore, composed of two terms: regularization and error. For the regularization term, all positive distances need to be minimized, while keeping all negative distances away from the positive ones by a large margin (analogous to the SVM). Following (Wang et al., 2010), a slack variable ξ is introduced in the error term to allow for soft margin. Hence, the whole optimization problem can be formulated as:

$$\min_{M_1, M_2, , , , M_C} O(M_1, M_2, , , , M_C) = (1-\lambda) \sum_{i,p \to i} \left(f_i - f_i^p\right)^T M_p \left(f_i - f_i^p\right) + \lambda \sum_{i,p \to i, n \to i} \xi_{ipn} \tag{5.6}$$

$subject\ to$:

$$\forall i, p, n: (f_i - f_i^n)^T M_n (f_i - f_i^n) - \left(f_i - f_i^p\right)^T M_p \left(f_i - f_i^p\right) \geq 1 - \xi_{ipn}$$

$$\forall i, p, n: \xi_{ipn} \geq 0$$

$$\forall c: M_c \succeq 0$$

The optimization problem in Equation 5.6 is an instance of semi-definite program (SDP) and we use the gradient descent based implementation provided in (Wang et al., 2010) to solve it and get the metrics M_c for each of our character class.

5.4 Computational Complexity

The rank-1 tensor decomposition strategy we use is based on the popular method called Tucker decomposition. The algorithmic implementation that we used for Tucker decomposition algorithm is based on alternating least square (ALS). It's an iterative algorithm that is not

guaranteed to converge at a global minimum. If, however, it converges, it takes (N) , where N is the number of iterative steps.

5.5 Experimental Evaluation

We evaluated our approach with various character datasets and performed experiments using different settings to report our results on those datasets. We also compare our method with several baseline methods.

5.5.1 Datasets and Parameters

The datasets we use in the following experiments are ICDAR03 and Chars74K. As in the previous chapter, we use its both versions, synthetic (Chars74K-Font) and English subset (Chars74K-Img).

For all experiments, we fixed the parameters for I2CDML following the implementation in (Wang et al., 2010): λ =0.95, α (this parameter controls the convergence speed to find M_c) is increased by a factor of 1.01 if the objective function decreases and is decreased by a factor of 0.5 if it increases. The maximum number of iterations in the optimization is fixed to 500.

5.5.2 Results

In the first experiment, we used Chars74K-Font synthetic font dataset. We randomly picked 15 samples each for training and testing to make fair comparison with the results in (de Campos et al., 2009). The results are shown in Table 5.1. The second column of this table presents interesting results when training on synthetic fonts and testing on Chars74K-Image test split

proposed in (de Campos et al., 2009). Here also, we show better performance than the reported method.

Table 5.1: Recognition performance on Chars74K-Font dataset & Chars74K-15 Test Split.

Method	**Chars74K-Font**	**Chars74K-15 Test Split**
GB+NN (de Campos et al., 2009)	69.71%	47.16%
Proposed Rank-1 Decomp.	73%	56%

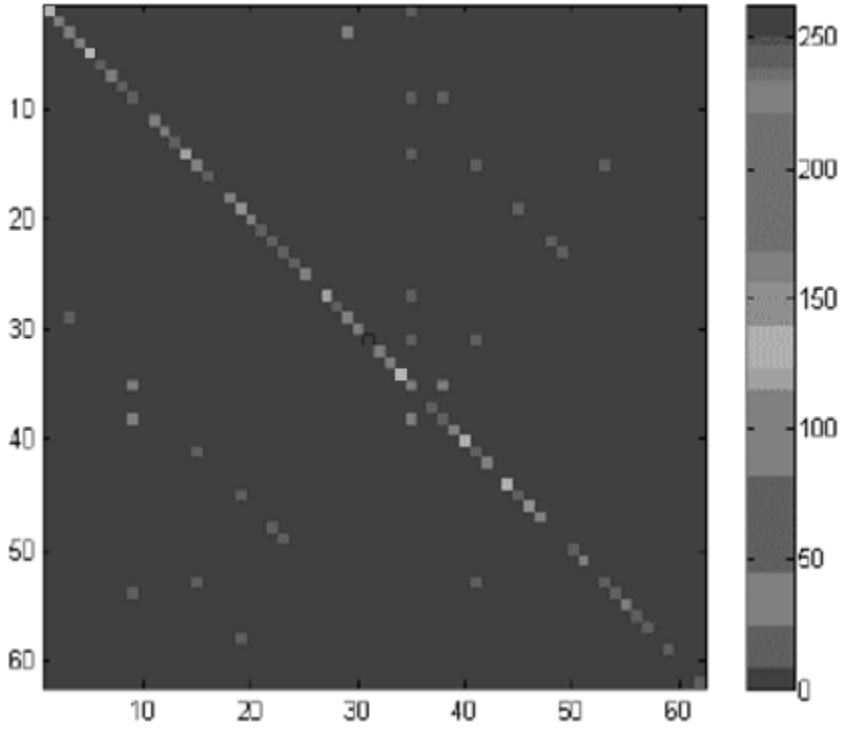


Figure 5.2: Confusion matrix for ICDAR03 test set. Numbers 1-62 show character classes A-Z, a-z,0-9. Lines parallel to the main diagonal show character confusions

In our second experiment, we used the whole ICDAR03 training set to get features for each class of characters. The accuracy on the test set came out to be 70% (see Table 5.2) and one of the reasons for misclassifications is case ambiguity among character classes. In Figure 5.2, the lines parallel to the main diagonal of the confusion matrix reflect these ambiguities, e.g. small case 'c' confused with 'C', etc. In Table 5.2, we also report our results on the training and test splits proposed by (de Campos et al., 2009) for Chars74K, viz., Chars74K-15, where the suffix '15' specifies the number of training and test samples to be used for the experiment.

Table 5.2: Character recognition performance on ICDAR03 and Chars74K-15 datasets.

Method	**ICDAR03**	**Chars74K-15**
GB+NN (de Campos et al., 2009)	41%	47.09%
HoG+NN (Wang and Belongie, 2010)	51.5%	58%
SYNTH+FERNS (Wang et al. 2011)	52%	47%
NATIVE+FERNS (Wang et al. 2011)	64%	54%
Stroke Config.(Yi and Tian, 2014)	62.8%	60%
Our Tensor+NN	69%	57.1%
Proposed Rank-1 Decomp.	70%	59%

Our third experiment was to test the performance of Chars74K-15 test split on a modified training set (using all training samples but those in the test split) as per (de Campos et al., 2009). The results are reported in Table 5.3 below.

Table 5.3: Performance on Chars74K-15 Test Split

Method	Chars74K-15 Test Split
ABBYY FineReader (www.abbyy.com)	31%
GB+NN (de Campos et al., 2009)	54.3%
Our Tensor+NN	68.5%
Proposed Rank-1 Decomp.	69%

The difference in results of Table 5.2 (2nd column) and Table 5.3 clearly show that the number of training samples in the former (15 in this case) is not sufficient to capture the variation in test samples. Hence, we were prompted to do another experiment with leave-random-one-out cross-validation (CV) setting. We show results in Table 5.4 for both Chars74K and ICDAR.

For ICDAR, we combined training and testing segments from ICDAR03 robust character dataset to get one big set for the cross-validation experiment. For Chars74K, as mentioned before, the data already comes without training and testing splits; so we just used it as such. The results in Table 5.4 show median accuracy over 100 trials.

Table 5.4: Recognition performance using leave-random-one-out cross-validation (CV).

Method	ICDAR	Chars74K
Proposed Rank-1 Decomp.	79%	74%

5.5.3 Discussion

5.5.3.1 Accuracy

In Table 5.5 we show recognition performance of our rank-1 tensor decomposition strategy in comparison with our other holistic strategies. The performance of our current approach is better than the previous ones because it combines the powers of both the previous strategies.

Table 5.5: Comparison of performance of our three holistic methods

Method	**ICDAR03**	**Chars74K-15**
Proposed RANK-1	69%	57.1%
Proposed Active Contour	62%	59%
Proposed Rank-1 Decomp.	70%	59%
Method	**ICDAR**	**Chars74K**
Proposed RANK-1 + CV	76%	72.5%
Proposed Active Contour + CV	65%	62%
Proposed Rank-1 Decomp. + CV	79%	74%

5.5.3.2 Pros and Cons

Pros:

1. Smooth frame to frame transitions of rotated copies of a single image gives a better feature descriptor compared with abrupt frame changes of our earlier rank-1 approach.

2. Unlike our rank-1 approach, we don't need to make the image upright before stacking up in tensor.

3. The recognition pipeline is simple and intuitive compared with those using local features.

Cons:

1. Binarization is still a performance bottleneck.

2. Some rotationally symmetric characters, e.g. 'M' and 'W' become ambiguous when rotated through 180 degrees.

5.6 Conclusion

In this chapter, we discussed a novel holistic solution to the problem of natural scene character recognition based on rank-1 tensor decomposition. Compared with Chapter 3, where we stacked all image samples of a class to form an image tensor, we make a tensor per image by stacking together rotated versions of the same image. This way, we were not only able to make frame to frame transitions in the tensor smooth but also captured rotation invariance. For each training/testing image, we formed a 3-mode tensor of rotated instances of the image and then decompose the tensor using rank-1 Tucker decomposition to get our holistic feature. For classification purpose, we use I2CDML framework to be able to better cope with large intra-class variations than the Nearest Neighbor classifier. Through our results we showed the potential of using our approach to better capture shape and font variations that occur in scene text images. We got better results than several baseline methods and achieved improved recognition performance on the datasets using leave-random-one-out cross-validation. In the next chapter, we would

compare all our holistic approaches with a local feature descriptor and in Chapter 7, we would incorporate our character level solution to word images.

CHAPTER 6

HOLISTIC APPROACH IS BETTER THAN LOCAL

In Chapters 3 through 5, we primarily discussed three holistic strategies to recognize characters cropped from natural scene images. Although our results on popular scene character datasets showed better or comparable performance to methods based on local features, in this chapter, we will make an explicit comparison between our holistic strategies and top of the line local feature based approach, namely Histograms of Oriented Gradients (HoG), to support our claim that holistic strategy is better than local and that it should be a part of a reliable scene character recognition framework.

The reason we pick HoG is that it is extensively used by researchers in the scene text recognition community since it was first proposed by (Dalal and Triggs, 2005). It's the underlying feature descriptor for many scene character recognition methods, e.g. (de Campos et al., 2009), (Wang and Belongie, 2010), (Wang et al., 2011), (Yi et al., 2013) etc.

6.1 Datasets for Comparison

We use Chars74K-Font and SYN-10 datasets for the purpose of comparison between holistic and local features. For training, we use Chars74K-Font dataset, which, as previously mentioned in chapter 1, is a synthetic dataset of English alphabet. For testing, we use our synthetic dataset (SYN-10), which utilizes ten (10) common fonts in English in different styles. The images in SYN-10 are generated using imageMagick 6.0 software. We also introduce 10 random rotations

and shear deformations (for each image in a specific font) to mimic some distortions found in natural scene images. Hence, we have 6,200 test synthetic test images in total for 62 classes.

The reason we use synthetic datasets is to be able to better control image quality in order to highlight quality of the feature vectors for the two approaches. Through our experiment and results, we show the comparative advantage of using our holistic strategy over local features like the HoG.

6.2 Training and Testing

For training, we extract both HoG and our holistic tensor and active contour based features for synthetic characters in Chars74K-Font dataset. In the case of HoG, we use the implementation given by Piotr Dollar[###] with the following configuration to extract the features: the image is spatially divided into 8×8 cells over which the spatially weighted histograms of gradients are computed over 9 orientation bins. The block size used is 2×2 and for each histogram, four L2 block normalizations are computed from adjacent histograms (except at the boundary). We collect and concatenate (rasterize) the normalized values of histograms into the final feature descriptor.

In the case of our holistic features, we adopt the same procedures and parameters as described in training sub-sections of Chapters 3, 4 and 5.

[###] https://github.com/pdollar/toolbox

For testing, we extract respective features from the test images in SYN-10 and use the nearest neighbor classifier along with dot product as the similarity metric.

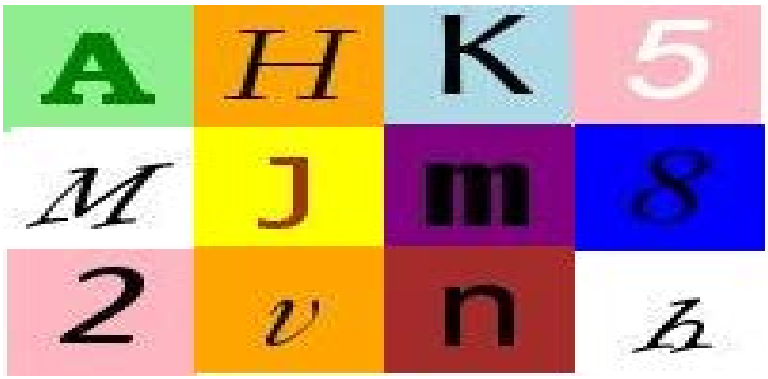


Figure 6.1: Sample images for SYN-10 Test Set

Figure 6.2 shows two images of the digit ‘1’ under different random orientations. For the purpose of quick visual comparison between our holistic features and those from the HoG, we plot the respective graphs of features in Figure 6.2 and 6.3. The figures clearly show that difference in

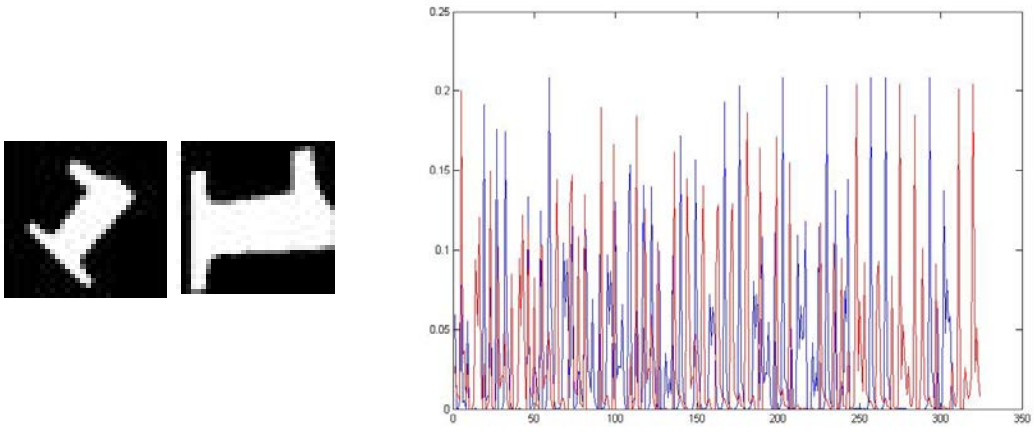


Figure 6.2: Local features (HoG) for the two images from SYN-10 dataset.

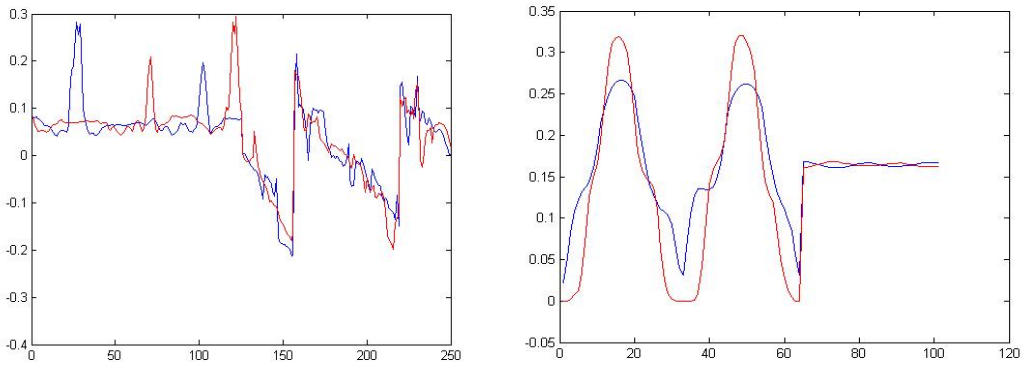


Figure 6.3: Holistic features for the two images in Figure 6.2:
(Left) Active contour based (Right) Rank-1 tensor decomposition based

6.3 Experimental Evaluation

We evaluated our proposed holistic techniques on the test dataset and compared the performance with a representative local feature model, namely histogram of gradients (HoG). The comparison has been carefully done under the same test conditions, vis-à-vis image preprocessing and classification. Datasets of synthetic fonts are chosen for training and testing so as to eliminate the impact of noise. As shown in Table 6.1, where we give random rotations and shears to the test images, our holistic approaches outperform HoG by about 12 to 16%. The results show the mean accuracy score of 10 trials.

It is interesting to note that in the second rows of the Tables 6.1 and 6.2 where we allow HoG to take advantage of color information in the test image, the results still do not show any improvement. In Table 6.2, we restrict test images to an upright position. In this case, HoG performs better than its performance with the rotated versions of the same images. However,

results in both tables clearly show that compared with HoG, our holistic methods are consistent in performance and considerably invariant to rotations.

Table 6.1: Recognition performance of local (HoG) vs proposed holistic features with random test image rotations

Method	**SYN-10**
HoG + Binarization	44.5%
HoG + Color	39.1%
Proposed Rank-1	56%
Proposed Active Contour	58.7%
Proposed Rank-1 Decomp.	60%

Table 6.2: Recognition performance of local (HoG) vs proposed holistic features without test image rotations

Method	**SYN-10**
HoG + Binarization	58.2%
HoG + Color	53.42%
Proposed Rank-1	62%
Proposed Active Contour	59.3%
Proposed Rank-1 Decomp.	61.6%

6.4 Conclusion

In this chapter, we presented a simple but insightful comparison between holistic and local features in the context of scene character recognition. To make a fair comparison between the two

types of features, we carefully controlled the training and test conditions. We used synthetic fonts for training and testing to eliminate the impact of noise. We preprocessed and classified the images in the same manner so as to only highlight the impact of feature quality on the recognition performance. In the end, our results clearly support the narrative that holistic is better than the local and that such a strategy should be a part of a reliable scene character recognition framework. In the next chapter we will apply our holistic models of character recognition to the cropped word recognition problem.

CHAPTER 7

WORD RECOGNITION IN NATURAL SCENE IMAGES

Up until now, we explored different holistic strategies for recognizing characters harvested from natural scene images. We would now like to extend the idea to recognize words cropped from natural scenes. In this chapter, we describe our approach to solve the cropped word recognition problem that utilizes our previously discussed holistic character recognition models. The main idea behind our strategy is to segment a word based on character recognition. In other words, we want to split a given word image into individual characters while using a recognition model to guide the segmentation process.

The notion of word segmentation is not new to the document OCR community where in most cases vertical projection profiles would do the work for a given scanned document image. However, the problem of word segmentation for scene words is very hard due to variable typefaces (regular as well as custom fonts) and noisy imaging conditions.

Considering the application aspect of scene word recognition, the problem has two aspects:

1. Open vocabulary word recognition (or simply word recognition)

2. Lexicon driven word recognition (aka word spotting)

In the case of word spotting, we a given a list of text labels for a cropped word image and the task is to pick the label that matches the most with the input image. While the former problem category is general and forms a part of full-image (aka end-to-end) scene text recognition, the latter

is important in case we already have some prior information available, e.g. a list for grocery shopping.

During the previous four to five years, researchers have put forth ideas to solve both versions of the problem. One popular strategy is to first segment the input word image into individual characters and then recognize individual characters.

(Neumann and Matas, 2011) used vertical projection profiles to get cues for possible character segments in their MSER based framework for word recognition. (Mishra et al., 2011) proposed an MRF model in an iterative graph cut framework to segment the foreground (text) from the background. (Field and Learned-Miller, 2012) proposed bilateral regression to segment and recognize words. Later, (Mishra et al., 2012a) used bottom up technique to segment word images along with top down cues from language models etc. to do the recognition task.

We utilize and build upon the idea of image seams from content aware image resizing proposed in (Avidan and Shamir, 2007) to come up with a novel segmentation technique for scene word images. As part of the segmentation process, we recognize individual characters along the way. The final word recognition is then based on predicting the most probable word using the available spell checking system. Compared to more traditional approaches to segment characters, our method is quite simple and efficient in that it doesn't rely on sliding windows or other computational intensive modelling techniques, e.g. MRF model in (Mishra et al, 2011).

7.1 Image Seam Analysis

Before moving forward with our strategy, it's worthwhile to recap the background of context aware image resizing problem (Avidan and Shamir, 2007). The goal is to find a set of pixels that cross image extents (width or height) and have a low visual significance. Hence, if such a set of pixels is found and removed from the image, there will be very little impact on the overall appearance of the image. Avidan and Shamir term such a set of pixels as an image *seam* and define it as follows:

Let I be an $n \times m$ image, a vertical seam $\boldsymbol{s}^x$ consisting of n pixels is the 8-connected path from top to the bottom of the image, is defined as:

$$\boldsymbol{s}^x = \{s_i^x\}_{i=1}^n = \{(x(i), i)\}_{i=1}^n , s.t. \forall i \; |x(i) - x(i-1) \leq 1| \tag{7.1}$$

where, x is a mapping from $[1, \dots n] \rightarrow [1, \dots m]$.

The pixels on the seam $\boldsymbol{s}^x$ would be:

$$I_s = \{I(\boldsymbol{s}^x)\}_{i=1}^n \tag{7.2}$$

Given an energy function, E, Avidan and Shamir define the optimal seam $\boldsymbol{s}^*$as follows:

$$\boldsymbol{s}^* = \min_s E(\boldsymbol{s}) = \min_s \sum_i^n E(I(s_i)) \tag{7.3}$$

The optimal seam can easily be found by the following dynamic programming formulation:

$$M(i,j) = E(i,j) + \min(M(i-1, j-1), M(i-1, j), M(i-1, j+1)) \tag{7.4}$$

For image energy, we use magnitude of image gradients as a simple energy function:

$$E(I) = \left|\frac{\partial(I)}{\partial x}\right| + \left|\frac{\partial(I)}{\partial y}\right| \tag{7.5}$$

Figure 7.1 shows partial segmentation of a word image from ICDAR03 dataset using the concept of image seams. In the next section, we describe our algorithm for a complete segmentation of a given word image.

Figure 7.1: (Left) Sample word image from ICDAR03 dataset. (Right) Image seam, shown in yellow, guiding the word segmentation

7.2 Word Segmentation & Recognition using Image Seams

Having known what a seam is, we are ready to give our heuristic algorithm to segment and recognize a cropped scene word image from a dataset into individual characters. Ours is a recursive algorithm that tries to split a word image into two partitions based on the location of the image seam. At each recursive step, the algorithm computes the width W of the region from the bounding box of the image. It also determines the maximum width of the connected components w_{max}. We determine whether to binary split the image or not based on whether $W > 1.5\, w_{max}$ as used in (Neumann and Matas, 2011). Whenever W falls within this threshold, i.e. $W \leq 1.5\, w_{max}$ the

algorithm attempts to recognize the content of the image and returns a confidence score. If the score is above a threshold value ρ_r (more on this in Section 7.3.2), the character corresponding to the maximum score is returned. Otherwise, the algorithm attempts to partition the image assuming that the characters contained therein are somehow connected.

Our word recognition framework is illustrated in Figure 7.2 and an example (with actual steps of the segmentation and recognition process) is shown in Figure 7.3.

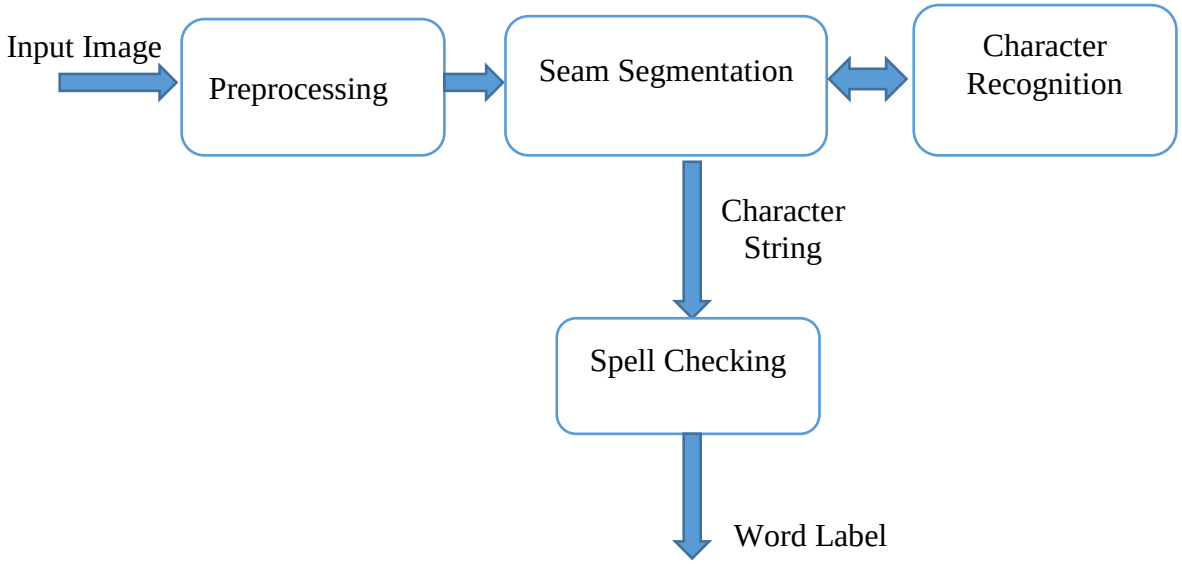


Figure 7.2: Our scene word recognition framework. The preprocessed input image goes through Seam Segmentation module where seam analysis is done with the help of Character Recognition module (based on our holistic model). The output character string is then fed to the Spell Checking module to get the final Word Label

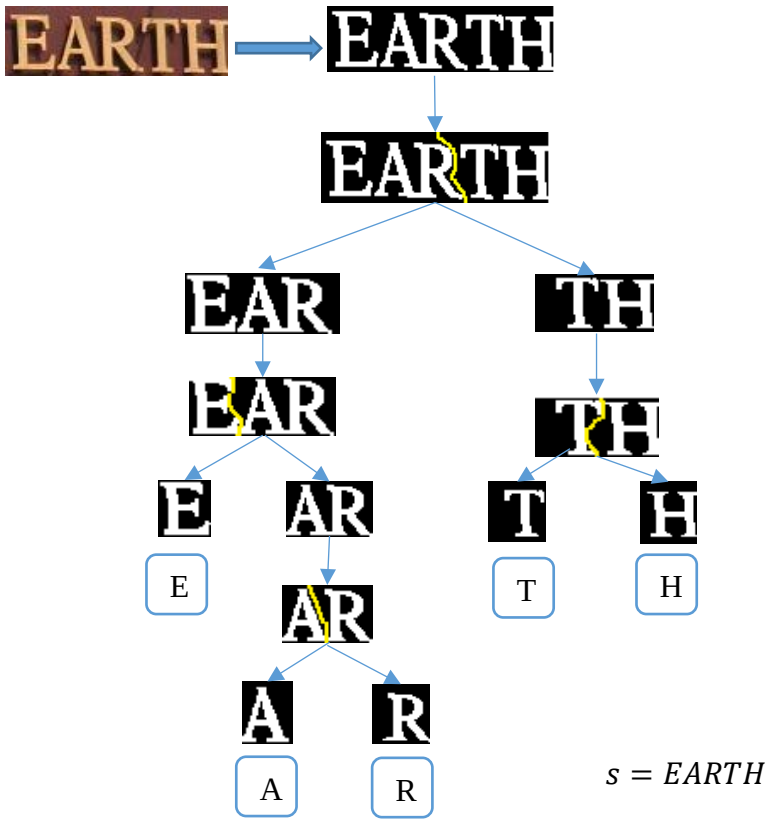


Figure 7.3: Seam segmentation and recognition process for an ICDAR03 dataset word

We begin by preprocessing the word image and converting it into a binary image using our algorithm described in Chapter 4. Then our algorithm for word segmentation works as follows:

1. Compute the width W of the region from the bounding box of the image

2. Compute w_{max} from the connected component analysis of the image region

3. Compare W and w_{max}

   a. If $W > 1.5\ w_{max}$, proceed as follows:

Get the image partitions, im_1& im_2 using image seams and then recursively call the algorithm to get two strings s_1, s_2 and return the concatenation in s

b. If $W \leq 1.5\ w_{max}$

Compute the area of image and if it's less than 5% of the bounding box, ignore it and return an empty string in s. Else proceed as follows:

i. Recognize the image and get the character c and ρ, the recognition confidence value.

ii. If $\rho \geq \rho_r$, return c in s

iii. If $\rho < \rho_r$, proceed as follows:

Get the image partitions, im_1& im_2 using image seams and then recursively call the algorithm to get two strings s_1, s_2 and return the concatenation in s

4. Input the string of characters s to a spell-checker module[§§§] to get a list of predicted words and select the top choice or the one with the minimum edit distance from a given lexicon (if available).

The last step of the algorithm attempts to auto-correct the predicted word. This can be quite useful in cases where some letters may not have been correctly recognized by the character

[§§§] We use the spell checker available with Microsoft Word application

recognition step which could be due to ambiguity in character shape, e.g. 'I' (capital eye) and 'l' (small ell) in typefaces, or due to potentially bad segmentation.

7.3 Computational Complexity

Two main steps in our recursive algorithm that impact its running time are: connected component analysis and seam analysis. At each level of recursion, where the given word is analyzed for splitting into two images, we do connected components processing which takes at most $O(nm)$ computations where n the number of rows is and m is the number of columns. The approximate number of binary seam splits we expect to perform can be bounded from above by $O(log\frac{W}{w_{max}}+1)$, where W is the width of the bounding box and w_{max} is the maximum width of a connected component in the image box.

For each seam split, we use dynamic programming, which in our case can takes $O(nm)$ operations. Hence, the total time complexity turns out to be $O(nm(log\frac{W}{w_{max}}+1))$.

7.4 Experimental Evaluation

We evaluated our word recognition framework for two scenarios: open vocabulary word recognition and word spotting. For both cases, we report results on two popular scene word datasets, namely ICDAR03 and SVT. We used our binary segmentation method discussed in Chapter 4 for preprocessing and for character recognition, we used our Rank-1 feature from Chapter 3, along with all the relevant training parameters. For all experiments, we used the recognition threshold parameter $\rho_r = 0.75$ and $\rho_r = 0.7$ for ICDAR03 and SVT respectively.

The parameter was computed to give best results on respective word spotting datasets for a lexicon of 50 words.

7.4.1 Datasets

We use SVT and ICDAR03-Word datasets for all our experiments. In order to make fair comparisons with previous results on the ICDAR03-Word, we used the experimental criteria laid out in (Wang et al., 2011) that ignores word images containing non-alphanumeric characters or the ones that have two or less characters in it. This brings the number of images from 1,111 to 859.

As mentioned before, the SVT dataset was proposed to introduce the problem of word spotting. Therefore, it contains a small lexicon of about 50 words including the word image to be recognized. ICDAR03-Word dataset is not specific for word spotting. Therefore, (Wang et al., 2011) proposed to use two sub-versions of ICDAR03, namely ICDAR03 (50) and ICDAR03 (FULL), for the problem to be cast as word spotting problem. The suffix '50' refers to 50 random words plus the ground truth for the image. While 'FULL' refers to the fact that the lexicon is made of all the ground truths from the dataset.

7.4.2 Results and Discussion

We report the results for word spotting on ICDAR03 (50), ICDAR03 (FULL), and SVT in Table 7.1. We compare with other baseline methods using word segmentation for recognition. Our recognition accuracy, which is the average of 50 trials, for the word spotting case compares favorably with the other approaches and we get our best result on ICDAR (50) at 80.5%.

Table 7.1: Recognition performance on Word Spotting problem

Method	ICDAR03 (50)	ICDAR03 (FULL)	SVT
(Wang et al., 2011)	73%	62%	57%
(Field and Learned-Miller, 2012)	79.47%	73.43%	54.2%
Proposed Seam Segmentation	80.5%	70%	52.6%

The performance drop for SVT dataset in Table 7.1 is partly due to the fact that images contain a lot of cropping artifacts (especially segments of nearby characters/words).

For the open vocabulary problem, we use ICDAR03-Word test set to compare our approach with the related methods. Following the experiments of (Mishra et al., 2012b), we show results on a subset of this dataset, which, as before, is created by removing all non-alphanumeric characters and all words having two or less characters in them in Table 7.2. As in (Mishra et al, 2012b), the results are presented with case sensitivity ignored. Our results demonstrate comparable performance with other methods.

Table 7.2: Recognition performance on open vocabulary word recognition problem

Method	ICDAR03-Word
(Mishra et al., 2012b)	57.92%
(Field and Learned-Miller, 2013)	62.76%
Proposed Seam Segmentation	60.2%

To analyze the impact of recognition threshold parameter ρ_r on the word recognition accuracy, we performed experiments on ICDAR03 and SVT datasets for word spotting problem. Figure 7.4 shows the variation in performance over a range of values of the parameter. Initially, the accuracy increases by increasing the value of parameter, which is quite intuitive. However, the trend breaks at points, $\rho_r = 0.75$ for ICDAR03 and $\rho_r = 0.7$ for SVT, where the accuracy starts to get down. The reason is that as the recognition threshold increases, the system becomes more and more sensitive and specific towards the learned character model from the character recognition system and begins to reject hypotheses. This results in over-segmentation because now the seam must segment an otherwise potential character candidate. For SVT, the threshold is lower than ICDAR03 due to the fact that the cropped dataset contain a lot of noisy artifacts.

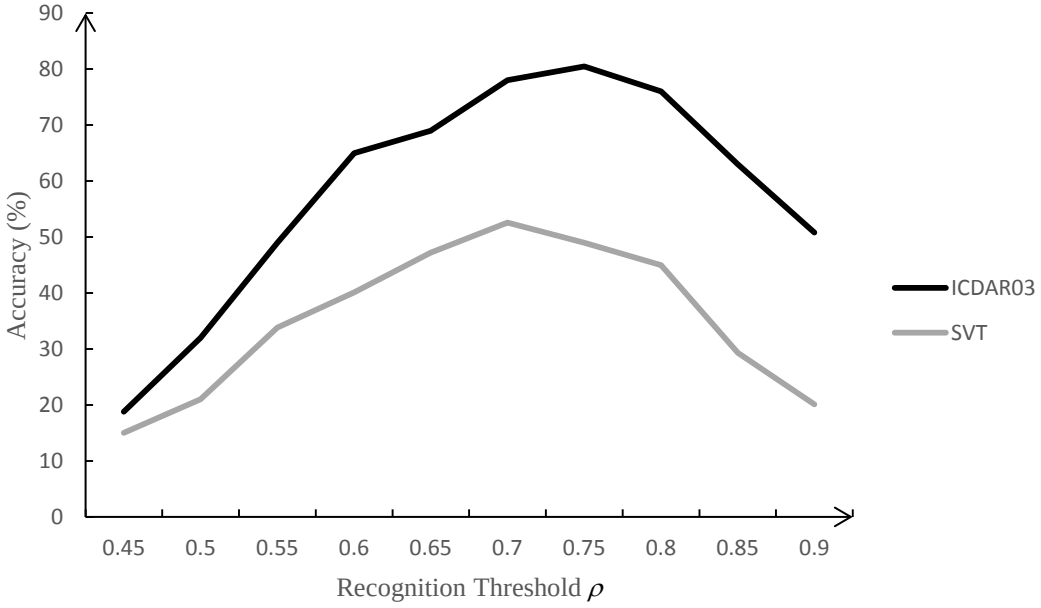


Figure 7.4: Accuracy vs Recognition threshold ρ_r for ICDAR05(50) and SVT datasets

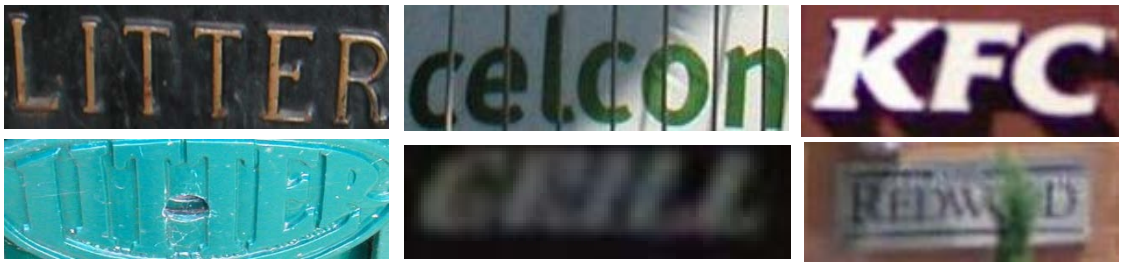


Figure 7.5:Some images from ICDAR03 & SVT that our system could recognize correctly (Top row) and could not recognize correctly (Bottom row)

7.4.3 Pros and Cons

Pros:

1. A simple and intuitive way to segment/recognize scene word image. Compared with competing methods, it doesn't require forming complex structure, MRF (Mishra et al, 2011) or CRF (Field and Learned-Miller, 2013).

2. Unlike vertical projection profiles (Neumann et al, 2011), our system can handle overlapping characters very well.

3. It brings the powers of holistic character recognition to word recognition.

Cons:

1. In the presence of noise, the system sometimes over segments the characters. This, however, can be remedied to some extent by post-processing in the spell-checker module.

2. The post-processing spell-checker we used is based on limited MS Word lexicon, which for proper nouns, e.g. names of businesses on storefronts or related signage, is not well equipped. For this we would need a more powerful post-processor.

7.5 Conclusion

In this chapter we presented a novel technique to segment a word image from challenging datasets like ICDAR03-Word and SVT. Our method used the concept of image seams in conjunction of our holistic character recognition strategy to segment and recognize words in a cropped image. We tackled the misclassifications due to noise and other text occlusions by incorporating spell-checking module in the final output of a word label.

We experimented with both lexicon driven and open vocabulary scenarios of cropped word recognition and our results demonstrate the potential of a promising line of work in this context. We noticed that our results could be improved by better binarization of the input image, which is a problem in its own right. Moreover, we intend to utilize more powerful web search based system to correct the recognized string in place of spell-checking module.

CHAPTER 8

CONCLUSION

8.1 Significance of our work

In this work, we have highlighted the importance of holistic features in the context of recognition of characters extracted from natural scene images and compared the performance of holistic approaches with more prevalent local feature bases methods. STR community has devoted its efforts more on local features and have yet to harness the power of holistic features effectively into STR systems. Our results clearly demonstrate the significance of holistic features and bring home the fact that reliable systems should incorporate holistic local recognition.

To recap, we put forth three strategies to recognize words extracted from natural scene images, aka text in the wild. The first one was based on modelling character images of a class as a 3-mode tensor and then factoring it into a set of rank-1 matrices and the associated mixing coefficients. Each set of rank-1 matrices span the solution subspace of a specific image class thus capturing the required class signature which, along with the mixing coefficients, is used for final character recognition.

The second approach we studied in this work lets us form a novel holistic feature for character recognition based on active contour model, also known as snakes. Our feature vector is based on two variables, direction and distance, cumulatively traversed by each point as the initial circular contour evolves under the force field induced by the character image. The initial contour

design in conjunction with cross-correlation based similarity metric enables us to account for rotational variance in the character image.

Our third approach was based on modelling a 3-mode tensor via rotation of a single image. This method is a different from our first approach above in that we form the tensor based on a single image instead of collecting a specific number of samples of a particular class. Here we do rotation through a predefined range of angles which enables us to better capture rotational variance in the scene character images as demonstrated by our results. This ultimately leads to better performance than the local approaches.

Finally, as an application, we used our character based recognition model to recognize word images extracted from natural scenes. Here we first introduced our novel segmentation method based on image seam analysis to split a word into individual character images. We then applied our holistic models to recognize individual letters and then use a spell-checker to get a plausible candidates for the word in the original image.

Our rigorous and extensive experiments on popular scene datasets like Chars74K-Font, Chars74K-Image, SVT and ICDAR03, and the results demonstrated comparable or better performance than several local features based methods (e.g. HoG, SIFT etc.) justifying our intuition about the importance of holistic strategies in scene text recognition (STR).

8.2 Future Work

STR is a hard problem and still an active area of research and experimentation. There is still a lot of room for researchers to solve different aspects of this problem and to seek solutions

that are comparable with human performance. In the context of our work on scene characters and cropped word recognition, we propose possible future directions/extensions as follows:

1. Better approaches toward segmentation of textual content from background to boost accuracy of recognition

2. Combining holistic with local features in a robust recognition framework

3. Development of scene text detector module that will help make complete scene text reader application

LIST OF REFERENCES

Avidan, S., and Shamir, A. (2007). Seam carving for content-aware image resizing. In ACM Transactions on Graphics. Vol. 26, Issue 3, Article No.10, 2007.

Chen, X., and Yuille, A., (2004). Detecting and reading text in natural scenes. In IEEE Conference on Computer Vision and Pattern Recognition (CVPR).. Vol. 2. pp. II–366, 2004.

Coates, A., Carpenter, B., Case, C., Satheesh, S., Suresh, B., Wang, T., Wu, D., and Ng, A., (2011). Text detection and character recognition in scene images with unsupervised feature learning. In IEEE International Conference on Document Analysis and Recognition (ICDAR). pp. 440–445, 2011.

Dalal, N., and Triggs, B., (2005). Histograms of oriented gradients for human detection. In IEEE International Conference on Computer Vision and Pattern Recognition (CVPR). pp.886-893, 2005.

de Campos, T.E., Babu, B. R., and Varma, M., (2009). Character recognition in natural images. In International Conference on Computer Vision Theory and Applications (VISAPP). pp. 273–280, 2009.

Donoser, M., Bischof, H., and Wagner, S., (2008). Using web search engines to improve text recognition. In IEEE 19th International Conference on Pattern Recognition, ICPR. Vol. No. 14, pp. 8-11, 2008.

Hazan, T., Polak, S., and Shashua, A., (2005). Sparse image coding using a 3D non-negative tensor factorization. In IEEE International Conference on Computer Vision (ICCV). Vol. 1, pp. 50–57, 2005.

Field, J., and Learned-Miller, E. (2012). Scene text recognition with bilateral regression. Department of Computer Science, University of Massachusetts, Amherst, Tech. Rep. UM-CS-2012-021, 2012.

Field, J., and Learned-Miller, E. (2013). Improving open-vocabulary scene text recognition. In IEEE International Conference on Document Analysis and Recognition (ICDAR). pp. 604–608, 2013.

Kita, K., and Wakahara, T. (2010). Binarization of color characters in scene images using k-means clustering and support vector machines. In IEEE International Conference on Pattern Recognition (ICPR). pp. 3183–3186, 2010.

Kolda, T. G.and Bader, B. W., (2009). Tensor decompositions and applications. SIAM review, 51(3): pp 455–500, 2009.

Ivins, J., and Porrill J. (2000). Everything you always wanted to know about snakes. AIVRU Technical Memo 86, July 1993 (Revised June 1995; March 2000)

Kass, M., Witkin, A., and Terzopoulos, D. (1987). Snakes: Active contour models. In International Journal of Computer Vision (IJCV). Vol. 1, n. 4, pp. 321-331, 1987.

Lucas, S.M., Panaretos, A., Sosa, L., Tang, A., Wong, S., and Young, R. (2003). ICDAR 2003 robust reading competitions. In Proceedings of the IEEE Seventh International Conference on Document Analysis and Recognition (ICDAR). Vol. 2, pp. 682–687, 2003.

Mishra, A., Alahari, K., and Jawahar, C. (2012a). Top-down and bottom-up cues for scene text recognition. In IEEE International Computer Vision and Pattern Recognition (CVPR). pp. 2687–2694, 2012.

Mishra, A., Alahari, K., and Jawahar, C. (2011). An MRF model for binarization of natural scene text. In IEEE International Conference on Document Analysis and Recognition (ICDAR). pp. 11–16, 2011.

Mishra, A., Alahari, K., and Jawahar, C. (2012b). Scene text recognition using higher order language priors. In British Machine Vision Conference, 2012.

Nagy, R., Dicker, A., and Meyer-Wegener, K. (2011). NEOCR: A configurable dataset for natural image text recognition. In CBDAR Workshop (ICDAR). pp. 53-58, 2011.

Neumann, L., and Matas, J. (2011). A method for text localization and recognition in real-world images. In Asian Conference on Computer Vision (ACCV). pp. 770–783, 2011.

Niblack, W. (1985). An introduction to digital image processing. Strandberg Publishing Company.

Otsu, N. (1979). A threshold selection method from gray-level histogram. In IEEE Transactions on System, Man and Cybernetics. Vol.9, pp.62-69, 1979.

Shashua, A., and Levin, A. (2001). Linear image coding for regression and classification using the tensor-rank principle. In IEEE International Conference on Computer Vision and Pattern Recognition (CVPR). Vol. 1, pp. 42-49, 2001.

Smith, D., Field, J., Learned-Miller, E. (2011). Enforcing similarity constraints with integer programming for better scene text recognition. In IEEE International Conference on Computer Vision and Pattern Recognition (CVPR). pp 345-353, 2011.

Sun, C., Junejo, I. N., and Foroosh, H. (2011). Action recognition using rank-1 approximation of joint self-similarity volume. In IEEE International Conference on Computer Vision (ICCV). pp. 1007-1012, 2011.

Tariq, A. and H. Foroosh, H. (2015). Feature-independent context estimation for automatic image annotation. In IEEE International Conference on Computer Vision and Pattern Recognition (CVPR). pp. 1958-1965, 2015.

Wang, T., Wu, D., Coates, A., and Ng, A. (2012). End-to-end text recognition with convolutional neural networks. In IEEE International Conference on Pattern Recognition (ICPR). pp. 330-338, 2012.

Wang, K., Babenko, B., and Belongie, S. (2011). End-to-end scene text recognition. In IEEE International Conference Computer Vision (ICCV). pp. 1457–1464, 2011.

Wang, K., and Belongie, S. (2010). Word spotting in the wild. In European Conference on Computer Vision (ECCV). pp. 591–604, 2010.

Wang, Z., Hu, Y., and Chia, L. (2010). Image-to-Class distance metric learning for image classification. In European Conference on Computer Vision. (ECCV). Part I, LNCS 6311, pp. 706-719, 2010.

Weinman, J., Learned-Miller, E., and Hanson, A. (2009). Scene text recognition using similarity and a lexicon with sparse belief propagation. In IEEE Transactions on Pattern Analysis and Machine Intelligence TPAMI. Vol. 31, No. 10, pp. 1733–1746, 2009.

Xianqian, L., and Sidiropoulos, N.D. (2001). Cramer-Rao lower bounds for low-rank decomposition of multidimensional arrays. In IEEE Transactions on Signal Processing. Vol. 49, Issue 9, pp. 2074-2086, 2001.

Xu, C., and Prince, J.L. (1997). Gradient vector flow: a new external force for snakes. In Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Los Alamitos: Computer Society Press, pp. 66-71, 1997.

Xu, C., and Prince, J.L. (1998). Snakes, shapes, and gradient vector flow. In IEEE Transactions on Image Processing. Vol. 7, Issue 3, pp 359-369, 1998.

Yi, C., and Tian, Y. (2014). Scene text recognition in mobile applications by character descriptor and structure configuration. In IEEE Transactions on Image Processing, pp 2972-2982, 2014

Yi, C., Yang, X., and Tian, Y. (2013). Feature representations for scene text character recognition: a comparative study. In IEEE International Conference on Document Analysis and Recognition (ICDAR). pp 2564-2569, 2013.

Yokobayashi M., and Wakahara, T. (2005). Segmentation and recognition of characters in scene images using selective binarization in color space and GAT correlation. In IEEE International Conference on Document Analysis and Recognition (ICDAR). pp. 167–171, 2005.