Exploiting similarity hierarchies for multi-script scene text understanding

A dissertation submitted by **Lluís Gómez-Bigordà** at Universitat Autònoma de Barcelona, Dept. Ciències de la Computació, to fulfil the degree of **Doctor of Philosophy** in Computer Science.

Bellaterra, February 19, 2016

Director: **Dr. Dimosthenis Karatzas**
Universitat Autònoma de Barcelona
Dept. Ciències de la Computació & Centre de Visió per Computador

This document was typeset by the author using LaTeX 2ε.

The research described in this book was carried out at the Computer Vision Center, Universitat Autònoma de Barcelona.

Abstract

This thesis addresses the problem of automatic scene text understanding in unconstrained conditions. In particular, we tackle the tasks of multi-language and arbitrary-oriented text detection, tracking, and recognition in natural scene images and videos. For this we have developed a set of generic methods that build on top of the basic assumption that text has always some visual key characteristics that are independent of the language or script in which it is written.

Scene text extraction methodologies are usually based in classification of individual regions or patches, using a priori knowledge for a given script or language. Human perception of text, on the other hand, is based on perceptual organisation through which text emerges as a perceptually significant group of atomic objects. In this thesis, we argue that the text extraction problem could be posed as the detection of meaningful groups of regions.

We address the problem of text segmentation in natural scenes from a hierarchical perspective, making explicit use of text structure, aiming directly to the detection of region groupings corresponding to text within a hierarchy produced by an agglomerative similarity clustering process over individual regions. We propose an optimal way to construct such an hierarchy introducing a feature space designed to produce text group hypotheses with high recall and a novel stopping rule combining a discriminative classifier and a probabilistic measure of group meaningfulness based in perceptual organization.

We propose a new Object Proposals algorithm that is specifically designed for text and compare it with other generic methods in the state of the art. At the same time we study to what extent the existing generic Object Proposals methods may be useful for scene text understanding.

Then, we present a hybrid algorithm for detection and tracking of scene text where the notion of region grouppings plays also central role. A scene text extraction module based on Maximally Stable Extremal Regions (MSER) is used to detect text asynchronously, while in parallel detected text objects are tracked by MSER propagation. The cooperation of these two modules goes beyond the full-detection approaches in terms of time performance optimization, and yields real-time video processing at high frame rates even on low-resource devices.

Finally, we focus on the problem of script identification in scene text images in order to build a multi-language end-to-end reading system. Facing this problem with state of the art CNN classifiers is not straightforward, as they fail to address a key characteristic of scene text instances: their extremely variable aspect ratio. Instead of resizing input images to a fixed size as in the typical use of holistic CNN classifiers, we propose a patch-based classification framework in order to preserve discriminative parts of the image that are characteristic of its class. We describe a novel method based on the use of ensembles of conjoined networks to jointly learn discriminative stroke-parts representations and their relative importance in a patch-based classification scheme. Our experiments with this learning procedure demonstrate the viability of script identification in natural scene images, paving the road towards true multi-lingual end-to-end scene text understanding.

Acknowledgments

Contents

Chapter 1

Introduction

Reading is the complex cognitive process of transforming forms (letters and words) into meanings [94] in order to understand and interpret written content. Since ancient times[1], written language has been the most important form of nonverbal communication, information storage, and sharing of ideas for humankind. Reading is therefore considered a basic skill for humans which allows them access to vast amount of knowledge available through text.

In the field of human-machine interfaces, the idea to equip computers with reading capabilities is as old as computer science itself. It arose as a natural way to acquire data from non-digital written content and to automate tasks related with text processing.

With its roots in the first Optical Character Recognition (OCR) systems developed around the 60's, Document Image Analysis is nowadays a mature research field in the crossroads between Pattern Recognition and Computer Vision. It is considered one of the most fruitful areas of Computer Vision, and off-the-shelf OCR solutions available today are extremely efficient in documents created with modern printers and standard font types. However, research in document analysis has still many open challenges that involve reading tasks in non-optimal conditions, e.g. handwritten text, historical and/or deteriorated documents, complex layouts, etc.

Although the majority of humans' reading activity is carried out over text written on paper or electronic display devices, texts may also appear written on objects around us: text on street signs, a motto painted on a wall, and even text produced by arranging small stones on the sand. This type of text, from now on referred as "scene text", is an important source of information in our daily life.

Scene text is ubiquitous in man-made environments and most of our everyday activities imply reading and understanding written information in the world around us (shopping, finding places, viewing advertisements, etc).

The automated recognition of scene text in uncontrolled environments is an open Computer Vision problem. At its core lies the extensive variability of scene text in terms of its location, physical appearance, and design (Figure 1.2).

[1] The use of ideographic symbols (proto-writing) is traced to 6000 BCE. Surviving evidences of today's in-use scripts, as the one shown in Figure 1.1, date from three centuries BCE.

Figure 1.1: Cippus Perusinus, Etruscan writing near Perugia, Italy. The beginning of the writing with the Latin alphabet, 3rd or 2nd century BCE.

Figure 1.2: The extensive variability of scene text in terms of its location, physical appearance, and design makes automated recognition of scene text a challenging problem.

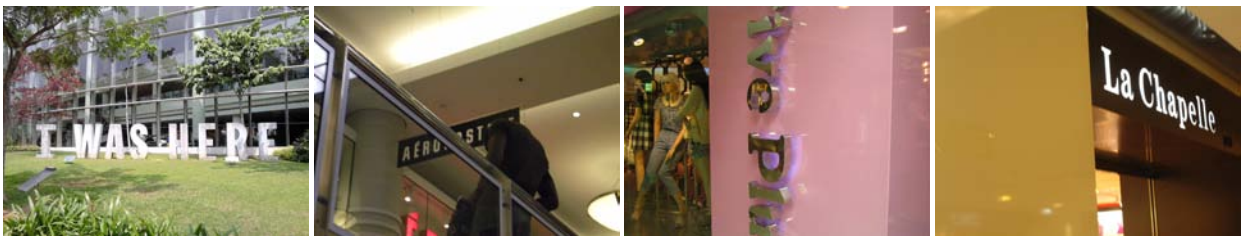
1.1 *Scene text understanding tasks*

Scene text reading systems typically follow a two-step pipeline design: first applying an algorithm to detect (localize) text regions in the input image, and then performing the actual text recognition into the previously localized regions. These two well differentiated tasks (localization and recognition) have been traditionally treated as isolated problems by the researchers, each one having its own evaluation framework. A frequent intermediate task is the pixel-level segmentation of text components, also known as text extraction, that, when attainable, would allow the use of document-oriented OCR techniques in the recognition step.

When a method is designed to do both localization and recognition tasks at once we call it an "end-to-end" reading system. We refer to the task of end-to-end word spotting when the recognition is constrained to a relatively small list of given words (queries).

The automated understanding of textual information in natural scene images has received increasing attention from computer vision researchers over the last decade. Text localization, extraction, and recognition methods have evolved significantly and their accuracy has increased drastically in recent years [78, 112, 60, 57]. However, the problem is far from being considered solved. The best performing methods in the last ICDAR Robust Reading Competition achieve only 69%, 74%, and 70% recall in the tasks of text localization, segmentation, and end-to-end recognition respectively. Moreover, although standard benchmark datasets have traditionally concentrated on English text that is well-focused and horizontally-aligned, new datasets have recently appeared covering much more unconstrained scenarios. These include incidental text [57], i.e. text appearing in the scene without the user intention, multi-script and arbitrary oriented text [143, 68].

If one considers multi-language environments, script and language identification become also crucial tasks for reliable scene text understanding. Since text recognition algorithms are language-dependent, detecting the script and language at hand allows selecting the correct language model to employ [130].



(a) (b) (c) (d)

Figure 1.3: Complexity of natural scene images: (a) cluttered background, (b) occlusions, (c) shadows and highlights, (d) perspective distortions.

1.2 *Challenges*

Most of the difficulty in scene text understanding comes from the fact that we have virtually no any restriction on how scene text can appear within an image. Thus, the first big challenge we face, is to deal in a robust way with high variability in font type, size, color, orientation, and alignment of text. See for example Figure 1.2. Other notable challenges stem from the intrinsic complexity of natural scene images (Figure 1.3), and from certain disadvantages inherent to the capturing technology of camera based systems (Figure 1.4).

There exist many aspects of scene text that makes its recognition a totally different problem to the one that OCR systems solve for scanned documents. Complex backgrounds and occlusions are difficult challenges to address in a completely uncontrolled environment. Low-contrast and uneven lighting, that can appear in the form of reflections and shadows, can make things even worse. The projective nature of camera based imaging makes the planar text in natural scenes to appear with perspective distortion, other kind of distortion arises when text appears in non-planar and non-rigid surfaces or when using wide-angle lenses.

Regarding the inherent challenges of digital camera imaging, in general, images with low resolution make difficult any task of text segmentation and at the same time are not well posed input for the final OCR step in the system. The same applies for blur and uneven focus that can appear in the image in situations where we are shooting moving objects, the camera is not static, or the light is low, if the sensor is not fast enough to maintain an optimal shutter speed. Color quantization performed in low-profile cameras makes a high

Figure 1.4: Examples of inherent challenges of digital camera imaging: (a) uneven focus, (b) motion blurring, (c) image compession artifacts.

(a) (b) (c)

coloring difference when comparing a scanner obtained image and a digital camera one, sensor noise tends to also be a common problem in consumer-grade imaging devices. Another important challenge of digital camera imaging is that often one must work with compressed images without access to the raw data.

Finally, if one considers portable devices as the final target for automated text reading systems, another important challenge to have in mind is the necessity for fast algorithms, as limited computing resources are often available on the device.

1.3 *Applications and socio-economic impact*

We live in the days of mobile and wearable computing revolution. Billions[2] of pervasive, personal, and portable devices, equipped with integrated built-in digital cameras, flood our streets and have become indispensable in our daily life. At the same time, our strongly networked society produces such amount of digital media data per day that was unimaginable just one decade ago. Providing those, already ubiquitous, imaging devices with automatic image understanding capabilities is a primary goal for the Computer Vision community. A field that is certainly being revolutionized these days.

[2] Global camera phone sales reached 1 Billion in 2011. http://www.strategyanalytics.com/default.aspx?mod=reportabstractviewer&a0=6216

Possible applications of scene text understanding are countless. Being able to automatically read scene text in any conditions would enable new and exciting applications such as automatic translation, way-finding and navigation aid, support tools for elderly and visually impaired persons, or contextual image indexing and search among many others.

Assisted navigation and translation tools are an important kind of applications with recurrent appearance in the literature [48, 148, 46, 83, 109, 7]. On one side elderly and visually impaired persons and on the other foreigner visitors can take advantage of these technologies to access textual information or text present in urban environments that would otherwise be unreadable for them.

Nowadays there exist reliable translation applications that are already in the hands of millions of users[3] [7]. However, such real world applications are limited to very well delivered conditions, e.g. horizontally oriented and well focused bi-level text, and often require the user to specify the location and the original language of the text to be translated. Figure 1.5 show common errors found in existing image-based text translation applications. Also, in some cases they

[3] WordLens, Google Translate, and Microsoft Translator services are examples of a real applications of end-to-end scene text detection and recognition that have acquired market-level maturity.

Figure 1.5: Existing image-based text translation applications, (a) Google Translate, (b) Word Lens, (c) Microsoft Translator, are still limited to well delivered conditions. Common errors of such applications include incorrect recognition for out-of-dictionary words, missing detections in non-bilevel or blurred text, and false detections in textures. Images from Neumann and Matas [101].

rely on large-scale data center infrastructure. Thus, there is still room for research on more robust and efficient methods.

Similarly, while there exist pre-market prototypes of what would be the future of assisted reading technology for visually impaired persons[4], reliable applications are still designed to perform specific tasks, e.g. as traditional OCR engines for camera based devices[5], rather than providing an anytime/anywhere real solution.

[4] http://www.orcam.com/

[5] http://www.knfbreader.com/

On a different sort of things, robust scene text reading systems can provide an important source of information for automatic annotation and indexing of images and videos.

Digital media archives (e.g. from broadcast channels, or film industry agents) have been densely annotating their databases traditionally by hand in order to make them searchable. Automatic algorithms to help or replace humans on such an effort are highly desirable and will certainly have a direct impact in industry. Nowadays, text-based image retrieval [91, 51, 38] is already a viable solution for searching in large scale collections of unlabeled data.

Other possible applications in this line include the use of textual information to improve lifelog photo stream analysis, automatic captioning algorithms, and on-line image search engines. The list can be increased with countless ad-hoc applications addressing specific purposes, like for example in forensic investigations of digital media[6].

[6] TRAIT-2016 is an open challenge for text detection and recognition algorithms to support forensic investigations. The presence of text may allow a location to be identified or to generate leads.http://www.nist.gov/itl/iad/ig/trait-2016.cfm

Other possible applications of scene text understanding for diverse automated systems include reading license plates, address numbers, gas meters, cargo container and warehouse codes, or athlete bibs in images and videos. In a similar manner, text data could also provide useful information to mobile robots and intelligent vehicle systems.

1.4 *Objectives and research hypotheses*

The main goal of this thesis is to contribute to the state of the art of automatic scene text understanding in unconstrained conditions. We put the focus in the most challenging scenario, when there is not any prior information about the kind of text that may be present in a given input image. In particular, we tackle the tasks of multi-language and arbitrary-oriented text detection, tracking, and recognition in natural scene images and videos.

For this we have developed a set of generic methods that build on top of the basic assumption that text has always some visual key characteristics that are independent of the language/script in which they are written. As can be appreciated in Figure 1.6 text instances in any language are always formed as groups of similar atomic parts, being them either individual characters, small stroke parts, or even words. This observation is central in the development of the work drawn in this thesis.

The primary research hypothesis of this thesis is that reliable scene text detection and tracking methods must take advantage from this

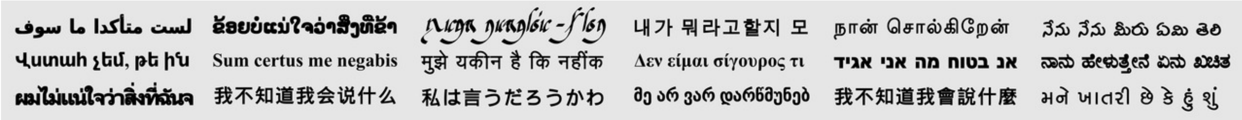


Figure 1.6: The observation that all text, independently of its language/script, has common visual key characteristics is central in the development of this thesis.

holistic (sum-of-parts) based perspective. Thus, we pose the problem of scene text localization as the detection of groups of regions, as opposed to the traditional classification of individual regions or image-patches as text or non-text entities by himself. Something that we consider not feasible because individual text components (e.g. a single character or stroke) lack any semantic information, and many times their shapes resemble other objects or object parts that are frequently found on scene images.

Throughout this thesis we make extensive use of Hierarchical Clustering algorithms in order to generate a hierarchy of groupings proposals that emerge bottom-up from an initial set of image regions that are agglomerated by their similarity. However, as we will see in the following chapters, the definition of such similarity measure is not trivial for scene text components. For example, while in general using color similarity may seem adequate, in some cases a given scene text instance may be composed by components with large color disparity, either by design or because it suffers from illumination highlights or shadows. Indeed, in natural scenes a relevant grouping may appear as the result of combining several grouping cues together. In this context, we will name "similarity heterarchy" to the ensemble of hierarchies generated by grouping the initial regions in a number of different ways. Such similarity heterarchies can be efficiently exploited for multi-script scene text understanding tasks.

Since the algorithms build on top of our primary hypothesis are designed with multi-language environments in mind, we further expand the research scope of this thesis to the task of script identification in the wild. For this, however, we turn our main intuition the other way around: while our claim is that for the detection and tracking tasks we will benefit from taking a holistic approach, in the case of script identification we put the focus back on the parts.

Examining text samples from different scripts in Figure 1.6, it is clear that some stroke-parts are quite discriminative when it comes to classify text instances by their script, whereas others can be trivially ignored as they occur in multiple scripts. Our research hypothesis here is that once we have identified a group of text components as a text instance, being it a single word or a text line, the ability to distinguish these relevant stroke-parts can be leveraged for recognizing the corresponding script.

1.5 Contributions

The following is the list of contributions that we have made throughout this Thesis:

Contributed data

In *Chapter 3* the ICDAR Robust Reading datasets are detailed. The author of this Thesis has contributed in the design of the evaluation protocol and in coordinating the labeling efforts of the *Reading Text in Video* challenge. In the same Chapter we introduce the Multi-Lingual end-to-end (MLe2e) dataset. MLe2e is the first multi-lingual dataset for the evaluation of scene text end-to-end reading systems and all intermediate stages. It is thus an important contribution of this Thesis that has been made public for the research community.

The use of Perceptual Organization for text detection

In *Chapter 4* we present a novel text extraction method based on perceptual organization laws. For this we pose that the fact that a region can be related to other neighboring ones (by similarity and/or proximity laws) is central to classifying the region as a text part. We claim that this is in-line with the essence of human perception of text, which is largely based on perceptual organization. We make use of a state of the art perceptual organization computational model to assess the meaningfulness of different candidate groups of regions. These groups emerge naturally through the activation of different visual similarity laws in collaboration with a proximity law.

Efficient analysis of similarity hierarchies for text detection

Chapter 5 presents a text detection method that efficiently exploit the inherent hierarchical structure of text. The main contributions of this chapter are the following: First, we learn an optimal feature space that encodes the similarities between text components thus allowing the Single Linkage Clustering algorithm to generate text group hypotheses with high recall, independently of their orientations and scripts. Second, we couple the hierarchical clustering algorithm with novel discriminative and probabilistic stopping rules, that allow the efficient detection of text groups in a single grouping step. Third, we propose a new set of features for text group classification, that can be efficiently calculated in an incremental way, able to capture the inherent structural properties of text related to the arrangement of text parts and the intra-group similarity.

Text specific Object Proposals

In *Chapter 6* we explore the applicability of Object Proposals techniques in scene text understanding, aiming to produce a set of word proposals with high recall in an efficient way. We propose a simple text specific selective search strategy, where initial regions in the image are grouped by agglomerative clustering in a hierarchy where each node defines a possible word hypothesis. We evaluate different state of the art Object Proposals methods in their ability of detecting text words in natural scenes. We compare the proposals obtained with well known class-independent methods with our own method,

demonstrating that our proposal is superior in its ability of producing good quality word proposals in an efficient way.

Group-based text tracking

Chapter 7 introduces our method for tracking of text in video sequences. The novelty of the proposed method lies in the ability to effectively track text regions' groupings (establishing a pixel level segmentation of constituent text parts in every frame), not merely their bounding boxes as usually done in state-of-the-art tracking-by-detection algorithms. The contribution of this chapter is mainly on the methodological aspect, presenting a simple but effective method for text detection and tracking suitable for devices with limited computational power. We demonstrate how a reasonably fast text detection method can be efficiently combined with tracking to give rise to real-time performance on such devices.

Part-based script identification in the wild

Finally, in *Chapter 8* we describe two different patch-based methods for script identification of scene text instances. We develop the idea that applying holistic CNNs to the problem of script identification is not straightforward, because the aspect ratio of scene text instances is largely variable (i.e. from a single character to a whole text line). The key intuition behind the presented methods is that in order to retain the discriminative power of small stroke patches we must rely in powerful local feature representations and use them within a patch-based classifier. We demonstrate that patch-based classifiers can be essential in tasks where scaling the input image to a fixed size is not feasible.

1.6 *Publications*

Articles published in conferences and workshops:

- Gómez, Lluís, and Dimosthenis Karatzas. "Multi-script text extraction from natural scenes." Document Analysis and Recognition (ICDAR), 2013 12th International Conference on. IEEE, 2013.

- Gómez, Lluís, and Dimosthenis Karatzas. "Demonstration of a Human Perception Inspired System for Text Extraction from Natural Scenes." CBDAR-Demos.

- Gómez, Lluís, and Dimosthenis Karatzas. "MSER-Based Real-Time Text Detection and Tracking." Pattern Recognition (ICPR), 2014 22nd International Conference on. IEEE, 2014.

- Gómez, Lluís, and Dimosthenis Karatzas. "Scene Text Recognition: No Country for Old Men?." Computer Vision-ACCV 2014 Workshops. Springer International Publishing, 2014.

- Gómez, Lluís, and Dimosthenis Karatzas. "Object Proposals for Text Extraction in the Wild." Document Analysis and Recognition (ICDAR), 2015 13th International Conference on. IEEE, 2015.
- Gómez, Lluís, and Dimosthenis Karatzas. "A Fine Grained Classification Approach to Scene Text Script Identification." DAS (2016).

Journal articles under review:

- Gómez, Lluis, and Dimosthenis Karatzas. "A fast hierarchical method for multi-script and arbitrary oriented scene text extraction." Submitted to IJDAR (2015).
- Gómez, Lluís, Anguelos Nicolaou, and Dimosthenis Karatzas. "Boosting patch-based scene text script identification with ensembles of conjoined networks." Submitted to Pattern Recognition (2016).

Non first authorship but related with this thesis' work:

- Karatzas, D., Shafait, F., Uchida, S., Iwamura, M., Gómez i Bigordà, L., Robles Mestre, S., ... , de las Heras, L. P. (2013, August). ICDAR 2013 robust reading competition. In Document Analysis and Recognition (ICDAR), 2013 12th International Conference on (pp. 1484-1493). IEEE.
- Karatzas, Dimosthenis, Sergi Robles, and Lluis Gómez. "An online platform for ground truthing and performance evaluation of text extraction systems." Document Analysis Systems (DAS), 2014 11th IAPR International Workshop on. IEEE, 2014.
- Jon Almazan, Suman K. Ghosh, Lluís, Gómez, Dimosthenis Karatzas and Ernest Valveny. "A Selective Search Framework for Efficient End-to-end Scene Text Recognition and Retrieval." Technical Report (2015).
- Dimosthenis Karatzas, Lluis Gómez-Bigordà, Anguelos Nicolaou, Suman Ghosh, Andrew Bagdanov, Masakazu Iwamura, Jiri Matas, Lukas Neumann, Vijay Ramaseshan Chandrasekhar, Shijian Lu, Faisal Shafait, Seiichi Uchida, Ernest Valveny. ICDAR 2015 robust reading competition. In Document Analysis and Recognition (ICDAR), 2015 13th International Conference on. IEEE.

Chapter 2

Related Work

In this Chapter we review the state of the art for all tasks related to the work presented in this thesis. At the same time we put in context our work by explicitly comparing with existing approaches.

First, in Section 2.1 we focus on methods for scene text detection in still images, while Section 2.3 presents the most relevant works for text detection and tracking in video sequences. In Section 2.2, we review generic object proposals methods that releted to our work, and compare them with our approach for text specific regions proposals. Section 2.4 is dedicated to end-to-end scene text recognition methods, where we also review works tackling the specific task of word spotting. Finally, in Section 2.4 a we review recent approaches for script identification in the wild.

2.1 *Scene text localization and extraction*

The automatic understanding of textual information in natural scenes has gained increasing attention over the past decade, giving rise to various new computer vision challenges. Ye and Doermann [145] offer an exhaustive survey of recent developments in text detection and recognition in imagery, while Jung *et al.* [56] and Liang *et al.* [75] have published corresponding surveys of early works on camera-based document and scene text analysis.

Scene text detection methods can be categorized into texture-based and region-based approaches. Texture-based methods usually work by performing a sliding window search over the image and extracting certain texture features in order to classify each possible patch as text or non-text. Coates *et al.* [17], and in a different flavour Wang *et al.* [137] and Netzer *et al.* [95], propose the use of unsupervised feature learning to generate the features for text versus non-text classification. Wang *et al.* [135], extending their previous work [136], have built an end-to-end scene text recognition system based on a sliding window character classifier using Random Ferns, with features originating from a HOG descriptor. Mishra *et al.* [90] propose a closely related end-to-end method based on HOG features and a SVM clas-

sifier. Texture based methods yield good text localisation results, although they do not directly address the issue of text segmentation (separation of text from background). Their main drawback compared to region based methods is their lower time performance, as sliding window approaches are confronted with a huge search space in such an unconstrained (i.e. variable scale, rotation, aspect-ratio) task. Moreover, these methods are usually limited to the detection of a single language and orientation for which they have been trained on, therefore they are not directly applicable to the multi-script and arbitrary oriented text scenario.

Region-based methods, on the other hand, are based on a typical bottom-up pipeline: first performing an image segmentation and subsequently classifying the resulting regions into text or non-text ones. Yao *et al.* [143] extract regions in the Stroke Width Transform (SWT) domain, proposed earlier for text detection by Epshtein *et al.* [27]. Yin *et al.* [146] obtain state-of-the-art performance with a method that prunes the tree of Maximally Stable Extremal Regions (MSER) using the strategy of minimizing regularized variations. The effectiveness of MSER for character candidates detection is also exploited by Chen *et al.* [13] and Novikova *et al.* [105], while Neumann *et al.* [98] propose a region representation derived from MSER where character/non-character classification is done for each possible Extremal Region (ER).

Most of the region-based methods are complemented with a post-processing step where regions assessed to be characters are grouped together into words or text lines. The hierarchical structure of text has been traditionally exploited in a post-processing stage with heuristic rules [27, 13] usually constrained to search for horizontally aligned text in order to avoid a combinatorial explosion of enumerating all possible text lines. Neumann and Matas [98] introduce an efficient exhaustive search algorithm using heuristic verification functions at different grouping levels (i.e. region pairs, triplets, etc.), but still constrained to horizontal text. Yao *et al.* [143] make use of a greedy agglomerative clustering where regions are grouped if their average alignment is under a certain threshold. Yin *et al.* [146] use a self-training distance metric learning algorithm that can learn distance weights and clustering thresholds simultaneously and automatically for text groups detection in a similarity feature space.

There exist two main differences between current state-of-the-art approaches and the methods proposed in this Thesis. On one side, methods relying on learning processes [135, 137, 17, 95, 136, 90] are usually constrained to detect the single script which they have been trained on. The feedback loop between localization and recognition they propose, although performing well on certain tasks (e.g. detecting English horizontal text), contradicts with the human ability to detect text structures even in scripts or languages never seen before. In comparison, the methodology proposed here requires no training, and is largely parameter free and independent to the text script.

On the other hand, there is an important distinction between the

way the grouping is used by existing methods [144, 27, 13] and the way perceptual organisation is used in Chapter 4. In past work, grouping is used solely as a post-processing step, once the text parts have already been identified as such, the main reason being to address the results / ground truth semantic level mismatch mentioned before. As a matter of fact, we also use a post-processing step to enable us to evaluate on standard datasets. However, crucially, in our approach perceptual organisation provides the means to perform classification of regions, based on whether there exists an interpretation of the image that involves their participation to a perceptually relevant group.

In Chapter 5 we present a novel hierarchical approach in which region hierarchies are built efficiently using Single Linkage Clustering in a weighted similarity feature space. The hierarchies are built in different color channels in order to diversify the number of hypotheses and thus increase the maximum theoretical recall. Our method is less heuristic in nature and faster than the greedy algorithm of Yao *et al.* [143], because the number of atomic objects in our clustering analysis is not increased by taking into account all possible region pairs; besides our method uses similarity and not collinearity for grouping. Yin *et al.* [146], make use of a two step architecture first doing an automatic clustering analysis in a similarity feature space and then classifying the groups obtained in the first step. The method presented here differs from such approaches in that our agglomerative clustering algorithm integrates a group classifier, acting as a stopping rule, that evaluates the conditional probability for each group in the hierarchy to correspond to a text group in an efficient manner through the use of incrementally computable descriptors. In this sense our work is related with Matas and Zimmerman [81] region detection algorithm, while the incremental descriptors proposed here are designed to find relevant groups of regions in a similarity dendrogram instead of the detection of individual regions in the component tree of the image. There is also a relationship between our method with the work of Van de Sande *et al.* [131] and Uijlings *et al.* [129] on using segmentation and grouping as selective search for object recognition. However, our approach is distinct in that their region grouping algorithm agglomerates regions in a class-independent way while our hierarchical clusterings are designed in order to maximize the chances of finding specifically text groups. Thus, our algorithm can be seen as a task-specific selective search. This idea motivates our work in object proposals for text detection and will be discussed further in chapter 6.

2.2 *Object proposals*

Over and obove the specific problem of scene text detection the use of Object Proposals methods to generate candidate class-independent object locations has become a popular trend in computer vision in recent times. A comprehensive survey can be found in Hosang *et*

al. [49]. In general terms, we can distinguish between two major types of Object proposals methods: the ones that make use of exhaustive search to evaluate a fast to compute objectness measure [1, 15, 151], and the ones where the search is segmentation-driven [128, 79, 65].

In the first category, Alexe *et al.* [1] propose a generic objectness measure for a given image window that combines several image cues, such as a saliency score , the color contrast to its immediate surrounding area, the edge density, and the number of straddling contours. Computation of these features is made efficient by using integral images. Cheng *et al.* [15] propose a very fast objectness score using the norm of image gradients in a sliding window, with a suitable resizing of windows into a small fixed size. A different sliding window driven approach is given by Zitnick *et al.* [151], where a box objectness score is measured as the number of edges [22] that are wholly contained in the box minus those that are members of contours that overlap the box's boundary. Using efficient data structures they manage to evaluate millions of candidate boxes in a fraction of second.

On the other hand, selective search methods make use of image's inherent structure through segmentation to guide the search. In this spirit, Gu *et al.*[47] make use of a hierarchical segmentation engine [4] and consider each node in the hierarchy as an object part hypothesis. Uijlings *et al.*[128] argue that a single segmentation and grouping strategy is not enough to generate high quality object locations in any conditions, and thus propose a selective search algorithm that uses multiple complementary strategies. In particular, they start from superpixels using different parameter settings [31] for a variety of color spaces, and then produce a set of hierarchies by merging adjacent regions using different complementary similarity measures. Another method based on superpixels merging is due to Manen *et al.*[79], using the connectivity graph induced by the segmentation [31] of an image, with edge weights representing the likelihood that two neighboring pixels belong to the same object, their Randomized Prim's algorithm generate proposals by sampling random partial spanning trees with large expected sum of weights. Finally, Krähenbühl *et al.* [65] compute an oversegmentation of the image using a fast edge detector [22] and the Geodesic K-means algorithm [108]. Then they identify a small set of seed superpixels, aiming to hit all objects in the image, and object proposals are identified as critical level sets of the Geodesic Distance Transforms (SGDT) computed for several foreground and background masks for these seeds.

The use of Object Proposals techniques in scene text understanding has been exploited very recently in two state-of-the-art word-spotting methods [51]. Jaderberg *et al.* [51] propose the use of a generic Object Proposals algorithm [151] and deep convolutional neural networks for whole word recognition.

In the method proposed in Chapter 6 initial regions in the image are grouped by agglomerative clustering, using complementary sim-

ilarity measures, in hierarchies where each node defines a possible word hypothesis.

2.3 *Scene text detection and tracking in video sequences*

While most of the recently published state-of-the-art text detection methods focus on still images, many authors have also addressed the task of text detection and tracking in digital video sequences. A comprehensive survey dedicated exclusively to video processing is presented by Zhang and Kasturi [150].

Relevant work on video based text detection and tracking can be categorized into batch-processing and online-processing methods. Within the former category, Li *et al.* [73] proposed a text tracking scheme where a SSD (Sum of Squared Differences) based correlation module was used to track the detected text between adjacent frames. Crandall *et al.* [19] proposed a method for caption text extraction where the motion vectors encoded in MPEG-compressed videos were used for tracking. Gllavata *et al.* [40] take advantage of temporal redundancy for detecting challenging text displayed against a complex background, by building a multi-frame clustering. Myers and Burns [93] propose a method to track planar regions of scene text with arbitrary 3-D rigid motion by correlating small patches and computing homographies on multi-frame blocks simultaneously. All the aforementioned text tracking methods were designed for off-line video processing, and thus are not directly applicable to a real-time system.

Regarding online-processing, Kim *et al.* [64] proposed a method that analyses the textural properties of text in images using a sliding window based classifier, and then locates and tracks the text regions by applying the continuously adaptive mean shift algorithm (CAMSHIFT) [9] on the texture classification results. Merino and Mirmehdi develop in [82] a real-time probabilistic tracking framework based on particle filtering, where SIFT matching is used to identify text regions from one frame to the next. Their region based text detection method is further extended in [83], where the authors present a head-mounted device for text recognition in natural scenes. A similar wearable camera system for the blind is presented in the work of Goto and Tanaka [46] where text strings are extracted using a DCT-based method [45] and then grouped into temporal chains by a text tracking method also based on particle filter. Minetto *et al.* [87] propose a method combining a region-based text detection algorithm [86] with a particle filter tracking module. Fragoso *et al.* [35] have developed an Augmented Reality (AR) see-through text translator where the initial text detection is done with help of the user, which has to tap on the screen near the center of the word of interest, then the plane in which text stands is tracked in real-time using efficient second-order minimization (ESM). Petter *et al.* [109] have extended this application with automatic text detection (not requiring the user interaction) based on Connected Components analysis on

the Canny edge detector output.

From the methods described, only [82] [46] and [109] report comparable frame rates to the method detailed in Chapter 7 (between 7 fps. and 10 fps.), but in [82] and [46] processing times are calculated in a standard PC and not in low-resource devices. The key difference is that [82] and [46] make use of a full-detection strategy, performing a full text detection on every frame, in order to provide observations to the tracking system, while in our method the detection module is only performed periodically in a subset of the incoming frames. On the other hand, although [73] and [87] use a similar strategy to combine and merge text detections and tracked text regions, in our method the detection algorithm is executed asynchronously in a separate thread, thus not affecting to the overall speed of the system. An analogue multi-threaded tracking solution is also used by Takeda *et al.* [126] in a document retrieval AR system to display relevant information as an image superimposed in real-time over the camera captured frames.

Another important difference between the work presented here and the ones in [87, 73, 64, 35] is that our system is able to obtain an updated segmentation of tracked characters at every frame, despite not doing a full detection. Since we are not tracking just bounding box information but the text-regions themselves, we can take advantage of several segmentations of each character that would eventually lead to improved recognition accuracy.

Finally, while most of the described methods are constrained to detect horizontally aligned text [82] [46] [87] [109], and pure translational movement models [87] [73] [19], our method can deal with multi-oriented text and is able to track it under scale, rotation, and perspective distortions.

2.4 *End-to-end methods*

Table 2.1 summarize several aspects of language models used in existing end-to-end approaches for unconstrained recognition, and a more detailed description of such methods is provided afterwards. Table 2.1 compares the use of character/word n-grams (as well as their sizes), smoothing, over-segmentation, and dictionaries (here the "Soft" keyword means the method allow Out-Of-Dictionary word recognition, while "Hard" means only "In-Dictionary" words are recognized).

Method	Dictionary	Char n-gram	Word n-gram	Smoothing	Over-segmentation
Neumann *et al.* [96, 97, 98, 99]	Soft (10k)	bi-gram	No	No	Yes
Wang *et al.* [137]	Hard (hunspell)	No	No	No	Yes
Neumann *et al.* [100]	No	3-gram	No	No	Yes
Yao *et al.* [142]	Soft (100k)	bi-gram	No	No	No
Bissacco *et al.* [7]	Soft (100k)	8-gram	4-gram	Yes	Yes

Table 2.1: Summary of different language model aspects of end-to-end scene text recognition methods.

In [96] Neumann and Matas propose the classification of Maximally Stable Extremal Regions (MSERs) as characters and non characters, and then the grouping of character candidates into text lines with multiple (mutually exclusive) hypotheses. For the recognition each MSER region mask is normalized and resized to a fixed size in order to extract a feature vector based on pixel directions along the chain-code of its perimeter. Character recognition is done with a SVM classifier trained with synthetic examples. Ambiguous recognition of upper-case and lower-case variants of certain letters (e.g. "C" and "c") are tackled as a single class, and then differentiated using a typographic model that also serves to split the line into words. Finally, a combination of bi-gram and dictionary-based language model scores each text line hypothesis individually and the most probable hypothesis is selected. The authors further extended the text detection part of their method in several ways [97, 98], increasing their end-to-end recognition results. In [99] they add a new inference layer to the recognition framework, where the best sequence selection is posed as an optimal path problem, solved by a standard dynamic programming algorithm, allowing the efficient processing of even more segmentation hypotheses.

Wang *et al.* [137] propose the use of Convolutional Neural Networks together with unsupervised feature learning to train a text detector and a character recognizer. The responses of the detector in a multi-scale sliding window, with Non-Maximal Suppression (NMS), give rise to text lines hypotheses. Word segmentation and recognition is then performed jointly for each text line using beam search. For every possible word the character recognizer is applied with an horizontal sliding window, giving a score matrix that (after NMS) can be used to compute an alignment score for all words in a small given lexicon. Words with low recognition score are pruned as being "non-text". Since the method is only able to recognize words in a small lexicon provided, in order to perform a more unconstrained recognition the authors make use of an off-the-shelf spell checking software to generate the lexicon given the raw recognition sequences.

In a very related work to the one presented in this paper Milyaev *et al.*[84] demonstrate that off-the-shelf OCR engines can still perform well on the scene text recognition task as long as appropriate image binarization is applied to input images. For this, they evaluate 12 existing binarization methods and propose a new one using graph cuts. Their binarization method is combined with an AdaBoost classifier trained with simple features for character/non-character classification. And the components accepted by the classifier are used to generate a graph by connecting pairs of regions that fulfill a set of heuristic rules on their distance and color similarity. Text lines obtained in such way are then split into words and passed to a commercial OCR engine[1] for recognition.

In [100] Neumann and Matas propose the detection of constant width strokes by convolving the gradient image with a set of bar filters at different orientations and scales. Assuming that charac-

[1] OCR Omnipage Professional, available at http://www.nuance.com/

ters consist in a limited number of such strokes a set of candidate bounding-boxes is created by the union of bounding boxes of 1 to 5 nearest strokes. Characters are detected and recognized by matching the stroke patterns with an Approximate Nearest Neighbor classifier trained with synthetic data. Each candidate bounding box is labeled with a set of character labels or rejected by the classifier. The non rejected regions are then agglomerated into text lines and the word recognition is posed as an optimal sequence search by maximizing an objective function that combines the probabilities of the character classifier, the probability of the inter-character spacing difference of each triplet, the probability of regions' relative positioning, and the characters adjacency probability given by a 3-gram language model.

Yao *et al.* [142] propose an arbitrary oriented scene text detection and recognition method that extracts connected components in the Stroke Width Transform (SWT) domain[27]. Component analysis filters out non-text components using a Random Forest classifier trained with novel rotation invariant features. This component level classifier performs both text detection and recognition. Remaining character candidates are then grouped into text lines using the algorithm proposed by Yin *et al.* [146], and text lines are split into words using the method in [27]. Finally, the authors propose a modified dictionary search method, based on the Levenshtein edit distance but relaxing the cost of the edit operation between very similar classes, to correct errors in individual character recognition using a large-lexicon dictionary[2]. To cope with out-of-dictionary words and numbers, n-gram based correction [122] is used if the distance with closest dictionary word is under a certain threshold.

[2] Word list is provided by the Microsoft Web N-Gram Service (http://webngram.research.microsoft.com/info/) with top 100k frequently searched words on the Bing search engine.

Bissacco *et al.* [7] propose a large-scale end-to-end method using the conventional multistage approach to text extraction. In order to achieve a high recall text detection the authors propose to combine the outputs of three different detection methods: a boosted cascade of Haar wavelets [133], a graph cuts based method similar to [107], and a novel approach based on anisotropic Gaussian filters. After splitting text regions into text lines a combination of two over-segmentation methods is applied, providing a set of possible segmentation points for each text line. Then beam search is used to maximize a score function among all possible segmentations in a given text line. The score function is the average per-character log-likelihood of the text line under the character classifier and the language model. The character classifier is a deep neural network trained on HOG features over a training set consisting of around 8 million examples. The output layer of the network is a softmax over 99 character classes and a noise (non-text) class. At test time this classifier evaluates all segmentation combinations selected by the beam search. The language model used in the score function to be optimized by the beam search is a compact character-level ngram model (8-gram). Once the beam search has found a solution the second level language model, a much larger distributed word-level ngram (4-gram), is used to rerank.

As a conclusion of the state of the art review we can see that the majority of reviewed methods make use of similar language models, based on a dictionary of frequent words and a character n-gram (usually a bi-gram). A much stronger language model has been proposed by Bissacco *et al.* [7]. On the other hand, while the system in [7] makes use of three different detection methods combined with an independent recognition module, in other cases both detection and recognition are treated together, e.g. using the same features for detection and recognition [142] or using multiple character detections as an over-segmentation cue for the recognition [97], giving rise to more compact and efficient methods.

2.5 *Script Identification*

Script identification is a well studied problem in document image analysis. Gosh *et al.* [37] has published a compehensive review of methods dealing with this problem. They identify two broad categories of methods: structure-based and visual appearance-based techniques. In the first category, Spitz and Ozaki [124, 123] propose the use of the vertical distribution of upward concavities in connected components and their optical density for page-wise script identification. Lee *et al.* [70], and Waked *et al.* [134] among others build on top of Spitz seminal work by incorporating additional connected component based features. Similarly, Chaudhuri *et al.* [12] use the projection profile, statistical and topological features, and stroke features for classification of text lines in printed documents. Hochberg *et al.* [55] propose the use of cluster-based templates to identify unique characteristic shapes. A method that is similar in spirit with the one presented in Chapter 8, while requiring textual symbols to be precisely segmented to generate the templates.

Regarding segmentation-free methods based on visual appearance of scripts, i.e. not directly analyzing the character patterns in the document, Wood *et al.* [139] experimented with the use of vertical and horizontal projection profiles of full-page document images. More recent methods in this category have used texture features from Gabor filters analysis [127, 106] or Local Binary Patterns [32].

All the methods discussed above are designed specifically with printed document images in mind. Structure-based methods require text connected components to be precisely segmented from the image, while visual appearance-based techniques are known to work better in bilevel text. Moreover, some of these methods require large blocks of text in order to obtain sufficient information and thus are not well suited for scene text which typically comprises a few words.

Contrary to the case of printed document images, research in script identification on non traditional paper layouts is scarce, and has been mainly dedicated to video overlaid-text until very recently. Gllavatta *et al.* [39], in the first work dealing with this task, proposed a method using the wavelet transform to detect edges in overlaid-text images. Then, they extract a set of low-level edge features, and make

use of a K-NN classifier.

Sharma *et al.* [113] have explored the use of traditional document analysis techniques for video overlaid-text script identification at word level. They analyze three sets of features: Zernike moments, Gabor filters, and a set of hand-crafted gradient features previously used for handwritten character recognition. They propose a number of pre-processing algorithms to overcome the inherent challenges of video overlaid-text. In their experiments the combination of super resolution, gradient features, and a SVM classifier perform significantly better that the other combinations.

Phan *et al.* [110] propose a method for combined detection of video text overlay and script identification. They propose the extraction of upper and lower extreme points for each connected component of Canny edges of text lines and analyse their smoothness and cursiveness.

Shivakumara *et al.* [118, 119] rely on skeletonization of the dominant gradients They analyze the angular curvatures [118] of skeleton components, and the spatial/structural [119] distribution of their end, joint, and intersection points to extract a set of hand-crafted features. For classification they build a set of feature templates from train data, and use the Nearest Neighbor rule for classifying scripts at word [118] or text block [119] level.

As said before, all these methods have been designed (and evaluated) specifically for video overlaid-text, which presents in general different challenges than scene text. Concretely, they mainly rely in accurate edge detection of text components and this is not always feasible in scene text.

More recently, Sharma *et al.* [114] explored the use of Bag-of-Visual Words based techniques for word-wise script identification in video-overlaid text. They use Bag-Of-Features (BoF) and Spatial Pyramid Matching (SPM) with patch based SIFT descriptors and found that the SPM pipeline outperforms traditional script identification techniques involving gradient based features (e.g. HoG) and texture based features (e.g. LBP).

Singh *et al.* [120] propose a closely related approach where densely computed (SIFT) local features are pooled to encode mid-level representations of the scripts.

In 2015, the ICDAR Competition on Video Script Identification (CVSI-2015) [115] challenged the document analysis community with a new competitive benchmark dataset. With images extracted from different video sources (news, sports etc.) covering mainly overlaid-text, but also a few instances of scene text. The top performing methods in the competition where all based in Convolutional Neural Networks, showing a clear difference in overall accuracies over pipelines using hand-crafted features (e.g. LBP and/or HoG).

The first dataset for script identification in real scene text images was provided by Shi *et al.*in [117], where the authors propose the Multi-stage Spatially-sensitive Pooling Network (MSPN) method. The MSPN network overcomes the limitation of having a fixed size

input in traditional Convolutional Neural Networks by pooling along each row of the intermediate layers' outputs by taking the maximum (or average) value in each row. Their method is extended in [116] by combining deep features and mid-level representations into a globally trainable deep model. They extract local deep features at every layer of the MSPN and describe images with a codebook-based encoding method that can be used to fine-tune the CNN weights.

Nicolaou *et al.* [103] has presented a method based on texture features producing state of the art results in script identification for both scene or overlaid text images. They rely in hand-crafted texture features, a variant of LBP, and a deep Multi Layer Perceptron to learn a metric space in which they perform K-NN classification.

In Chapter 8 we propose a patch-based method for script identification useing convolutional features, extracted from small image patches, and the Naive-Bayes Nearest Neighbour classifier (NBNN). We also present a simple weighting strategy in order to discover the most discriminative parts per class in a fine-grained classification approach.

We take inspiration from recent methods in fine-grained recognition. In particular, Krause *et al.* [66] focus on learning expressive appearance descriptors and localizing discriminative parts. By analyzing images of objects with the same pose they automatically discover which are the most important parts for class discrimination. Yao *et al.* [141] obtain image representations by running template matching using a large number of randomly generated image templates. Then they use a bagging-based algorithm to build a classifier by aggregating a set of discriminative yet largely uncorrelated classifiers.

Our method resembles [141, 66] in trying to discover the most discriminative parts (or templates) per class. However, in our case we do not assume those discriminative parts to be constrained in space, because the relative arrangement of individual patches in text samples of the same script is largely variable.

Also in Chapter 8 we extend this patch-based methodology it two ways: On one side, we make use of a much deeper Convolutional Neural Network model. On the other hand, we replace the weighted NBNN classifier by a patch-based classification rule that can be integrated in the CNN training process by using an Ensemble of Conjoined Networks. This way, our CNN model is able to learn at the same time expressive representations for image patches and their relative contribution to the patch-based classification rule.

From all reviewed methods the ones proposed in this thesis, along with [114] and [120], are the obnly ones based in a patch-based classification framework. While [114] and [120] make use traditional image recognition techniques, our method tries to combine the streghness of CNN representations and patch-based frameworks. Our intuition is that in cases where holistic CNN models are not directly applicable, as in the case of text images (because of their highly variable aspect ratios), the contribution of rich parts descriptors without any deterioration (either by image distortion or by descriptor quantiza-

tion) is essential for correct image classification. In the experimental section we demonstrate that our approach improves the state-of-the-art in the SIW-13 [116] dataset for scene text script classification by a large margin of 5 percentage points, while performs competitively in the CVSI-2015 [115] video overlaid-text dataset.

Chapter 3
Datasets

Standardized datasets and evaluation frameworks offer us with the possibility to measure our research' progress in a quantitatively way, putting our work in context with existing methods. Public scene text datasets constitute also an invaluable source of training data that we have used extensively in our work.

In this Chapter we offer a description of all datasets that are used throughout this thesis. We start in Section 3.1 with details of the different datasets that are part of ICDAR Robust Reading Competitions series. These include the earliest scene text dataset from 2003, its respective revisions, and the newly added video, and incidental text datasets. Then, we give details of several multi-language datasets for scene text detection and script identification in Sections 3.2 and 3.3 respectively. In Section 3.4 we introduce the Multi Language end-to-end dataset (MLe2e) that has been build by us. Finally, Section 3.5 shortly describes the on-line platform for ground truthing and performance evaluation of text extraction systems that was developed for the organization of the ICDAR Robust Reading competition and has been made public.

3.1 The ICDAR Robust Reading Competition

The series of Robust Reading Competitions addresses the need to quantify and track progress in the domain of text extraction from versatile text containers like born-digital images, real scenes, and videos. The competition was started in 2003 by Simon Lucas initially focusing only in scene text detection and recognition [78]. In 2011, the competition introduced a new challenge on the text extraction from born-digital images [112][58]. The 2013 edition of the Robust Reading Competition [60], further integrated the two challenges on real scenes and born-digital images, unifying their performance evaluation metrics, ground truth specification, and the list of offered tasks. Also in 2013 a new challenge was established on text extraction and tracking from video sequences, introducing a new dataset, tools and evaluation frameworks. In its latest edition (2015) a new dataset on Incidental Scene Text has been made public, and the task of end-to-

end recognition has been introduced for all existing datasets.

Focused Scene Text dataset

In its original version, created for the ICDAR 2003 Robust Reading competition [78], the Scene Text dataset contains 509 images, splitted in a training set of 258 and a test set of 251, with resolutions varying from 640×480 to 1280×960. During the different editions of the competition, in 2005, 2011, and 2013 the dataset was revamped to remove duplicated images, and to improve the quality of the annotations. In its last version the ICDAR2013 dataset [60] contains 462 images, from which 229 comprise the training set and 233 images the test set.

While covering only English and horizontally aligned text, this dataset represents the reference benchmark up to date for the evaluation of scene text understanding related tasks: localization, segmentation, and recognition.

Initially the Focused Scene Text dataset was designed to evaluate three different tasks: scene text localization, words recognition, and character recognition. For the task of text localization the dataset has ground-truth defined at the word level with axis oriented bounding boxes and the proposed evaluation framework is detailed by [78] for the 2003 version and by Wolf and Jolion [138] for the all the following editions since 2005. For the tasks of word and character recognition, the cropped (pre-segmented) words/characters images are provided and it is required to provide a unique word/character transcription for each one.

For the task of text segmentation, introduced in 2013, the dataset has ground-truth defined at the pixel level in the form of binary masks. The evaluation framework for this task is defined in [16] by Clavelli and Karatzas.

Finally, the end-to-end task, introduced in 2015, evaluate the ability of scene text reading systems to jointly detect and recognize text in the dataset full images. The evaluation protocol proposed by Wang [135] is used which considers a detection as a correct if it overlaps a ground truth bounding box by more than 50% and their transcriptions match, ignoring the case. At test time end-to-end methods can make use of a list of provided lexicon to improve their recognition. This way methods are evaluated in different tables depending on the size of the lexicon used: strongly contextualized recognition,

Figure 3.1: Sample images from the ICDAR 2003/2013 dataset.

weakly contextualized recognition, or generic recognition.

Reading Text in Video

The ICDAR 2013 Robust Reading competition established a new challenge on text extraction from video sequences, introducing a new dataset, tools and evaluation framework. The Reading Text in Video dataset responds to the necessity of adapting to a new format. Text understanding in video sequences differs from still images in many aspects and it is not a straightforward assumption that a method devised and trained on static text images would be applicable to video sequences.

A key characteristic of those video sequences is the temporal redundancy of text, which calls for tracking based processes taking advantage of past history to increase the stability and quality of detections. Keeping constant track of a text object throughout all the frames where it is visible is desirable for example to ensure a unique response of the system (e.g. translation, or text to speech conversion) for each distinct text, and also to be able to enhance the text regions, or to select the best frames in which they appear, before doing the full text recognition. Moreover, one can take advantage of the tracking process in order to obtain a real-time system, under the assumption that the scene does not change much from frame to frame.

The released dataset provides a total of 49 annotated video sequences, 25 videos in the training set and 24 in the test set, with duration between 10 seconds and 1 minute long. The video sequences represent real-life applications in both indoor and outdoor scenarios, and were recorded using different devices including mobile phones, hand-held cameras and head-mounted cameras. Figure 3.2 show some frames extracted from the dataset videos.

The competition task requires that words are localized and recognized correctly in every frame and tracked over the video sequence, and thus an evaluation procedure has been designed in those terms. From the numerous evaluation frameworks proposed for multiple object tracking systems, we selected to use the CLEAR-MOT [6] and VACE [61] metrics adapted to the specificities of text detection, recognition, and tracking, extending the CLEAR-MOT code provided by Bagdanov et al. [5].

Figure 3.2: Sample images from the ICDAR 2015 Scene Text video dataset.

Figure 3.3: Sample images from the ICDAR 2015 Incidental Text dataset.

Incidental Scene Text

The ICDAR 2015 competition introduced a new challenge on Incidental Scene Text detection and recognition, based on a dataset of 2000 images (splited into equal size sets for training and test). All images were acquired with wearable devices in a variety of indoor/outdoor scenarios and have a resolution of 1280×720 pixels. Sample images of this dataset are shown in Figure 3.3.

Incidental scene text refers to text that appears in the scene without the user having taken any specific prior action to cause its appearance or improve its positioning or quality in the frame. While focused scene text (see Section 3.1) is the expected input for applications such as translation on demand, incidental scene text covers another wide range of applications linked to wearable cameras or massive urban captures where the capture is difficult or undesirable to control.

Similarly as in the case of Focused Scene Text, the dataset id designed to evaluate different tasks: localization, word recognition, and end-to-end, but not pixel-level segmentation. The ground truth is defined again at the word level, but in this case using arbitrary oriented bounding boxes (described by their four points coordinates). The evaluation framework for the localization and end-to-end tasks is based on a single Intersection-over-Union criterion, with a threshold of 50%, similarly to standard practice in object recognition and Pascal VOC challenge [29].

3.2 *Multi-language scene text detection*

KAIST

The KAIST dataset [71] comprises 3000 natural scene images, with a resolution of 640x480, categorized according to the language of the scene text captured: Korean, English, and Mixed (Korean + English). Images were obtained with digital camera and mobile phone from a variety of scenarios (indoor and outdoor) and the type of text found is similar as in the ICDAR Focused Scene Text. Sample images are shown in Figure 3.4. This dataset does not provide a fixed split for training and test, in our experiments we use only 800 images for testing corresponding to the Mixed subset in accordance to other reported results.

The dataset provides ground-truth annotations for the tasks of text localization and segmentation. For localization the grountruth

Figure 3.4: Sample images from the KAIST dataset.

is defined at the levels of word and individual characters, and the evaluation framework is the same as for the ICDAR Focused Scene Text dataset.

For the segmentation task the ground-truth is defined at the pixel level in the form of binary masks. The evaluation metrics precision p and recall r are defined as $p = |E \cap T|/|E|$ and $r = |E \cap T|/|T|$, where E is the set of pixels estimated as text and T is the set of pixels corresponding to text components in the ground truth.

MSRA-TD500

The MSRA-TD500 dataset [144] contains arbitrary oriented text in both English and Chinese and is proposed as an alternative to the ICDAR2003 [78] dataset where only horizontal English text appears. The dataset contains 500 images in total, with varying resolutions from 1296×864 to 1920×1280. Images are taken from indoor (office and mall) and outdoor (street) scenes using a packet camera and depict typically well focused text. Sample images are shown in Figure 3.5. Ground truth annotations are defined at the text-line level with oriented rectangles defined by their axis oriented version and the rotation angle.

The evaluation is done as proposed in [144] using minimum area rectangles. For an estimated minimum area rectangle D to be considered a true positive, it is required to find a ground truth rectangle G such that:

$$A(D' \cap G')/A(D' \cup G') > 0.5, abs(\alpha_D - \alpha_G) < \pi/8$$

where D' and G' are the axis oriented versions of D and G, $A(D' \cap G')$ and $A(D' \cup G')$ are respectively the area of their intersection and union, and α_D and α_G their rotation angles. The definitions

Figure 3.5: Sample images from the MSRA-TD500 dataset.

Figure 3.6: Sample images from the MSRRC dataset.

of precision p and recall r are: $p = |TP|/|E|$, $r = |TP|/|T|$ where TP is the set of true positive detections while E and T are the sets of estimated rectangles and ground truth rectangles.

ICDAR2013 Multi-Script Robust Reading Competition

A competition to detect mixed text in Kannada and/or English from scene images was organized by the MILE[1] Lab in 2013. The motivation was to look for script-independent algorithms that detect text and extract it from scene images, which may be applied directly to an unknown script.

[1] http://mile.ee.iisc.ernet.in/mrrc/

The Multi-Script Robust Reading Competition (MSRRC) dataset [68] comprises 334 camera-captured scene images, 167 in the training and 167 in the test set respectively, with sizes around 1.2MP, annotated for text localization and segmentation tasks, as well as for cropped word recognition. It covers Latin, Chinese, Kannada, and Devanagari scripts, and includes text with multiple orientations. Sample images are shown in Figure 3.6.

For the localization task the ground-truth is defined at the word level with axis oriented bounding boxes and the proposed evaluation framework is by Wolf and Jolion [138]. For the segmentation task the ground-truth is defined at the pixel level in the form of binary masks and the evaluation metrics are the same as explained in Section 3.2 for the KAIST dataset.

3.3 *Scene Text Script Identification*

ICDAR2015 Competition on Video Script Identification

The CVSI-2015 [115] dataset comprises pre-segmented words in ten scripts: English, Hindi, Bengali, Oriya, Gujrathi, Punjabi, Kannada, Tamil, Telegu, and Arabic. The dataset contains about 1000 words for each script and is divided into three parts: a training set (60% of the total images), a validation set (10%), and a test set (30%). Text is extracted from various video sources (news, sports etc.) and, while it contains a few instances of scene text, it covers mainly overlay video text. Sample pre-segmented words are shown in Figure 3.7(a).

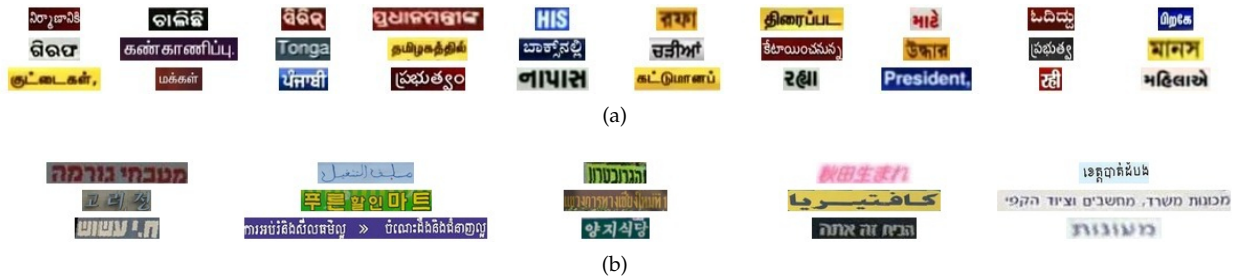


Figure 3.7: Sample images from the CVSI-2015 (a) and SIW-13 (b) datasets.

Script Identification in the Wild (SIW-13)

The SIW-13 dataset [116] comprises 16291 pre-segmented text lines in thirteen scripts: Arabic, Cambodian, Chinese, English, Greek, Hebrew, Japanese, Kannada, Korean, Mongolian, Russian, Thai, and Tibetan. The test set contains 500 text lines for each script, 6500 in total, and all the other images are provided for training. In this case, text was extracted from natural scene images from Google Street View. Sample text lines are shown in Figure 3.7(b).

3.4 *MLe2e multi-lingual end-to-end dataset*

This section introduces the Multi-Lingual end-to-end (MLe2e) dataset for the evaluation of scene text end-to-end reading systems and all intermediate stages: text detection, script identification, and text recognition. The MLe2e dataset has been harvested from various existing scene text datasets for which the images and ground-truth have been revised in order to make them homogeneous. The original images come from the following datasets: Multilanguage(ML) [107] and MSRA-TD500 [143] contribute Latin and Chinese text samples, Chars74K [20] and MSRRC [68] contribute Latin and Kannada samples, and KAIST [71] contributes Latin and Hangul samples.

In order to provide a homogeneous dataset, all images have been resized proportionally to fit in 640×480 pixels, which is the default image size of the KAIST dataset. Moreover, the ground-truth has been revised to ensure a common text line annotation level [59]. During this process human annotators were asked to review all resized images, adding the script class labels and text transcriptions to the ground-truth, and checking for annotation consistency: discarding images with unknown scripts or where all text is unreadable (this may happen because images were resized); joining individual word annotations into text line level annotations; discarding images where correct text line segmentation is not clear or cannot be established, and images where a bounding box annotation contains significant parts of more than one script or significant parts of background (this may happen with heavily slanted or curved text). Arabic numerals $(0, .., 9)$, widely used in combination with many (if not all) scripts, are labeled as follows. A text line containing text and Arabic numerals is labeled as the script of the text it contains, while a text line containing only Arabic numerals is labeled as Latin.

Figure 3.8: Sample images from the MLe2e dataset.

The MLe2e dataset contains a total of 711 scene images covering four different scripts (Latin, Chinese, Kannada, and Hangul) and a large variability of scene text samples. The dataset is split into a train and a test set with 450 and 261 images respectively. The split was done randomly, but in a way that the test set contains a balanced number of instances of each class (approx. 160 text lines samples of each script), leaving the rest of the images for the train set (which is not balanced by default). The dataset is suitable for evaluating various typical stages of end-to-end pipelines, such as multi-script text detection, joint detection and script identification, end-to-end multi-lingual recognition, and script identification in pre-segmented text lines. For the latter, the dataset also provides the cropped images with the text lines corresponding to each data split: 1178 and 643 images in the train and test set respectively.

While being a dataset that has been harvested from a mix of existing datasets it is important to notice that building it has supposed an important annotation effort: since some of the original datasets did not provide text transcriptions, and/or where annotated at different granularity levels. Moreover, despite the number of languages in the dataset is rather limited (only four scripts) it is one of the two first public datasets (along with ILST [120]) to cover the evaluation of all stages of multi-lingual end-to-end systems for scene text understanding in natural scenes. We think it is an important contribution of this paper and hope will be used by other researchers in the community.

3.5 *An on-line platform for ground truthing and performance evaluation of text extraction systems*

Coinciding with the 11th IAPR Workshop on Document Analysis Systems (DAS) the Computer Vision Center has released the CVC Annotation and Performance Evaluation Platform for Text Extraction (APEP-te) [59]: a set of on-line software tools that facilitate ground truthing and streamline performance evaluation over a range of text

extraction research tasks. The platform supports distributed ground truthing allowing multiple users to work in parallel, while maintaining centralized management and quality control of the process. It supports annotation at different text representation levels, from pixels to text lines. The platform supports the definition of evaluation scenarios for text localization, text segmentation and word recognition tasks, and brings together implementations of state of the art performance evaluation algorithms, on-line calculation of performance metrics and per image visualization of results.

The APEP-te platform has been used extensively for ground truth creation, while it provides the submission management, performance evaluation and results visualization functionality of the ICDAR 2011 and 2013 Robust Reading competitions. The on-line framework is available for public use. An installation pack that allows local deployment of the Web portal is available through http://www.cvc.uab.es/apep.

Chapter 4

Scene Text Extraction based on Perceptual Organization

A KEY CHARACTERISTIC OF TEXT is the fact that it emerges as a gestalt: a perceptually significant group of similar atomic objects. These atomic objects in the case of text are the character strokes giving rise to text-parts, be it well-separated characters, disjoint parts of characters, or merged groups of characters such as in cursive text. Such text-parts carry little semantic value when viewed separately (see Figure 4.1(a)), but become semantically relevant and easily identifiable when perceived as a group. Indeed, it can be shown that humans detect text without problems when perceptual organization is evident irrespectively of the script or language - actually they are able to do so for non-languages as well (see Figure 4.1(b)).

(a) (b) (c)

Figure 4.1: (a) Should a single character be considered "text"? (b) An example of automatically created non-language text[1]. (c) Our method exploits perceptual organization laws always present in text, irrespective of scripts and languages.

In this sense, text detection is an interesting problem since it can be posed as the detection of meaningful groups of regions, as opposed to the analysis and classification of individual regions. Still, the latter is the approach typically adopted in state of the art methodologies. Some methods do include a post-processing stage where identified text regions are grouped into higher level entities: words, text lines or paragraphs. This grouping stage is not meant to facilitate or complement the detection of text parts, but to prepare the already detected text regions for evaluation, as the ground truth is

[1] Daniel Uzquiano's random stroke generator: http://danieluzquiano.com/491

specified at the word or text line level. This mismatch of semantic level between results and ground truth is a recognized problem that has given rise to specific evaluation methodologies that intend to tackle the problem [138].

We pose that the fact that a region can be related to other neighboring ones is central to classifying the region as a text part, and is in-line with the essence of human perception of text, which is largely based on perceptual organization. The research hypothesis of this chapter is thus that building an automatic text detection process around the above fact can help overcome numerous identified difficulties of state of the art text detection methods.

To test the above hypothesis a state of the art perceptual organization computational model is employed to assess the meaningfulness of different candidate groups of regions. These groups emerge naturally through the activation of different visual similarity laws in collaboration with a proximity law.

Similarity between regions is not strictly defined in our framework. This is intentional, as due to design, scene layout, and environment effects different perceptual organization laws might be active in each case. Since text is a strong gestalt, a subset of such laws are expected to be active in parallel (collaborating) at any given instance. As a result a flexible approach is proposed, where various similarity laws are taken into account and the groups emerging through the individual activation of each similarity law provide the evidence to decide on the final set of most meaningful groups. The resulting method does not depend on the script, language or orientation of the text to be detected.

4.1 *Text Localization Method*

We present a region based method where an initial set of image regions are grouped together in a bottom-up manner guided by similarity evidence obtained over various modalities such as color, size, or stroke width among others, in order to obtain meaningful groups likely to be text gestalts, i.e. paragraphs, text lines, or words. Figure 4.2 shows the pipeline of the proposed algorithm for text extraction where the process is divided in three main steps: region decomposition, perceptual organization based analysis, and line formation.

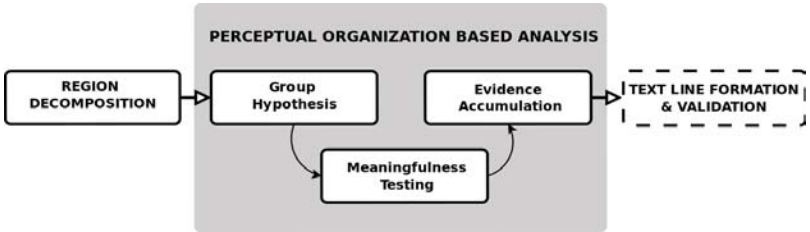


Figure 4.2: Text Extraction algorithm pipeline.

4.1.1 *Region Decomposition*

We make use of the Maximally Stable Extremal Regions (MSER) [80] algorithm for detecting text character candidates in the input images. The MSER algorithm builds a tree of regions with an extremal property of the intensity function over its outer boundary, and this property is normally present in all text parts as they are explicitly designed with high contrast in order to be easily read by humans.

The resulting MSER tree is pruned by filtering the regions that are not likely to be text parts by their size, aspect ratio, stroke width variance, and number of holes.

4.1.2 *Perceptual Organization Clustering*

The perceptual organization clustering is applied to the entire set of resulting MSERs in three stages. First we create a number of possible grouping hypotheses by examining different feature sub-spaces. Then, these groups of regions are analyzed and we keep the most meaningful ones, thus providing an ensemble of clusterings. Finally, those meaningful clusterings are combined based on evidence accumulation [36].

Group Hypothesis Creation

We aim to use simple and low computational cost features describing similarity relations between characters of a word or text line. The list of features we use for this kind of similarity grouping are:

Geometrical features. Characters in the same word usually have similar geometric appearance. We make use of the bounding box area, number of pixels, and diameter of the bounding circle.

Intensity and color mean of the region. We calculate the mean intensity value and the mean color, in the L*a*b* colorspace, of the pixels that belong to the region.

Intensity and color mean of the outer boundary. Same as before but for the pixels in the immediate outer boundary of the region.

Stroke width. To determine the stroke width of a region we make here use of the Distance Transform as in [13].

Gradient magnitude mean on the border. We calculate the mean of the gradient magnitude on the border of the region.

Each of these similarity features is coupled with spatial information, i.e. x, y coordinates of the regions' centers, in order to capture the collaboration of the proximity and similarity laws. So, independently of the similarity feature we consider, we restrict the groups of regions that are of interest to those that comprise spatially close regions.

Although it is usually expected that text parts belonging to the same word or text line share similar colors, stroke widths, and sizes, the previous assumption does not always hold (see Figure 4.3). This is why in our method we explore each of the similarity spaces in parallel.

(a) (b) (c)

Figure 4.3: There is no single best feature for character clustering: Characters in the same word may appear with different color (a), stroke width (b) or sizes (c).

We build a dendrogram using Single Linkage Clustering analysis for each of the feature sub-spaces described above. Each node in the obtained dendrograms represents a group hypothesis whose perceptual meaningfulness will be evaluated in the next step of the pipeline.

Meaningfulness Testing

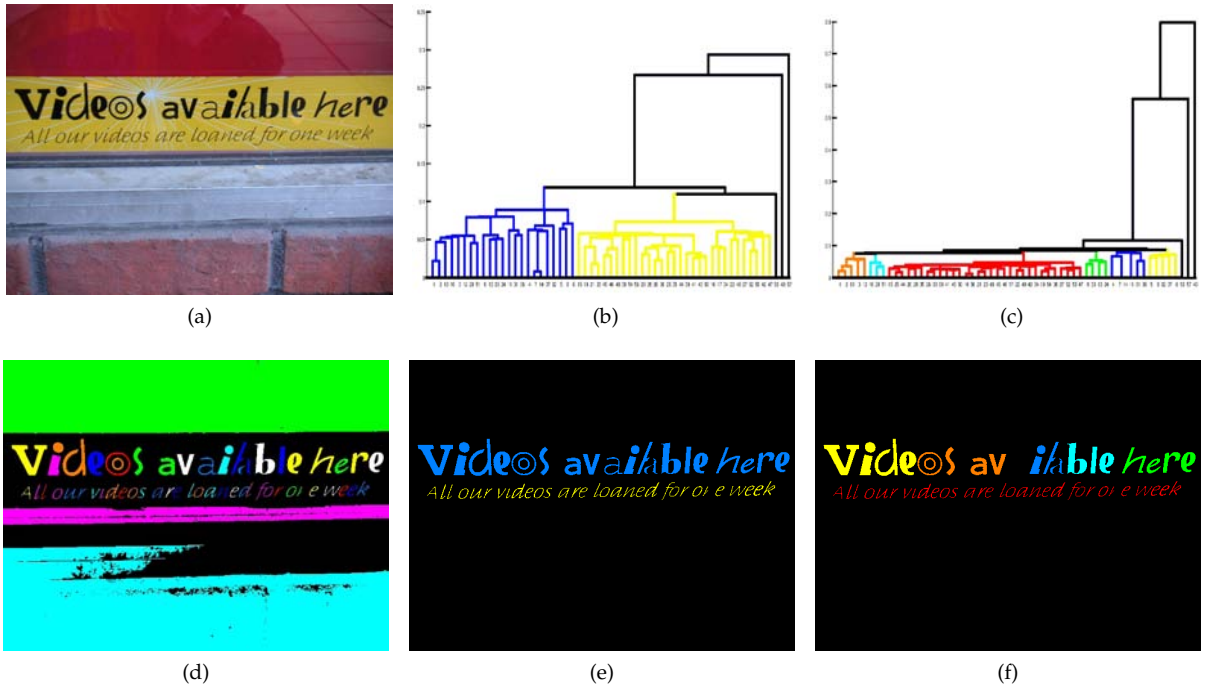
In order to find meaningful groups of regions in each of the defined feature sub-spaces we make use of a probabilistic approach to Gestalt Theory as formalized by Desolneux *et al.* [21]. The cornerstone of this theoretical model of perceptual organization is the Helmholtz principle, which could be informally summarized as: "We do not perceive anything in a uniformly random image", or with its equivalent *"a contrario"* statement: "Whenever some large deviation from randomness occurs in an image some structure is perceived". This general perception law, also known as the principle of common cause or of non-accidentalness, has been stated several times in Computer Vision, with Lowe [76] being the first to pose it in probabilistic terms.

The Helmholtz principle provides the basis to derive a statistical approach to automatically detect deviations from randomness, corresponding to meaningful events. Consider that n atomic objects are present in the image and that a group G of k of them have a feature in common. We need to answer the question of whether this common feature is happening by chance or not (and thus is a significant property of the group). Assuming that the observed quality has been distributed randomly and uniformly across all objects, the probability that the observed distribution for G is a random realization of this uniform process is given by the tail of the binomial distribution:

$$\mathcal{B}_G(k,n,p) = \sum_{i=k}^{n} \binom{n}{i} p^i (1-p)^{n-i} \tag{4.1}$$

where p is the probability of a single object having the aforementioned feature.

We make use of this metric in the dendrogram of each of the feature sub-spaces produced in the previous step separately to assess the meaningfulness of all produced grouping hypotheses. We calculate (4.1) for each node (merging step) of the dendrogram, using as p the ratio of the volume defined by the distribution of features of the samples forming the group with respect to the total volume of the feature sub-space. We then select as maximally meaningful a cluster A iif for every successor B and every ancestor C, it is $\mathcal{B}_B(k,n,p) > \mathcal{B}_A(k,n,p)$ and $\mathcal{B}_C(k,n,p) \geq \mathcal{B}_A(k,n,p)$. Notice that by using this maximality criteria no region is allowed to belong to more than one meaningful group at the same time. The clustering



(a) (b) (c)

(d) (e) (f)

Figure 4.4: (a) A scene image from the ICDAR2003 dataset, (b) its MSER decomposition. (c) Dendrogram of the feature sub-space (x, y coordinates, intensity mean), and (d) the maximal meaningful clusters found; (e)(f) same for the feature sub-space (x, y coordinates, stroke width).

analysis is done without specifying any parameter or cut-off value and without making any assumption on the number of meaningful clusters, but just comparing the values of (4.1) at each node in the dendrogram.

Figure 4.4 shows the maximal meaningful clusters detected in a natural scene image for two of the feature sub-sets defined, in Figure 4.4*(e)* image regions are clustered in a three dimensional space based on proximity and intensity value, while in Figure 4.4*(f)* they are clustered based on proximity and stroke width. This behavior, of arriving to different grouping results depending on the similarity modality examined is desirable, as it allows us to deal with variabilities in text design, illumination, perspective distortions, and so on.

Evidence Accumulation

Once we have detected the set of maximally meaningful clusters P^i in each feature sub-space $i \in N$, the clustering ensemble $\mathbb{P} = \{P^1, P^2, P^3, ..., P^N\}$ is used to calculate the evidence [36] for each pair of regions to belong to the same group, producing a co-occurrence matrix $\mathcal{D}$ defined as:

$$\mathcal{D}(i,j) = \frac{m_{ij}}{N} \tag{4.2}$$

where m_{ij} is the number of times the regions i and j have been assigned to the same maximal meaningful cluster in $\mathbb{P}$.

The co-occurrence matrix $\mathcal{D}$ is used as a dissimilarity matrix in order to perform the final clustering analysis of the regions, by applying the same hierarchical clustering process described in section 4.1.2.

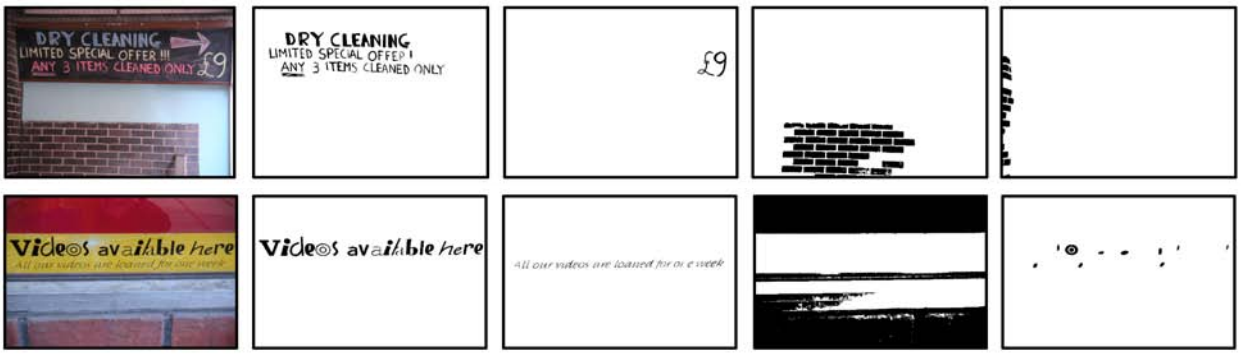


Figure 4.5: Maximally meaningful clusters validated through the Evidence Accumulation framework.

Figure 4.5 shows two examples of the results obtained where the method exhibits its ability to deal with a flexible definition of what a text group is: on the bottom row text characters do not have the same stroke width but they share the same color, while on the top, on the contrary, text characters do not have the same color but have similar stroke, size, etc.

As expected, not only text is detected as meaningful, but also any kind of region arrangement with a text like structure. It is important however to note at this stage that the algorithm produces relatively pure text-only clusters, with almost all text parts falling in clusters that contain virtually no non-text components. In order to filter non-text meaningful groups we use a combination of two classifiers. First, each region of the group is scored with the probability of being or not a character by a Real AdaBoost classifier using features combining information of stroke width, area, perimeter, number and area of holes. Then, simple statistics of this scores are fed into a second Real AdaBoost classifier for text/non-text group classification together with same features as before (but in this case for the whole group and not for independent regions) and a histogram of edge orientations of the Delaunay triangulation built with the group regions centers. Both classifiers are trained using the ICDAR2003 [78] and MSRA-TD500 [144] training sets as well as with synthetic text images, using different scripts, to ensure script-independence.

4.2 *Experiments and Results*

The proposed method has been evaluated on three multi-script datasets and one English-only dataset for different tasks, in one hand for text segmentation on the KAIST and MSRRC datasets, and on the other for text localization in the MSRA-TD500 and ICDAR2003 datasets (see Chapter 3 for datasets details).

Text Segmentation

Table 4.1 show the obtained results on the KAIST and MSRRC datasets. Sample qualitative results are shown in Figure 4.6, where it can be appreciated the ability of our method to robustly extract regions in

difficult cases like curved and multi-colored text.

	KAIST			MSRRC		
	p	r	f	p	r	f
Lee *et al.*[71]	**0.69**	0.60	0.64	-	-	-
OTCYMIST [69]	0.52	0.61	0.56	0.50	0.29	0.37
Sethi *et al.*[68]	-	-	-	0.33	0.72	0.45
Yin *et al.*[146, 68]	-	-	-	**0.71**	**0.67**	**0.69**
This Chapter	0.67	**0.78**	**0.71**	0.64	0.58	0.61

Table 4.1: Scene text segmentation results in KAIST and MSRRC datasets.

The method presented in this Chapter participated as a competing entry in the 2013 Multi-script Robust Reading Competition (MSRRC) reaching the second place. While, as can be appreciated in Table 4.1, the winner of the competition [146] show better numbers in both precision and recall metrics, our proposal outperformed the other two entries with a noticeable margin. Consistently, our method also outperforms Lee *et al.* [71] and OTCYMIST [69] methods in KAIST dataset.

Text Detection

Since the perceptually meaningful text groups detected by our method rarely correspond directly to the semantic level ground truth information is defined in (words in the case of ICDAR, and lines in the case of the MSRA-TD500 dataset), the proposed method is extended with a simple post-processing step in order to obtain text line level bounding boxes.

We consider a group of regions as a valid text line if the mean of the y-centers of its constituent regions lies in an interval of 40% around the y-center of their bounding box and the variation coefficient of their distribution is lower than 0.2. Notice that in MSRA-TD500, as we are considering text lines at any possible orientation, the orientation of the group (and consequently the definition of the

Figure 4.6: Qualitative segmentation results on sample images of the KAIST (top) and MSRCC (bottom) datasets.

y-axis) is always defined in relation to the axes of the circumscribed rectangle of minimum area for the given group. If the collinearity test fails, it may be the case that the group comprises more than one text line. Thus in such a case we perform a histogram projection analysis in order to identify the text lines orientation, and then split the initial group into possible lines by clustering regions on the identified direction. This process is iteratively repeated until all regions have either been assigned to a valid text line or rejected, using the collinearity test described above, or until no more partitions can be found.

Table 4.2 show a comparison of the obtained results with other state of the art methods on the MSRA-TD500 and ICDAR2003 datasets. Sample qualitative results are shown in Figure 4.7

	MSRA-TD500			ICDAR2003		
	P	R	f	P	R	f
Chen *et al.* [14]	0.05	0.05	0.05	0.60	0.60	0.58
Epshtein *et al.* [27]	0.25	0.25	0.25	**0.73**	0.60	0.66
Li *et al.*[74]	0.30	0.32	0.31	0.45	**0.80**	0.57
TD-ICDAR [144]	0.53	0.52	0.54	0.68	0.66	0.66
TD-Mixture [144]	**0.63**	**0.63**	**0.60**	0.69	0.66	**0.67**
This Chapter	0.58	0.54	0.56	0.71	0.57	0.64

Table 4.2: Scene text localization results (precision, recall, and f-score) in MSRA-TD500 and ICDAR2003 datasets.

While our method is outperformed by TD-Mixture [144] in both text localization datasets, the most important outcome from the comparison in Table 4.2 is that our results are consistent with the claim of being script and orientation independent. Therefore, we see that the method presented in this Chapter outperforms other methods in MSRA-TD500 that are designed with only horizontal text in mind, despite some of them perform better that us in ICDAR2003.

By analyzing the errors of our method in both localization and segmentation tasks we have found that in many cases we fail to detect small texts. Something that can be explained by the fact that the meaningfulness measure does not find those small clusters perceptually meaningful. On the other hand, we also found that some times the late fusion of the different similarity modalities through the evidence accumulation algorithm, while helps in finding consensus and thus in increasing precision by removing duplicate detections, also

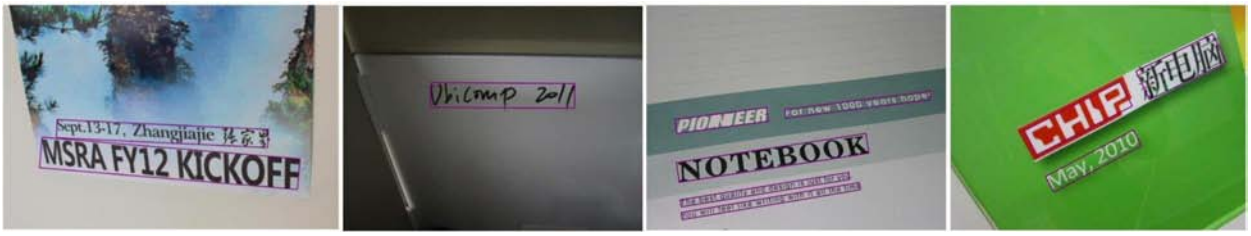


Figure 4.7: Qualitative localization results on sample images of the MSRA-TD500 dataset.

removes groups that are detected as meaningful only in one of the similarity cues. This effectively reduces the potential recall rate of the method. To mitigate this weaknesses in next Chapter we explore the idea of an early fusion of the similarity modalities, while we also complement the meaningfulness test presented in this Chapter with an efficient discriminative classifier.

4·3 *Conclusion*

In this chapter we have proposed a new methodology for scene text extraction inspired by the human perception of textual content, largely based on perceptual organization. The proposed method requires practically no training as the perceptual organization based analysis is parameter free. It is totally independent of the language and script in which text appears, it can deal efficiently with any type of font and text size, while it makes no assumptions about the orientation of the text. Experimental results demonstrate competitive performance when compared with state of the art.

Chapter 5

Optimal design and efficient analysis of similarity hierarchies

Hierarchical organization is an essential feature of text. Induced by typography and layout the hierarchical arrangement of text strokes leads to the structural formation of text component groupings at different levels (e.g. words, text lines, paragraphs, etc.), see Figure 5.1. This hierarchical property applies independently of the script, language, or style of the glyphs, thus it allows us to pose the problem of text detection in natural scenes in a holistic framework rather than as the classification of individual patches or regions as text or non-text.

Hierarchies are well-studied structures in mathematics and computer science, where they are used extensively as data structures. They have several interesting properties that make them suitable (desirable) for many different tasks, e.g. tree searches, cluster analysis, knowledge representation, etc. One particularly interesting property of hierarchies, stemming from their recursive definition, is the inclusion relation that exists between every node and its ancestors on the hierarchy. This property can be exploited for example to efficiently update cluster distances at each step of a hierarchical clustering analysis[1] or, in a similar way, for the incremental computation of certain cluster features [81, 131, 98].

In this Chapter we investigate how the recursive nature of hierarchical structures can be exploited for scene text detection.

[1] See the Lance–Williams family of agglomerative hierarchical clustering algorithms.

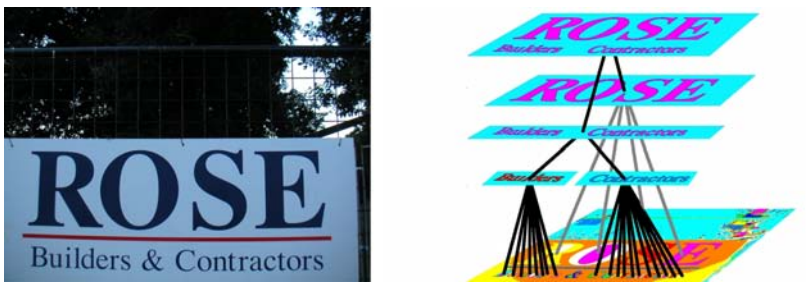


Figure 5.1: A natural scene image and a hierarchical representation of its text. Atomic objects (characters) extracted in the bottom layer are agglomerated into text groupings at different levels of the hierarchy.

Figure 5.2: Individual text-parts are less distinguishable when viewed separately, but become structurally relevant and easily identifiable when perceived as a group.

For this, we follow in essence the same intuitions developed on the previous Chapter: we tackle directly the problem of detecting groups of text regions, instead of the classification of individual regions, see Figure 5.2; and the proposed method is again driven by an agglomerative clustering process exploiting the strong similarities that are expected between text components in such groups.

However, we take here a step further on exploiting the hierarchical structure of text grouping proposals, and make the following contributions in order to build a more robust and faster method: First, we learn an optimal feature space that encodes the similarities between text components thus allowing the Single Linkage Clustering algorithm to generate text group hypotheses with high recall, independently of their orientations and scripts. Second, we couple the hierarchical clustering algorithm with a discriminative stopping rule, that allow the efficient detection of text groups in a single clustering step. Third, we propose a new set of features for text group classification, that can be efficiently calculated in an incremental way, able to capture the inherent structural properties of text related to the arrangement of text parts and the intra-group similarity. The use of incrementally computable group descriptors makes possible the direct evaluation of all group hypotheses generated by the clustering algorithm without affecting the overall time complexity of the method.

5.1 *Hierarchy guided text extraction*

Our hierarchical approach to text extraction involves an initial region decomposition step where non-overlapping atomic text parts are identified. These regions are then grouped with an agglomerative process lead by their similarity, producing a dendrogram where each node represents a text group hypothesis. We can then find the branches corresponding to text groups by simply traversing the dendrogram with an efficient stopping rule. Figure 5.3 shows an example of the main steps of the pipeline.

As in the previous Chapter we make use of the MSER [80] algorithm to get the initial set of low-level text regions candidates.

Recall in character detection is increased by extracting regions from different single channel projections of the image (i.e. gray, red, green and blue channel). This technique, denoted MSER++ [97], is a way of diversifying the segmentation step in order to maximize the chances of detecting all text regions.

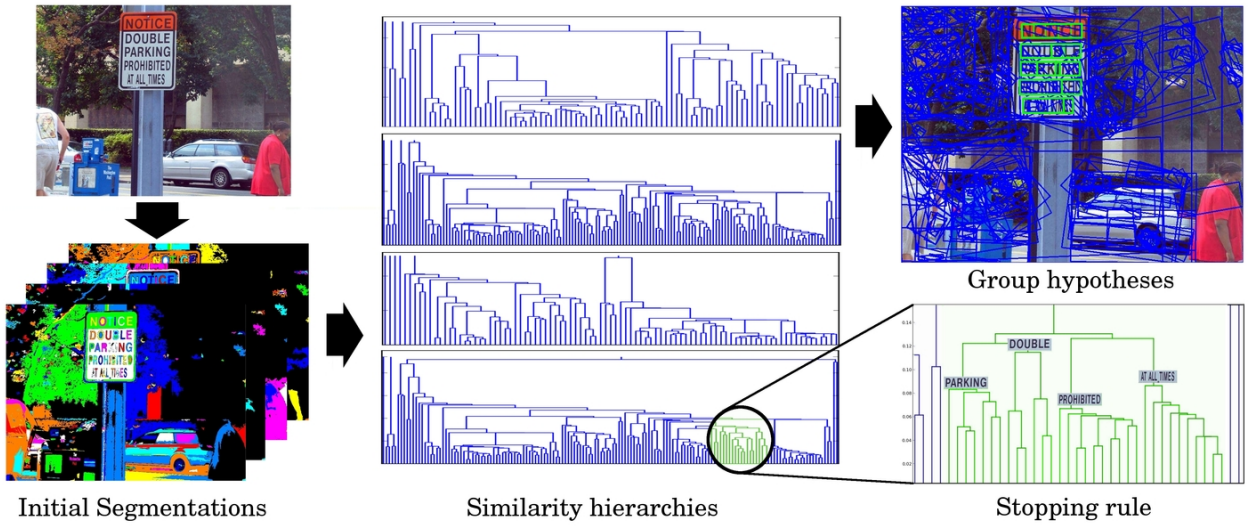


Figure 5.3: A bottom-up agglomerative clustering of individual regions produces a dendrogram in which each node represents a text group hypothesis. Our work focuses on learning the optimal features allowing the generation of pure text groups (comprising only text regions) with high recall, and designing a stopping rule that allows the efficient detection of those groups in a single grouping step.

5.1.1 *Optimal clustering feature space*

Our agglomerative grouping algorithm is based on the fact that regions belonging to the same word or text line necessarily have some properties in common that make such a group perceptually meaningful. As in previous Chapter we make use of low computational cost features to define similarity measures between characters of a word or text line. The list of similarity measures used here is as follows: size (major axis of regions' fitting ellipse), intensity mean of the regions, intensity and gradient magnitude mean on the border of the regions.

In addition to those similarity measures we make use of spatial information, i.e. x and y coordinates of the regions' centers. This way we restrict the groups of regions that are of interest to those that comprise spatially close regions.

Distinctly from previous Chapter, now we consider that it is possible to weight those simple similarity features obtaining an optimal feature space projection that maximizes the probabilities of finding pure text groups (groups comprising only regions that correspond to text parts) in a Single Linkage Clustering (SLC) dendrogram.

Let $\mathcal{R}_c$ be the initial set of individual regions extracted with the MSER algorithm from channel c. We start an agglomerative clustering process, where initially each region $r \in \mathcal{R}_c$ starts in its own cluster and then the closest pair of clusters (A, B) is merged iteratively, using the single linkage criterion $(\min\{\, \mathrm{d}(r_a, r_b) : r_a \in A,\, r_b \in B \,\})$, until all regions are clustered together $(C \equiv \mathcal{R}_c)$. The distance between two regions $\mathrm{d}(r_a, r_b)$ is calculated as the squared Euclidean distance between their weighted feature vectors, adding a spatial constraint term (the squared Euclidean distance between their centers' coordinates c_a and c_b) in order to induce neighboring regions to

merge first:

$$\mathrm{d}(r_a, r_b) = \|c_a - c_b\|_2^2 + \sum_{i=1}^{D} (w_i \cdot (a_i - b_i))^2 \tag{5.1}$$

where we consider the 5-dimensional similarity space ($D = 5$) comprising the features described before: mean gray value of the region, mean gray value in the immediate outer boundary of the region, region's major axis, mean stroke width, and mean of the gradient magnitude at the region's border. This way, a region ($r_a \in \mathcal{R}_c$) is defined by its center coordinates (c_a) and its feature vector ($\mathbf{a}$), with everything being rescaled to the range $[0, 1]$: dividing the features in $\mathbf{a}$ by the maximum values found in training data, and the coordinates in c_a by the image width/height respectively.

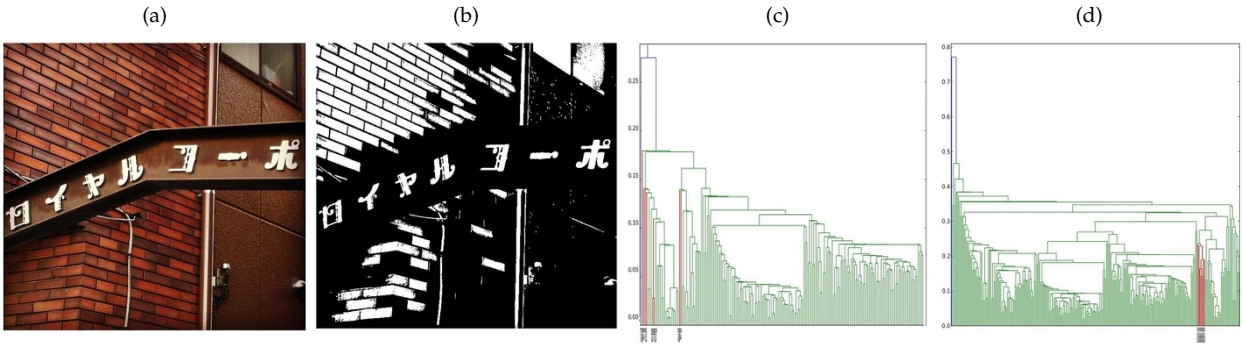
It is worth noting that using the squared Euclidean distance for the spatial term in equation 5.1, our clustering analysis remains rotation invariant, thus the obtained hierarchy generates the same text group hypotheses independently of the image orientation. For example, rotating the image in Figure 5.4*(a)* by any degree would produce exactly the same SLC dendrograms shown in Figures 5.4*(c)* and 5.4*(d)*. This is intentional as we want our method to be capable of detecting text in arbitrary orientations. In this way, our algorithm deals naturally with arbitrary oriented text without using any heuristic assumption or threshold.

An alternative formulation for an "orientation dependent" distance would be to include the center coordinates in the feature vector, thus allowing the x and y coordinates to weight differently in the distance sum. We evaluate this approach in Section 5.2 for the case of horizontally-aligned text datasets.

Given a possible set of weights $\mathbf{w}$ in equation 5.1, the SLC algorithm produces a dendrogram D_w where each node $H \in D_w$ is a subset of $\mathcal{R}_c$ and represents a text group hypothesis. The text group recall represents the ability of a particular weighting configuration to produce pure text groupings (comprising only text regions) corresponding to words or text lines in the ground-truth. Figure 5.4 shows an example of how different weight configurations lead to different recall rates.

We make use of a metric learning algorithm in order to learn the optimal weights $\mathbf{w}$ in equation 5.1 and maximize the text group recall

Figure 5.4: (a) Scene image, (b) its MSER decomposition, and (c,d) two possible hierarchies built from two different weight configurations, red nodes indicate pure text groupings.



in our training dataset. For this we define the following loss function:

$$L(\mathbf{w}) = \sum_{r_q, r_p, r_n} I(\mathrm{d}(r_q, r_p) > \mathrm{d}(r_q, r_n)) \tag{5.2}$$

where $I(g) = 1$ if g is true or $I(g) = 0$ otherwise, and the triplets $\{r_q, r_p, r_n\}$ stand for regions in $\mathcal{R}_c$ with the following relation: $\{r_q, r_p\}$ are two regions that are part of the same text group (i.e. word or text line), and $\{r_n\}$ is a region that do not belong to their group.

To find the $\mathbf{w}$ that minimizes the loss $L(\mathbf{w})$ we use stochastic gradient descent (SGD) to minimize the convex surrogate $L_S(\mathbf{w}) \geq L(\mathbf{w})$:

$$L_S(\mathbf{w}) = \sum_{r_q, r_p, r_n} max(0, \mathrm{d}(r_q, r_p) - \mathrm{d}(r_q, r_n)) \tag{5.3}$$

Thus, our SGD update rule is:

$$\mathbf{w} \leftarrow \mathbf{w} - \lambda \frac{\partial L_S(\mathbf{w})}{\partial \mathbf{w}}\bigg|_{r_q, r_p, r_n} \tag{5.4}$$

which evals to:

$$\begin{cases} \mathbf{w} & if\ \mathrm{d}(r_q, r_p) < \mathrm{d}(r_q, r_n) \\ \mathbf{w} - \lambda[(\mathbf{q} - \mathbf{p})^{\circ 2} - (\mathbf{q} - \mathbf{n})^{\circ 2}] & otherwise \end{cases} \tag{5.5}$$

where $\mathbf{q}$, $\mathbf{p}$, $\mathbf{n}$ are the feature vectors of regions r_q, r_p, and r_n respectively, and $(\cdot)^{\circ 2}$ stands for the Hadamard power (element-wise) operation.

We have assembled a mixed set of training examples using the MSRRC and ICDAR training sets, which contain 167 and 229 images respectively. We have manually separated all text-lines and words in the ground truth data of these images, giving rise to 1611 examples of text groups. Figure 5.5 shows the group examples extracted from one of the training set images. Since MSRRC and ICDAR datasets already provide pixel level segmentation ground truth, the manual work performed to get our training data was limited to separate the individual groups (words and text lines).

The use of a single mixed training set for learning the optimal weights $\mathbf{w}$ is intentional, as we want to capture the importance of each similarity measure in equation 5.1 in a domain independent manner.

At training time, we perform SGD over the whole train set in the following way: at each iteration we randomly pick an image and a region r_q that belongs to a text group (i.e. word or text line) in its ground-truth annotation. Then, r_p is selected as the closest contiguous region in the same text group as r_q, and r_n is selected as the region with the smaller distance $\mathrm{d}(r_q, r_n)$ among the MSER regions extracted from the image that do not overlap with any ground-truth region in same group of r_q. The learning rate λ has been set to a small value by carefully inspection, in order to not allow the weights in $\mathbf{w}$ reach negative values in any case.

Figure 5.5: Our training set is assembled by manually separating the pixel level ground-truth of train images into all possible text groups (lines and words).

As discussed before, there is no single best way to define similarity between text parts, hence there is no single best set of weights for our strategy, instead missing groups under a particular configuration may be potentially found under another. An alternative to using a single feature space would be to diversify our clustering strategy, adding more hypotheses to the system by building different hierarchies obtained from different weight configurations (similarly to what we do with different color channels). As diversification strategy, after an optimized set of weights $\mathbf{w}_{opt}$ is obtained we subsequently remove from the training set the groups that have been correctly detected, and then learn again new optimal weights $(\mathbf{w}_{opt_2}, \ldots, \mathbf{w}_{opt_n})$ with the remaining groups. We evaluate this diversification strategy in section 5.2.

At test time, each of the optimal weight configurations is used to generate a dendrogram where each node represents a text group hypothesis. Selecting the branches corresponding to text groups is done by traversing the dendrogram with an efficient stopping rule.

5.1.2 *Discriminative and Probabilistic Stopping Rules*

Given a dendrogram representing a set of text groups hypotheses from the SLC algorithm, we need a strategy to determine the partition of the data that best fits our objective of finding pure text groups. A rule to decide the best partition in a Hierarchical Clustering is known as a *stopping rule* because it can be seen as stopping the agglomerative process. Differently from standard clustering stopping rules here we do not expect to obtain a full partition of regions in $\mathcal{R}_c$. In fact we do not even know if there are any text clusters at all in $\mathcal{R}_c$. Moreover, in our case we have a quite clear model for the kind of groups sought, corresponding to text. These particularities motivate the next contribution of this paper. We propose a stopping rule, to select a subset of meaningful clusters in a given dendrogram D_w, comprising the combination of the following two elements:

- A text group discriminative classifier.
- A probabilistic measure for hierarchical clustering validity assessment [11].

Discriminative Text Group Classifier

The first part of our stopping rule takes advantage of supervised learning, building a discriminative classification model $\mathcal{F}$ in a group-level feature space. Thus, given a group hypothesis H and its feature vector $\mathbf{h}$, our stopping rule will accept H only if $\mathcal{F}(\mathbf{h}) = 1$. We use a Real AdaBoost classifier [111] with decision stumps. Our group-level features originate from four different types: 1) Color and edge intra-similarity statistics, since we expect to see regions in the same word having low variation in color and contrast to their background; 2) Geometric intra-similarity statistics, same as before for the size of the regions; 3) Shape similarity of participating regions, in order to discriminate repetitive patterns, such as bricks or windows in a building, which tend to be confused with text; 4) Structural collinearity and equidistance features, measure the text-like structure of text groups by using statistics of the 2-D Minimum Spanning Tree (MST) built with their regions centers. The list of the 14 used features is as follows:

Color and edge features:

- Foreground intensities standard deviation.
- Background intensities standard deviation.
- Mean gradient standard deviation.

Geometric features:

- Major axis coefficient of variation.
- Aspect ratios coefficient of variation.

Shape features:

- Stroke widths [13] coefficient of variation.
- Hu's invariant moments [50] average Euclidean distance.
- Convex hull compactness [98] mean and standard deviation.
- Convexity defects coefficient of variation.

Structural features:

- MST angles mean and standard deviation.
- MST edge distances coefficient of variation.
- MST edge distances mean vs. regions' diameters mean ratio.

The AdaBoost classifier is trained using the same training set described in section 5.1.1. We have two sources of positive samples: 1) Using each GT group as if it were the output of the region decomposition step; 2) we run MSER and SLC (w_{opt}) against a train image

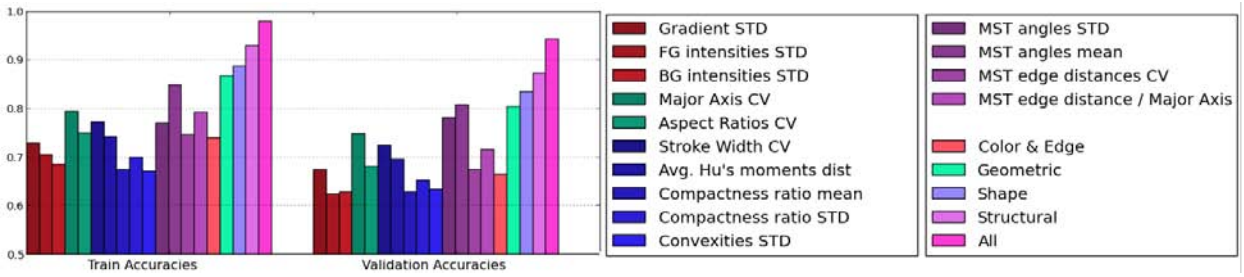


Figure 5.6: Classification accuracies of the text group classifier using different features.

and use as positive samples those pure-text groups in the SLC tree with more than 80% match with a GT group. From the same tree we extract negative examples as nodes with 0 matchings. This gives us around 3k positive and 15k negative samples. We balance the positive and negative data and train a first classifier that is used to select around 500 hard negatives that are used to re-train and improve accuracy.

Figure 5.6 shows the accuracy of the AdaBoost classifier using different features in order to visualize the contribution of each group feature in the classifier performance. Reported accuracies are obtained by dividing the training samples described before in two sets for train (80%) and test (20%). The final test accuracy using all 18 group features is 94.25%.

In order to compare our text group classifier with other state of the art approaches for text classification we evaluate the performance of the T-HOG gradient-based descriptor [89] using exactly the same data as in Figure 5.6. In these conditions T-HOG obtains a 94.56% test accuracy, just slightly better than ours. However, it is very important to notice here that the computational time needed by T-HOG (as well as many other descriptors in the literature) makes their use unfeasible in our pipeline, since its computation cannot be performed in an incremental way. The processing time needed by T-HOG to evaluate the whole hypotheses dendrograms in our pipeline was on average 32 seconds, while our method is able to process the same number of hypotheses in 1.6 seconds. In this sense what makes our classifier a better choice is not only its discriminative power but also the ability to be calculated efficiently over the whole set of group hypotheses defined by the SLC analysis, as will be explained next.

Incrementally computable descriptors

Since at test time we have to calculate the group level features at each node of the similarity hierarchy, it is important that they are fast to compute. We take advantage of the inclusion relation of the dendrogram's nodes in order to make such features incrementally computable when possible. This allows us to compute the probability of each possible group of regions to be a text group without affecting the overall time complexity of our algorithm.

Group level features consisting of simple statistics over individ-

ual region features (e.g. diameters, strokes, intensity, etc.) can be incrementally computed straightforwardly with a few arithmetic operations and so have a constant complexity $\mathcal{O}(1)$.

Regarding the MST based features, an incremental algorithm (i.e. propagating the MST of children nodes to their parent) computing the MST on each node of the dendrogram takes $\mathcal{O}(n \times \log^2 n)$ in the worst case. Although this complexity is much lower than the $\mathcal{O}(n^2)$ complexity of the SLC step and thus does not affect the overall complexity of the algorithm, this has noticeable impact in run time. For this reason we add an heuristic rule on the maximum size of valid clusters: clusters with more than a certain number of regions are immediately discarded and there is no need to compute their features. By taking the length of the largest text line in the MSRRC training set (50) as the maximum cluster size, the run-time growth due to the features calculation in our algorithm is negligible and the obtained results are not affected at all.

Probabilistic cluster meaningfulness estimation

The way our classifier $\mathcal{F}$ is designed may eventually make the discriminative stopping rule to accept groups with outliers. For example, Figure 5.7 shows the situation where a node of the dendrogram consisting in a correctly detected word is merged with a (character like) region which is not part of the text group (outlier). In order to increase the discriminative power of our *stopping rule* in such situations, we make use of a probabilistic measure of cluster meaningfulness [21, 11]. This probabilistic measure, also used in previous Chapter, provides us with a way to compare clusters' qualities in order to decide if a given node in the dendrogram is a better text candidate than its children.

The Number of False Alarms ($\mathcal{NFA}$) [21, 11], based on the principle on non-accidentalness, measures the meaningfulness of a particular group of regions in $\mathcal{R}_c$ by quantifying how the distribution of their features deviates from randomness.

$$\mathcal{NFA}(H) = \mathcal{B}_G(k, n, p) = \sum_{i=k}^{n} \binom{n}{i} p^i (1-p)^{n-i} \tag{5.6}$$

where n is the number of regions in $\mathcal{R}_c$, k is the number of regions in the group hypothesis H we are evaluating, and p is the probability that the feature vector of a randomly selected region from $\mathcal{R}_c$ falls in the volume defined by the distribution of the feature vectors of the comprising regions ($h \in H$) in the $5-D$ feature space defined in Section 5.1.1. The lower the $\mathcal{NFA}$ is, the more meaningful H is.

Our stopping rule is defined recursively in order to accept a particular hypothesis H as a valid group *iif* its classifier predicted label is "text" ($\mathcal{F}(\mathbf{h}) = 1$) and its meaningfulness measure is higher than the respective meaningfulness measures of every successor $A \in suc(H)$ and every ancestor $B \in anc(H)$ labeled as text, i.e. the following

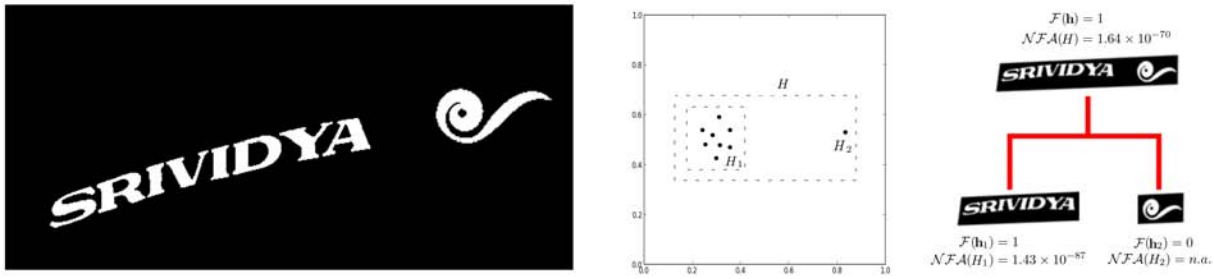


Figure 5.7: A node in a similarity dendrogram consisting in a correctly detected word (H_1) is merged with a cluster consisting of a single region outlier (H_2). Our stopping rule will not consider valid the resulting cluster $H = \{H_1 \cup H_2\}$ although the classifier has labeled it as a text group ($\mathcal{F}(\mathbf{h}) = 1$) because $\mathcal{NFA}(H)$ is larger than $\mathcal{NFA}(H_1)$. The scatter plot simulates the arrangement of the feature vectors of the regions forming H_1, H_2, and H in the similarity feature space.

inequalities hold:

$$\mathcal{NFA}(H) < \mathcal{NFA}(A), \forall A \in suc(H) \mid \mathcal{F}(A) = 1 \tag{5.7}$$

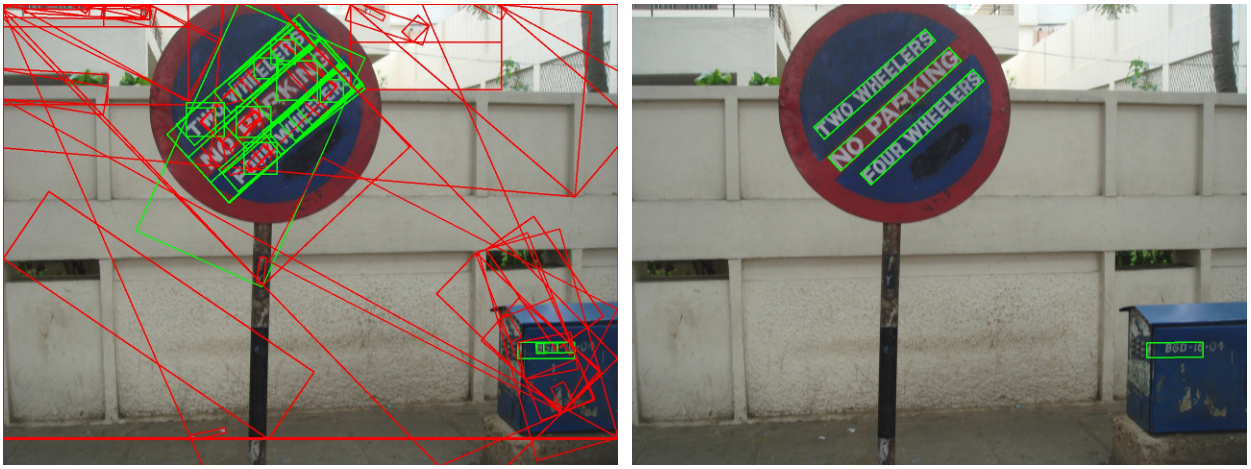
$$\mathcal{NFA}(H) < \mathcal{NFA}(B), \forall B \in anc(H) \mid \mathcal{F}(B) = 1 \tag{5.8}$$

Again it is important to notice that by using this criteria no region is allowed to belong to more than one text group at the same time. Thus, the final selection of non-overlapping text clusters that conform the output of our method is done in a parameter-free procedure, just by comparing the values of (5.6) at all nodes in the dendrogram that are labeled as "text" by the discriminative classifier ($\mathcal{F}(H) = 1$), without making any assumption on the number of desired clusters. Figure 5.8 shows the effect of the $\mathcal{NFA}$ selection criteria over the output of the discriminative classifier in a particular image hypotheses tree. See Figure 5.7 for an synthetic example on how this stopping rule is able to detect outliers. As a side effect, the stopping rule is eventually able to correctly separate different words in a text line.

At this point, applying the method described so far our algorithm is able to produce results for the scene text segmentation task. The segmentation task is evaluated at pixel-level, this is the algorithm must provide a binary image where white pixels correspond to text and black pixels to background. All segmentation results given in section 5.2 are obtained with this algorithm, trained with a single mixed dataset and without any further post-processing, by setting to white the pixels corresponding to the detected text groups. Figures 5.9 and 5.11 show segmentation results of our method in the MSRRC and KAIST datasets.

5.1.3 *From Pixel Level Segmentation to Bounding Box Localization*

In order to evaluate our method in the text localization task we extend our method with a simple post-processing to obtain word and text line bounding boxes depending on the semantic level ground truth information is defined (e.g. words in the case of ICDAR and MSRRC datasets, lines in the case of the MSRA-TD500 dataset). This is because the text groups detected by our stopping rule may correspond indistinctly to words, lines, or even paragraphs in some cases,



(a) (b)

Figure 5.8: (a) Selective search of text groups defined by our SLC analysis, each rectangle represents a text group hypothesis: in green, groups labeled as text by the discriminative classifier, in red, groups labeled as non-text; (b) Text groups selected with the $\mathcal{NFA}$ criteria over the hypotheses tree.

depending on the particular typography and layout of the detected text.

First of all, region groups selected as text by our stopping rule in the different dendrograms are combined in a procedure that serves to deduplicate repeated groups (e.g. the same group may potentially be found in several channels or weights configurations) and to merge collinear groups that may have been detected by chunks. Two given text groups are merged if they are collinear, and their relative distance and height ratio are under thresholds learned during training.

After that, if needed by the granularity of the ground-truth level, we split resulted text lines into words by considering as word boundaries all spaces between regions with a larger distance than a certain threshold, learned during training, proportional to the group's average inter-region distance.

5.2 *Experiments*

The proposed method has been extensively evaluated on three multi-script and arbitrary oriented text datasets and two English-only datasets for the tasks of scene text segmentation and localization.

5.2.1 *Baseline analysis*

We have evaluated different variants of our method in order to assess the contribution of each of the proposed techniques. This baseline analysis is performed in the MSRRC test set. The baseline method is configured by setting all weights to 1 (w_1) and accepting all group hypotheses which are labeled as text by the classifier ($\mathcal{F}(H) = 1$). We compare this baseline with the variants making use of the learned optimal weights w_{opt}, and with including the meaningfulness criteria to our *stopping rule*. Finally we have evaluated the impact of different diversification strategies to the initial segmentation, both in the number of image channels (MSER vs. MSER++), and in the

number of weight configurations by adding a variable number of optimal weighted configurations $w_{opt_2}, \ldots, w_{opt_n}$ into the system. Table 5.1 shows segmentation results of our method in the MSRRC 2013 test set comparing different variants of our method and different diversification strategies. The table includes also two variants of our previously published work [42] for comparison. We chose to use the MSRRC dataset for this analysis as it is representative of the targeted scenario of multi-script and arbitrarily oriented text.

Method	Precision	Recall	F-score
Previous Chapter [42, 68]	0.64	0.58	0.61
Previous Chapter MSER++	0.50	0.71	0.59
MSER | w_I	0.69	0.58	0.63
MSER | w_{opt}	0.69	0.62	0.65
MSER | w_I | *stop-rule*	0.76	0.60	0.67
MSER | w_{opt} | *stop-rule*	**0.77**	0.62	0.69
MSER++ | w_{opt} | *stop-rule*	0.75	0.71	**0.73**
MSER++ | $w_{opt}, w_{opt_2}, \ldots, w_{opt_4}$ | *stop-rule*	0.67	0.73	0.70
MSER++ | $w_{opt}, w_{opt_2}, \ldots, w_{opt_7}$ | *stop-rule*	0.56	**0.74**	0.64

Table 5.1: Segmentation results comparing different variants of our method.

From the obtained results we can see that the optimized weights w_{opt} have a noticeable impact in the method recall, while the *stopping rule* leads to a considerable increase in precision without any recall deterioration. Regarding diversification, if one wants to maximize the harmonic mean between precision and recall, the use of MSER++ is well justified even though it produces a slight precision drop. However, examining the effect of further diversification using more optimal weighting configurations, we can see that the obtained gain in recall by adding more hypotheses does not help improving the f-score as it produces a significant precision deterioration. Such a diversification strategy should be considered only if one wants to maximize the system's recall.

5.2.2 *Scene text segmentation results*

Table 5.2 compares the final results of our method with the state-of-the-art on the scene text segmentation task in three different benchmarks (KAIST, MSRRC, and ICDAR2013). A set of example qualita-

Figure 5.9: Qualitative segmentation results on the MSRRC 2013 dataset.

tive results are shown in Figures 5.9 and 5.11.

	KAIST			MSRRC			IC13		
	p	r	f	p	r	f	p	r	f
Previous Chapter	0.67	0.78	0.71	0.64	0.58	0.61	0.63	0.59	0.61
NUS FAR [60] *	-	-	-	-	-	-	**0.82**	**0.75**	**0.78**
Lee *et al.*[71]	**0.69**	0.60	0.64	-	-	-	-	-	-
NSTextractor [60]	-	-	-	-	-	-	0.76	0.61	0.68
OTCYMIST [69]	0.52	0.61	0.56	0.50	0.29	0.37	0.46	0.59	0.52
Sethi *et al.*[68]	-	-	-	0.33	0.72	0.45	-	-	-
TextDetect. [60]	-	-	-	-	-	-	0.76	0.65	0.70
FuStar [146]	-	-	-	-	-	-	0.74	0.70	0.72
Yin *et al.*[146, 68]	-	-	-	0.71	0.67	0.69	-	-	-
This Chapter	0.67	**0.89**	**0.76**	**0.75**	**0.71**	**0.73**	0.74	0.71	0.73
(Horiz. only)	-	-	-	-	-	-	0.77	0.73	0.75

Table 5.2: Segmentation results in standard datasets. Methods marked with * have not been published up to date.

As can be seen in Table 5.2, our method outperforms previous state-of-the-art in KAIST and MSRRC benchmarks, while providing a competitive results in the ICDAR dataset. In the case of horizontally aligned text datasets we also provide results of a specialized version of our method (Horiz. only) by using the 'orientation dependent' distance explained in section 5.1.1 and filtering detections with non-horizontal orientations.

Figure 5.10 show the inverse grade curves of different methods in the MSRRC dataset. The inverse grade curve plots the f-score divided by the ratio of text pixels for each image, and inversely sorts these values by the amount of text pixels, thus larger values in the x-axis correspond to images with less text. As can be seen our curve is the nearest to follow the ground-truth benchmark curve.

The interpretation of the high increase in recall observed in the KAIST dataset compared to the obtained in MSRRC and ICDAR follows the fact that in KAIST dataset small text characters are not labeled in the ground-truth. These small text components are in general the ones more difficult to detect. On the other hand, in some cases precision suffers when such small text is correctly detected as it counts as false positive.

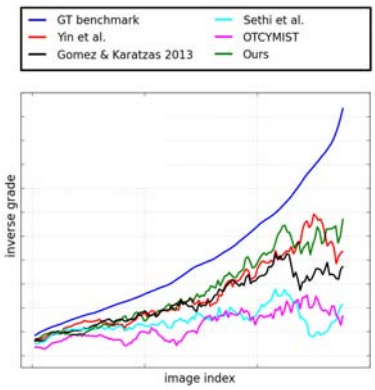


Figure 5.10: Inverse grade curves of different methods in the MSRRC dataset [68].

Figure 5.11: Qualitative segmentation results on the KAIST dataset.

5.2.3 *Scene text localization results*

Table 5.3 compares the final results of our method with state-of-the-art scene text localization methods in five benchmarks (KAIST, MSRA-TD500, MSRRC, ICDAR 2013, and ICDAR2003). A set of example qualitative results are shown in Figures 5.12 and 5.13.

	KAIST			MSRA-TD500			MSRRC			IC13			IC03		
	P	R	f	P	R	f	P	R	f	P	R	f	P	R	f
CASIA NLPR [60]	-	-	-	-	-	-	-	-	-	0.79	0.68	0.73	-	-	-
Chen *et al.* [14]	-	-	-	0.05	0.05	0.05	-	-	-	-	-	-	0.60	0.60	0.58
Epshtein *et al.* [27]	-	-	-	0.25	0.25	0.25	-	-	-	-	-	-	0.73	0.60	0.66
Previous Chapter	-	-	-	0.58	0.54	0.56	-	-	-	-	-	-	0.71	0.57	0.64
Lee *et al.*[71]	0.69	0.60	0.64	-	-	-	-	-	-	-	-	-	-	-	-
Li *et al.*[74]	0.59	0.79	0.67	0.30	0.32	0.31	-	-	-	-	-	-	0.45	**0.80**	0.57
I2R NUS FAR [60] *	-	-	-	-	-	-	-	-	-	0.75	**0.69**	0.72	-	-	-
Text Detection [60]	-	-	-	-	-	-	-	-	-	0.74	0.53	0.62	-	-	-
TextSpotter [60, 98]	-	-	-	-	-	-	-	-	-	**0.88**	0.65	0.74	-	-	-
TD-ICDAR [144]	-	-	-	0.53	0.52	0.54	-	-	-	-	-	-	0.68	0.66	0.66
TD-Mixture [144]	-	-	-	0.63	0.63	0.60	-	-	-	-	-	-	0.69	0.66	0.67
Yin *et al.* [146, 68, 60]	-	-	-	-	-	-	**0.64**	0.42	0.51	**0.88**	0.66	**0.76**	-	-	-
This Chapter	**0.71**	**0.83**	**0.77**	**0.69**	0.54	**0.61**	0.63	**0.54**	**0.58**	0.78	0.67	0.72	**0.74**	0.65	**0.69**
(Horiz. only)	-	-	-	-	-	-	-	-	-	0.80	**0.69**	0.74	**0.75**	0.66	**0.70**

Table 5.3: Scene text localization results (precision, recall, and f-score) in standard datasets.

Results in Table 5.3 demonstrate that the method proposed in this paper outperforms other state-of-the-art methods in KAIST, MSRA-TD500, and MSRRC datasets, while being competitive with the ICDAR2013 robust reading competition results. ICDAR2013 results have a coherent interpretation as we aim for the highest generality of our method, addressing the unconstrained problem of detecting text irrespective of its language, script, and orientation. Contrary to our method, most methods listed in ICDAR columns of Table 5.3 have been trained explicitly for horizontally aligned English text and address only this particular scenario. In this sense, it is important to notice that some of the top scoring methods in the IC03 column have been evaluated in the MSRA-TD500 arbitrary oriented text dataset with a much worse performance compared to the method proposed here.

The average time performance of our method using a standard commodity *i*7 CPU ranges from 0.5 to 3 seconds depending on the input image size. This time stamp could be reduced in a factor of $\times 4$

Figure 5.12: Qualitative localization results on the ICDAR 2013 dataset.

Figure 5.13: Qualitative localization results on the MSRA-TD500 dataset.

if each input color channel is processed independently in parallel.

5.3 *Conclusions*

This Chapter details a novel scene text extraction method in which the exploitation of the hierarchical structure of text plays an integral part. We have seen that the algorithm can efficiently detect text groups with arbitrary orientation in a single clustering process that involves: a learned optimal clustering feature space for text region grouping, novel discriminative and probabilistic stopping rules, and a new set of features for text group classification that can be efficiently calculated in an incremental way.

Experimental results demonstrate that the presented algorithm outperforms other state of the art methods in three multi-script and arbitrary oriented scene text standard datasets while it stays competitive in the more restricted scenario of horizontally-aligned English text ICDAR dataset. Moreover, the presented results in all datasets are obtained with a single (mixed) training set, demonstrating the general purpose character of the method which yields robust performance in a variety of distinctly different scenarios.

Finally, the baseline analysis of the algorithm reveals that overall system recall can be substantially increased if needed by using feature space diversification. Our findings are positioned in line with recent advances in object recognition [129] where bottom-up grouping of an initial segmentation is used to generate object location hypotheses, producing a substantially reduced search space in comparison to the traditional sliding window approaches. An intuition that will be further explored in the next Chapter.

Chapter 6

Text Regions Proposals

Object Proposals is a recent computer vision technique for generation of high quality object locations. The main interest of such methods is their ability to speed up recognition pipelines that make use of complex and expensive classifiers by considering only a few thousands of bounding boxes. It therefore constitutes an alternative to exhaustive search, which has many well known drawbacks, and enables the efficient use of more powerful classifiers in end-to-end pipelines by greatly reducing the search space as shown in Figure 6.1.

On the other hand, in the context of scene text understanding, whole-word recognition methods [41, 2, 51] have demonstrated great success in difficult tasks like word spotting or text based retrieval, however they are usually based in expensive techniques. In this scenario the underlying process is similar to the one in multiclass object recognition. It is therefore suggestive for the use of Object Proposals techniques mimicking the state of the art object recognition pipelines.

Traditionally, high precision specialized detectors have been used for segmentation of text in natural scenes, and afterwards text recognition techniques applied to their output [98, 137, 142, 43]. But it is a well known fact that the perfect text detector, able to work in any conditions, does not exist up to date. In fact, to mitigate the lack of a perfect detector Bissacco *et al.* [7] propose an end-to-end scene text recognition pipeline using a combination of several detection methods running in parallel. Demonstrating that if you have a robust

Figure 6.1: Sliding a window for all possible locations, sizes, and aspect ratios represents a considerable waste of resources. The best ranked 250 proposals generated with our text specific selective search method provide 100% recall and high-quality coverage of words in this particular image.

recognition method at the end of the pipeline the most important thing in earlier stages is to achieve high recall while precision is not so critical.

The dilemma is thus to choose between having a small set of detections with very high precision but most likely losing some of the words in the scene, or a larger set of proposals, usually in the range of few thousands, with better coverage and then let the recognizer to make the final decision. The later seems to be a well-grounded procedure in the case of word-spotting and retrieval for various reasons. First, as said before, we have powerful whole-word recognizers but they are complex and expensive, second, the recall of current text detection methods may limit their accuracy, and third, sliding window can not be considered an efficient option mainly because words do not have a constrained aspect ratio.

In this chapter we explore the applicability of Object Proposals techniques in scene text understanding, aiming to produce a set of word proposals with high recall in an efficient way. We propose a simple text-specific selective search strategy, where initial regions in the image are grouped by agglomerative clustering in a hierarchy where each node defines a possible word hypothesis. Moreover, we evaluate different state of the art Object Proposals methods in their ability of detecting text words in natural scenes. We compare the proposals obtained with well known class-independent methods with our own method, demonstrating that our algorithm is superior in its ability of producing good quality word proposals in an efficient way.

6.1 *Text Specific Selective Search*

Here we make use of the same grouping framework developed in the preceding chapters of this thesis, where a set of complementary grouping cues are used in parallel to generate hierarchies in which each node correspond to a text-group hypotheses. Our algorithm is divided in three main steps: segmentation, creation of hypotheses through bottom-up clustering, and ranking.

6.1.1 *Creation of hypotheses*

The grouping process starts with a set of MSER regions and the SLC algorithm builds a hierarchy of grouping proposals using a certain distance metric $d(r_a, r_b)$. Similarly to [128] we assume that there is no single grouping strategy that is guaranteed to work well in all cases. Thus, our basic agglomerative process is extended with several diversification strategies in order to ensure the detection of the highest number of text regions in any case. First, we extract regions separately from different color channels (i.e. Red, Green, Blue, and Gray) and spatial pyramid levels. Second, on each of the obtained segmentations we apply SLC using different complementary distance

metrics:

$$d^{(i)}(r_a, r_b) = (f^i(r_a) - f^i(r_b))^2 + (x_a - x_b)^2 + (y_a - y_b)^2 \qquad (6.1)$$

where the (x_a, y_a) and (x_b, y_b) are the centers of regions r_a and r_b, and $f^i(r)$ is a feature aimed to measure the similarity of two regions. Here, as in precvious Chaper we make use of the following features: mean gray value of the region, mean gray value in the immediate outer boundary of the region, region's major axis, mean stroke width, and mean of the gradient magnitude at the region's border.

6.1.2 *Ranking*

Once we have created our similarity hierarchies each one providing a set of text group hypotheses, we need an efficient way to sort them in order to provide a ranked list of proposals prioritizing the best hypotheses. In the experimental section we explore the use of the following rankings:

Pseudo-random ranking

We make use of the same ranking strategy proposed by Uijlings *et al.*in [128]. Particularly, each hypothesis is assigned with an increasing integer value, starting from 1 for the root node of a hierarchy and subsequently incrementing for the rest of the nodes up to the leaves of the tree. Then each of this values is multiplied with a random number between zero and one, thus providing a ranking that is randomly produced but prioritizes larger regions. As in [128] the ranking process is performed before removing duplicate hypotheses. This way if a particular grouping has been found several times within the different hierarchies, indicating a more consistent hypothesis under different similarity cues, this group is going to have more probabilities to be ranked in the top of the list.

Cluster meaningfulness ranking

Instead of assigning an increasing value prioritizing larger groups, we propose here to use the cluster quality measure that we have detailed in section 5.1.2 of previous Chapter. Intuitively this value is going to very small for groups comprising a set of very similar regions, that are densely concentrated in small volumes of the feature space. This measure is thus well indicated in the case of measuring text-likeliness of groups because such a strong similarity property is expected to be found in text groups. However, the ranking provided by calculating the $\mathcal{NFA}$ (equation 5.6) of each node in our hierarchies is going to prioritize large text groups, e.g. paragraphs, rather that individual words, and thus we multiply the ranking provided by equation 5.6 with a random number between zero and one as done before, providing a pseudo-random ranking where more meaningful hypothesis are prioritized.

Text classifier confidence

Finally, we propose the use of a weak classifier to ranking text grouping candidates. The basic idea here is to train a classifier to discriminate between text and non-text hypotheses and to produce a confidence value that can be used to rank group hypotheses. Since the classifier is going to be evaluated on every node of our hierarchies, we aim to use a fast classifier and features with low computational cost. We train a Real AdaBoost classifier with decision stumps using as features the coefficients of variation of the individual region features f^i described in section 6.1.1: $F^i(G) = \sigma^i / \mu^i$, where μ^i and σ^i are respectively the mean and standard deviation of the region features f^i in a particular group G, $\{f^i(r) : r \in G\}$. Intuitively the value of F^i is smaller for text hypotheses than for non-text groups, and thus the classifier would be able to generate a ranking prioritizing the best hypotheses. Notice that all F^i group features can be computed efficiently in an incremental way along the SLC hierarchies, and that all f^i region features have been previously computed.

6.2 *Experiments and Results*

In our experiments we make use of two standard scene text datasets: the ICDAR Robust Reading Competition dataset (ICDAR2013) [60] and the Street View Dataset (SVT) [136]. In both cases we provide results for their test sets, consisting in 233 and 249 images respectively, using the original word level ground-truth annotations.

The evaluation framework used is the standard for Object Proposals methods [49] and is based on the analysis of the detection recall achieved by a given method under certain conditions. Recall is calculated as the ratio of GT bounding boxes that have been predicted among the object proposals with an intersection over union (IoU) larger than a given threshold. This way, we evaluate the recall as a function of the number of proposals, and the quality of the first ranked N proposals by calculating their recall at different IoU thresholds.

6.2.1 *Evaluation of diversification strategies*

First, we analyse the performance of different variants of our method by evaluating the combination of diversification strategies presented in Section 6.1. Table 6.1 shows the average number of proposals per image, recall rates, and time performance obtained with some of the possible combinations. We select two of them, that we will call "FAST" and "FULL" as they represent a trade-off between recall and time complexity, for further evaluation.

6.2.2 *Evaluation of proposals' rankings*

Figure 6.2 shows the performance of our "FAST" pipeline at 0.5 IoU using the various ranking strategies discussed in Section 6.1. The

Method	# prop.	0.5 IoU	0.7 IoU	0.9 IoU	time(s)
I+D	536	0.84	0.65	0.41	0.26
I+DF	993	0.91	0.78	0.53	0.29
I+DFBGS	1323	0.95	0.86	0.60	0.45
RGB+DF	3359	0.96	0.91	0.69	0.73
RGBI+DFBGS	5659	0.98	0.94	0.75	1.72
P2+RGBI+DFBGS	8164	0.98	0.96	0.79	2.18

Table 6.1: Max recall at different IoU thresholds and running time comparison of different diversification strategies in the ICDAR2013 dataset. We indicate the use of individual color channels: (R), (G), (B), and (I); spatial pyramid levels: (P2); and similarity cues: (D) Diameter, (F) Foreground intensity, (B) Background intensity, (G) Gradient, and (S) Stroke width.

area under the curve (AUC) is 0.39 for NFA, 0.43 both for PR and PR-NFA rankings, while a slightly better 0.46 for the ranking provided by the weak classifier. Since the overhead of using the classifier is negligible we use this ranking strategy for the rest of the experiments.

6.2.3 *Comparison with state of the art*

In the following we further evaluate the performance of our method in the ICDAR2013 and SVT datasets, and compare it with the following state of the art Object Proposals methods: BING [15], EdgeBoxes [151], Randomized Prim's [79] (RP), and Geodesic Object Proposals [65] (GOP).

In our experiments we use publicly available code of these methods with the following setup. For BING we use the default parameters: base of 2 for the window size quantization, feature window size of 8×8, and non maximal suppression (NMS) size of 2. For EdgeBoxes we also use the default parameters: step size of the sliding window of 0.65, and NMS threshold of 0.75; but we change the max number of boxes to 10^6. GOP is configured with Multi-Scale Structured Forests for the segmentation, 150 seeds heuristically placed, and 8 segmentations per seed; in this case we tried other configurations in order to increase the number and quality of the proposals without success. For RP we use the default configuration with 4 color spaces (HSV,Lab,Opponent,RG) because it provided much bet-

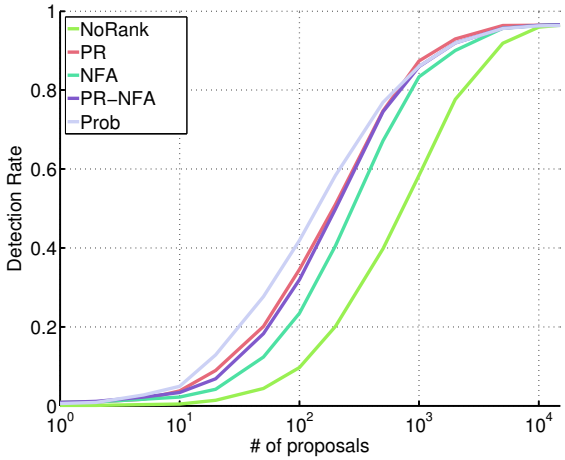


Figure 6.2: Performance of our "FAST" pipeline at 0.5 IoU using different ranking strategies: (PR) Pseudo-random ranking, (NFA) Meaningfulness ranking, (PR-NFA) Randomized NFA ranking, (Prob) the ranking provided by the weak classifier.

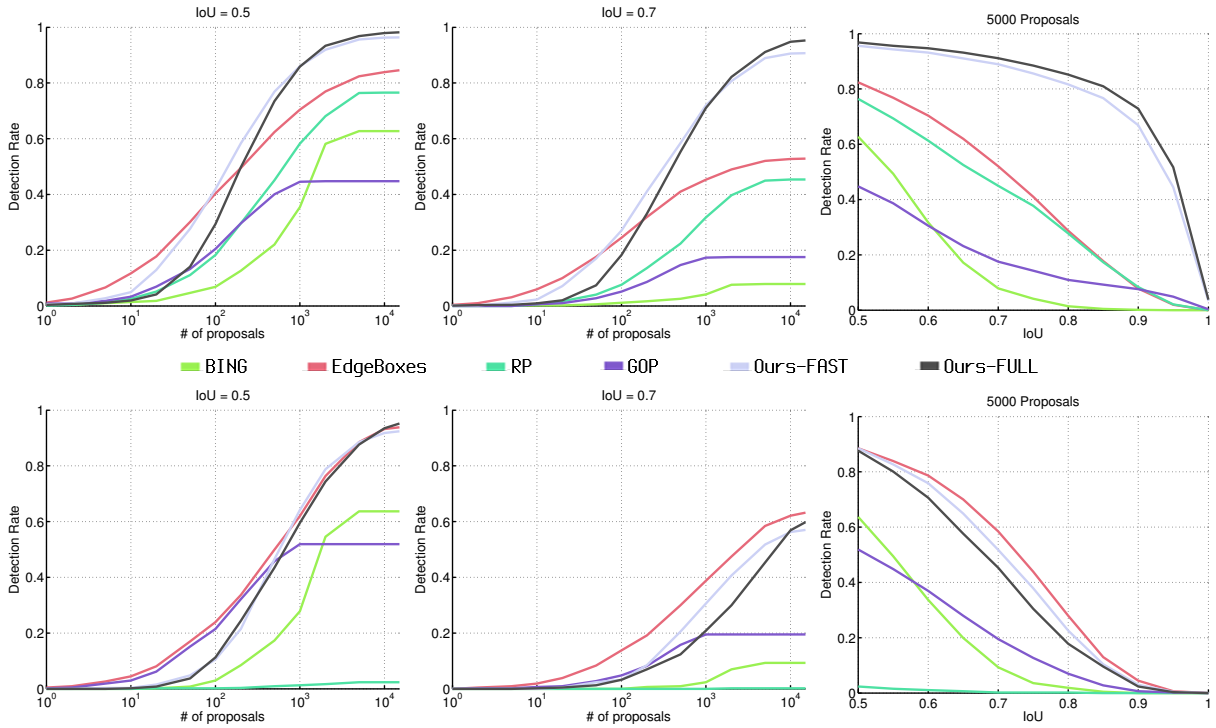


Figure 6.3: A comparison of various state-of-the-art object proposals methods in the ICDAR2013 (top) and SVT (bottom) datasets. (left and center) Detection rate versus number of proposals for various intersection over union thresholds. (right) Detection rate versus intersection over union threshold for various fixed numbers of proposals.

ter results than sampling from a single graph, while being 4 times slower.

Tables 6.2 and 6.3 show the performance comparison of all the evaluated methods in ICDAR2013 and SVT datasets respectively. A more detailed comparison is provided in Figure 6.3. All time measurements in Tables 6.2 and 6.3 have been calculated by executing code in a single thread on the same i7 CPU for fair comparison, while most of them allow parallelization. For instance the multi-threaded version of our method is able to achieve execution times of 0.31 and 0.71 seconds respectively for the "FAST" and "FULL" variants in the ICDAR2013 dataset.

Method	# prop.	0.5 IoU	0.7 IoU	0.9 IoU	time(s)
BING [15]	2716	0.63	0.08	0.00	1.21
EdgeBoxes [151]	9554	0.85	0.53	0.08	2.24
RP [79]	3393	0.77	0.45	0.08	12.80
GOP [65]	855	0.45	0.18	0.08	4.76
Ours-FAST	3359	**0.96**	**0.91**	**0.69**	**0.79**
Ours-FULL	8164	**0.98**	**0.96**	**0.79**	2.25

Table 6.2: Average number of proposals, recall at different IoU thresholds, and running time comparison with Object Proposals state of the art algorithms in the ICDAR2013 dataset.

As can be seen in Table 6.2 and Figure 6.3 our method outperforms all the evaluated algorithms in terms of detection recall on the ICDAR2013 dataset. Moreover, it is important to notice that detection rates of all the generic Object Proposals heavily deteriorate for large IoU thresholds while our text specific method provides much more stable rates indicating a better coverage of text objects, see the

high AUC difference in Figure 6.3 bottom plots.

Method	# prop.	0.5 IoU	0.7 IoU	0.9 IoU	time(s)
BING [15]	2987	0.64	0.09	0.00	0.81
EdgeBoxes [151]	15319	0.94	**0.63**	0.04	2.71
RP [79]	5620	0.02	0.00	0.00	10.51
GOP [65]	778	0.53	0.19	0.03	4.31
Ours-FAST	3791	0.90	0.46	0.03	**0.66**
Ours-FULL	10365	**0.95**	0.61	**0.06**	2.22

Table 6.3: Average number of proposals, recall at different IoU thresholds, and running time comparison with Object Proposals state of the art algorithms in the SVT dataset.

The results on the SVT dataset in Table6.3 and Figure 6.3 exhibit a radically distinct scenario. While our "FULL" pipeline is slightly better than EdgeBoxes at 0.5 IoU, the later is able to outperform both of our pipelines at 0.7 and our "FAST" variant at 0.5. Moreover, in this dataset our method does not provide the same stability properties shown before. This can be explained because both datasets are very different in nature, SVT contains more challenging text, with lower quality and many times under bad illumination conditions, while in ICDAR2013 text is mostly well focussed and flatly illuminated. Still, the AUC in most of the plots in Figure 6.3 show a fairly competitive performance for our method.

6.3 *Conclusion*

In this chapter we have evaluated the performance of generic Object Proposals algorithms in the task of detecting text words in natural scenes. We have presented a text specific method that is able to outperform generic methods in many cases, or to show competitive numbers in others. Moreover, the proposed algorithm is parameter free and fits well the multi-script and arbitrary oriented text scenario.

An interesting observation of our experiments is that while in class-independent object detection generic methods suffice with near a thousand proposals to achieve high detection recall, in the case of text we still need around 10000 in order achieve similar rates, indicating there is a large room for improvement in specific text Object Proposals methods.

Chapter 7

Efficient tracking of text groupings

Text detection in video sequences differs from still images in many aspects and it is not a straightforward assumption that a method devised and trained on static text images would be applicable to video sequences. A key characteristic of video sequences is the temporal redundancy of text, which calls for tracking based processes taking advantage of past history to increase the stability and quality of detections.

Keeping constant track of a text object throughout all the frames where it is visible is desirable for example to ensure a unique response of the system (e.g. translation, or text to speech conversion) for each distinct text, and also to be able to enhance the text regions [72], or to select the best frames in which they appear, before doing the full text recognition. Moreover, one can take advantage of the tracking process in order to obtain a real-time detection system, under the assumption that the scene does not change much from frame to frame. This yields an extra speed-up that can be exploited in see-though applications (e.g. Augmented Reality translation [35, 109] and augmented documents [126]) or street-view navigation [87].

In this Chapter we develop a real-time scene text detection algorithm that combines the text detection method presented in Chapter 5 with an MSER-based tracking module [24]. As both detection and tracking modules are based on MSER regions they can be integrated symbiotically, improving robustness and providing a speed boost to the system. Real-time text detection is simulated by propagating in time the previously detected text-regions, until a new text detection takes place.

The novelty of the proposed method lies in the ability to effectively track text regions' groupings (establishing a pixel level segmentation of constituent text parts in every frame), not merely their bounding boxes as usually done in state-of-the-art tracking-by-detection algorithms [140], while providing a considerable speed-up in comparison to performing a full frame text detection on each frame. Moreover, the proposed method can deal with rotation, translation, scaling, and perspective deformations of detected text.

7.1 Text Detection and Tracking Method

In this Section we proceed to explain the proposed text tracking algorithm, and the way it is integrated with our text detection method. Figure 7.1 shows a timeline representation of the interations between detection and tracking modules. The main idea behind our proposal is that even with a slow text detection method it is possible to achieve real-time performance by running it periodically while in parallel a fast tracking module takes care of propagating previous detections for the frames that are not processed by the detection module. The overall speed of the process really only depends on the tracking module, therefore the system can be real-time. However, the speed of the text detector is nevertheless important, as it is the only source of new information to the system, and long times between consecutive detections could deteriorate substantially the overall performance.

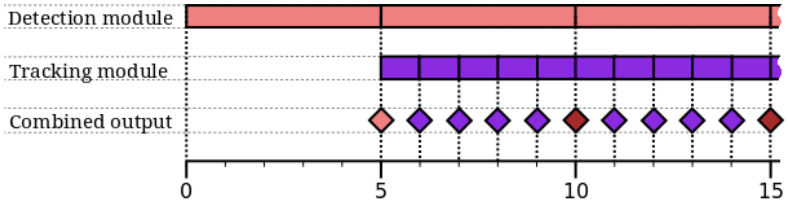


Figure 7.1: Timeline view of the text detection and tracking modules' combination. Rectangles represent lasting processes in time and diamonds are the output of the system. The detection module creates the first estimates of text object positions, which are propagated to subsequent frames by the tracking module. Each time the detection module provides new results, a merging mechanism combines the detected and tracked objects in a unique output (maroon diamonds).

7.1.1 Tracking Module

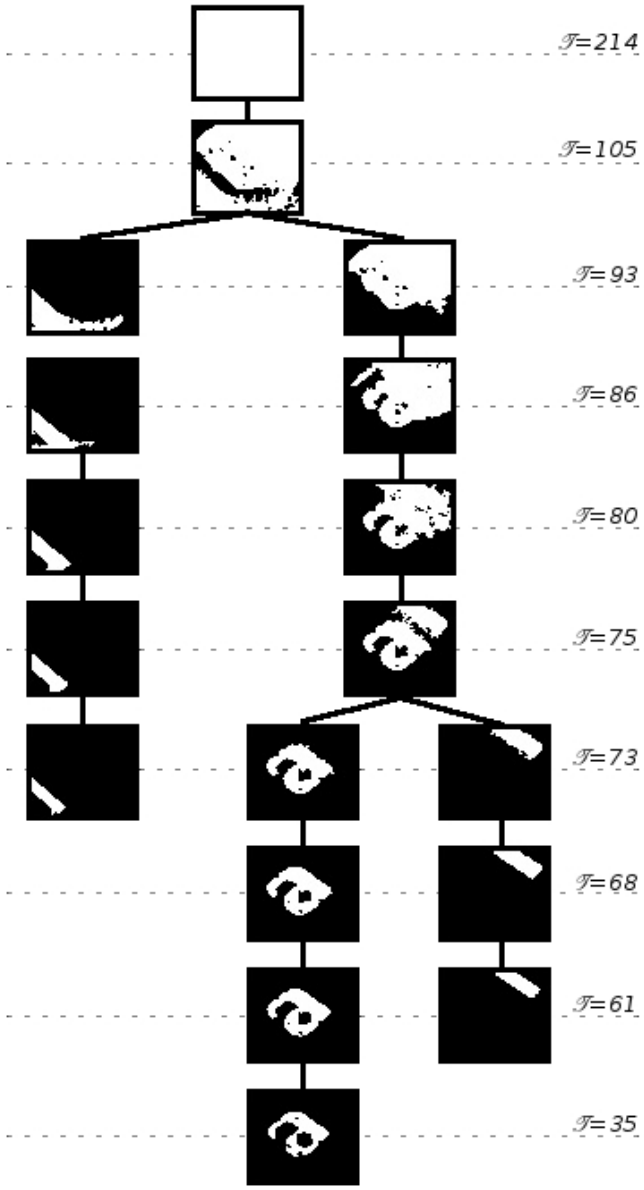
Our tracking module is built upon the framework proposed by Donoser and Bischof in [24] where tracking of single MSERs in successive frames is posed as a correspondence problem within a window surrounding their previous location. The component tree of this small windows can be used as an efficient data structure to solve the correspondence problem: contains all the information needed to search the best matching region, and obviously can be computed much faster than for the whole image.

Figure 7.2 shows how finding the corresponding MSER between two consecutive frames can be done efficiently by constraining the search in two ways: searching only in the component tree of a small window (Figure 7.2b), and looking only in a sub-set of the tree levels (Figure 7.2c). These two constraints define two parameters for the tracking method: the size of the window (with respect to the query region size), and the levels interval to look for (with respect to the level of the query region).

Notice that the search process is done among all the regions in the component tree and would be able to find correct matches even when the target region has lost the stability criteria (e.g. appears blurred) in the consecutive frame.

MSER-tracking has been used effectively for license plate [23] and hand [25] tracking using a weighted vector of simple features: mean gray value, region size, center of mass, width and height of the

(a) (b)



(c)

Figure 7.2: Tracking of text grouping elements posed as search in the component tree of a small region of interest. Given a text regions' grouping detection (a) in frame t_0, the location of its constituent elements in the next frame t_1 is constrained to a small regions of interest around their initial locations at t_0 (b). The structure of the component tree of this small region (c) can be used to search for the best matching component, and again this search can be constrained, e.g. by looking only at certain levels of the tree that correspond to components with an intensity (and/or size) similar to the query.

bounding box, and stability. A further extension in [26] makes use of novel shape descriptors in order to increment the robustness of the tracking by considering shape similarity.

The text tracking module proposed here differs from the work of Donoser and Bischof in two aspects by considering the specificities of text regions' groupings: 1) We use invariant moments as features to find correspondences, as a tradeoff between the fast computation of the simple features used in [24] and the robustness of the shape descriptors in [26]. 2) Here we consider groups of regions (text lines) instead of a single MSER regions as done in [23] and [25], and thus we can detect mismatches using RANSAC when they do not fit an underlying line model.

MSER-tracking with incrementally computable invariant moments

The inclusion relation between regions in the component tree can be exploited to extract incrementally computable descriptors without any extra computational cost as proposed in [81] [98]. Geometric moments [50] [33] can be calculated in this way and thus result in a invariant descriptor that can be used efficiently by the MSER-tracking algorithm for matching. At each grow step of the MSER algorithm the raw moments up to order three are updated with constant complexity. Then, when required in order to find correspondences, those raw moments can derive the seven Hu's moments [50] and four Affine Invariant Moments [33] again with constant complexity.

Invariant moments have generally robust performance for rigid objects with simple contour shapes and simple transformations such as scaling, rotation, and affine transformations [149]. This is the case for text characters [34], assuming that they do not change much in shape between successive frames. We consider this compact descriptor to be tradeoff between of the simple feature based analysis in [24] and the integral shape descriptors used in [26], as they provide a richer representation than the former while being much less computationally expensive than the latter.

Moreover, in cases where invariant moments are prone to fail: e.g. for particularly weak shaped characters (e.g. letter "I"), partial occlusions, or motion blur mismatches, we can take advantage from two particularities in our scenario: first we have quite a constrained search along the component tree, and second we can exploit the group-level (text line) coherence in order to detect and reject mismatching correspondences via RANSAC.

Mismatch detection with RANSAC

Fitting a simple linear regression model against individually tracked MSERs using RANSAC allows to improve the whole text-line tracking because false correspondences do not affect the tracking process as they are eliminated by the RANSAC algorithm consensus set and thus not propagated to subsequent frames. This way the tracking

module is able to robustly track text lines even when some of their characters are not correctly tracked. This outlier detection makes the method more robust in case of partial occlusion of the text line tracked.

Notice that RANSAC here is used as an outlier detection mechanism and not as an homography estimator as done in other tracking algorithms. Regions not correctly tracked are detected as outliers in a simple linear regression model and removed from the tracking system for the following frames. This process allows also to naturally stop the tracking process when the number of inliers for a text-line in a given frame is less than 3 regions.

7.1.2 *Merging detected regions and propagated regions*

Each time the detection module provides new results from a new full detection a merging mechanism, depicted with maroon diamonds in Figure 7.1, is needed in order to identify if the newly arrived detections are the same we are already tracking or not. This matching is done with the Hungarian algorithm by optimizing the one-to-one overlapping of the min. enclosing boxes of detected and tracked text lines.

Matched text lines are updated with the newly detected MSERs, thus regenerating the tracking process with new evidences, and may also recuperate regions that have been lost during the tracking process (i.e. detected as outliers and removed from the system). Not matched text-lines are treated as "first time viewed" objects and start their own tracking process from their initial locations.

7.2 *Experiments*

We have evaluated our algorithm for the task of text detection and tracking in a dataset of synthetically generated video sequences. The dataset contains 10 sequences of 400 frames, with a resolution of 640×480 pixels, where still images from the ICDAR [60] and MSRRC [68] datasets are deformed iteratively with random rotation, translation, scale changes, and perspective transformations. Figure 7.4 shows two example sequences from the generated synthetic dataset. The main reason for the use of synthetic data in this series of experiments is that ground truth data can be created automatically, without any labelling effort.

Figure 7.3 shows the CLEAR-MOT metrics [62] performance comparison of the proposed method against two other approaches: Performing the full-detection on every frame, and MSER-tracking with simple features as in [24]. We can see how the proposed method outperforms the others both in tracking precision (MOTP) and accuracy (MOTA). MOTP is basically an average measure of the overlapping of correct detections and ground-truth text lines over the whole video sequence. We can see how this value is lower for the MSER-tracking with simpler features, indicating that the simple features

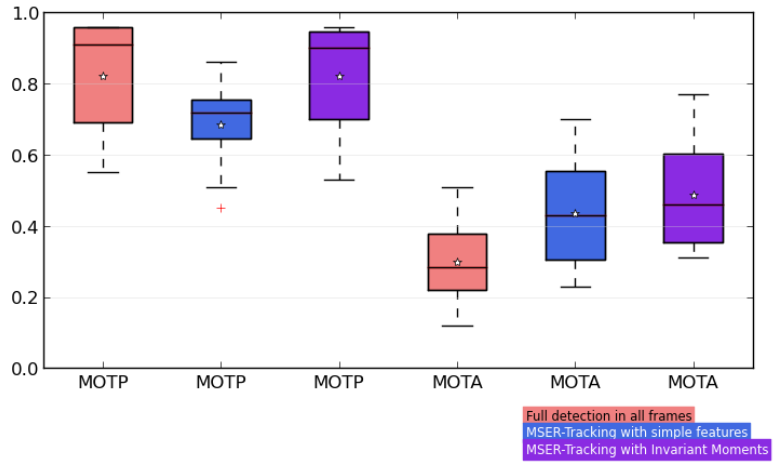


Figure 7.3: Text Detection and Tracking performance comparison in the synthetic video dataset.

are not enough to correctly track individual regions and thus produce less accurate bounding boxes at the text line level. The MOTA metric accounts for all errors made by the tracker: false positives, misses, and mismatches, over all the frames in a video sequence. The lower accuracy for the full-detection approach is due to missing objects, that the MSER-tracking is able to compensate by correctly propagating the detections of previous frames. In the MSER-tracking approaches the main source of MOTA errors are false positive detections provided by the detection module, which are propagated in time. On the other hand, the difference in accuracy between the two MSER-tracking methods indicate that invariant moments are more robust than the simpler feature vector.

Figure 7.4: Still frame results from two of the synthetic image sequences (top) and two of the real scene image sequences (bottom) used for performance evaluation.

We further evaluated our method qualitatively in two sets of real scene image sequences: a set of 4 videos provided in [82], and a set of 10 videos obtained by the authors with a mobile device. The obtained results (see Figure 7.4) show that in general the system is able to detect and track the targeted text correctly dealing with rotation, translation, scaling, and perspective deformations. There are however errors of missing text components in the presence of motion blur and strong illumination changes. Nevertheless it is worth noting that in such situations the tracking module is still able to propagate some regions that would otherwise result in missed text by the detection module.

7.2.1 *Time performance*

In order to obtain time performance measurements and qualitative results for the task of text detection and tracking, we have implemented the proposed method using the Android development framework provided by the OpenCV [1] library, and tested it in a tablet computer with a 1.5GHz quad-core processor.

[1] http://opencv.org/platforms/android.html

Table 7.1 shows average time performance measurements for each of the method modules. The average frame rate of the system is 15 fps., a value just slightly lower than the achievable from the Android camera service (without any image processing) for the concrete device used in the evaluation. Such a fast performance is achieved thanks to the negligible timestamp of the tracking module (40 ms. in average). On the other hand, the detection module running in the background needs about 1 second in average to provide localization results, this is roughly double the time to do the same processing in the main thread. Such a relatively low performance affects the system with a noticeable delay when new targets appear into the scene and to recover a missed target.

	Android device		Standard PC	
	avg. time	fps.	avg. time	fps.
Text detection module (async.)	1041.01 ms.	n.a.	53.45 ms.	n.a.
Initial merging and tracking	125.57 ms.	7.96	17.11 ms.	58.44
Text tracking module	40.01 ms.	24.99	6.91 ms.	144.71

Table 7.1: Average time performance measurements.

As shown in Table 7.1 the system has a variable frame rate: it is able to achieve a really high average frame rate (near 25 fps.) during the tracking process (this is most of the time), but once the asynchronous detection process finishes (every 1 second in average) the frame rate slows down to 8 fps. but just for one frame, as the new detections must be propagated and matched with the existing ones. All in all, the proposed method demonstrates real-time performance in low-resource devices with an average frame rate of 15 fps.

Time performance measurements for the tracking module would increase linearly with the number of tracked regions and the size of their search windows. In the evaluated sequences the average number of tracked regions is 21, covering around 10% of the input image size, which fits well with a realistic scene text detection scenario.

7.3 *Conclusion*

In this Chapter we have presented a method for detection and tracking of scene text able to work in real-time even on low-resource mobile devices. Although far from being a final solution, the proposed method goes beyond the full-detection approaches in terms of time performance optimization. The combination of text detection with a tracker, provides considerable stability, allowing the system to pro-

vide predicted estimates in cases where the detection module itself is not capable of returning a valid response. The use of MSER-tracking as an alternative, fast technique to provide simulated text detections for the frames that are not processed by the full frame text detector proves to be an adequate solution, providing the system with enough information to continue tracking until the text detector returns updated positions.

The main limitation of the proposed method is the tracking degradation in presence of severe motion blur or strong illumination changes.

As in all tracking systems, the longer the full frame text detector takes to provide a result, the higher the uncertainty of the tracker will grow. At the same time, the response of the full frame text detector will be less reliable as more frames pass since the one being processed. Therefore, it is important that reasonably fast text detection methods are used in such a framework to ensure that tracking does not deteriorate rapidly.

Chapter 8

Scene text script identification

Script and language identification are important steps in modern OCR systems designed for multi-language environments. Since text recognition algorithms are language-dependent, detecting the script and language at hand allows selecting the correct language model to employ [130]. While script identification has been widely studied in document analysis, it remains an almost unexplored problem for scene text. In contrast to document images, scene text presents a set of specific challenges, stemming from the high variability in terms of perspective distortion, physical appearance, variable illumination and typeface design. At the same time, scene text comprises typically a few words, contrary to longer text passages available in document images.

Current end-to-end systems for scene text reading [7, 52, 102] assume single script and language inputs given beforehand, i.e. provided by the user, or inferred from available meta-data. The unconstrained text understanding problem for large collections of images from unknown sources (see Figure 8.1) has not been considered up to very recently [117, 116, 103, 44].

In this Chapter we address the problem of script identification in natural scene images, paving the road towards true multi-lingual end-to-end scene text understanding.

Figure 8.1: Collections of images from unknown sources may contain textual information in different scripts.

Figure 8.2: (best viewed in color) Certain stroke-parts (in green) are discriminative for the identification of a particular script (left), while others (in red) can be trivially ignored because are frequent in other classes (right).

Multi-script text exhibits high intra-class variability (words written in the same script vary a lot) and high inter-class similarity (certain scripts resemble each other). Examining text samples from different scripts, it is clear that some stroke-parts are quite discriminative, whereas others can be trivially ignored as they occur in multiple scripts. The ability to distinguish these relevant stroke-parts can be leveraged for recognizing the corresponding script. Figure 8.2 shows an example of this idea.

The use of state of the art CNN classifiers for script identification is not straightforward, as they fail to address a key characteristic of scene text instances: their extremely variable aspect ratio. As can be seen in Figure 8.3, scene text images may span from single characters to long text sentences, and thus resizing images to a fixed aspect ratio, as in the typical use of holistic CNN classifiers, will deteriorate discriminative parts of the image that are characteristic of its class. The key intuition behind the methodology proposed in this Chapter is that in order to retain the discriminative power of stroke parts we must rely in powerful local feature representations and use them within a patch-based classifier. In other words, while holistic CNNs have superseded patch-based methods for image classification, we claim that patch-based classifiers can still be essential in tasks where image shrinkage is not feasible.

To test our intuition we build a method for script identification that combines convolutional features, extracted by sliding a window with a single layer Convolutional Neural Network (CNN) [18], with the Naive-Bayes Nearest Neighbor (NBNN) classifier [8], obtaining promising results. Afterwards, we demonstrate far superior performance by extending our previous work in two different ways: First, we use deep CNN architectures in order to learn more discriminative representations for the individual image patches; Second, we propose a novel learning methodology to jointly learn the patch representations and their importance (contribution) in a global image to class probabilistic measure. For this, we train our CNN using an Ensemble of Conjoined Networks and a loss function that takes into account the global classification error for a group of N patches instead of looking only into a single image patch. Thus, at training time our network is presented with a group of N patches sharing the same class label and produces a single probability distribution over the classes for all them. This way we model the goal for which the network is trained, not only to learn good local patch representa-

Figure 8.3: Scene text images with the larger/smaller aspect ratio available in three different datasets: MLe2e(left), SIW-13(center), and CVSI(right).

tions, but also to learn their relative importance in the global image classification task.

Experiments performed over three public datasets for scene text classification demonstrate state-of-the-art results. In particular we are able to reduce classification error by 5 percentage points in the SIW-13 dataset.

8.1 *Patch-based script identification*

Our method for script identification in scene images follows a multi-stage approach. Given a text line provided by a text detection algorithm, our script identification method proceeds as follows: First we resize the input image to a fixed height of 64 pixels, but maintaining its original aspect ratio in order to preserve the appearance of stroke-parts. Second we densely extract 32×32 image patches with sliding window. And third, each image patch is fed into a single layer Convolutional Neural Network to obtain its feature representation. These steps are illustrated in Figure 8.4 which shows an end-to-end system pipeline incorporating our method (the script-agnostic text detection module is abstracted in a single step as the focus of this Chapter is on the script identification part).

This way, each input region is represented by a variable number of descriptors (one for each image patch), the number of which depends on the length of the input region. Thus, a given text line representation can be seen as a bag of image patch descriptors. However, in our method we do not make use of the Bag of visual Words model, as the quantization process severely degrades informative (rare) descriptors [8]. Instead we directly classify the text lines using the Naive Bayes Nearest Neighbor classifier.

8.1.1 *Patch representation with Convolutional Features*

Convolutional Features provide the expressive representations of image patches needed in our method. We make use of a single layer Convolutional Neural Network [18] which provides us with highly

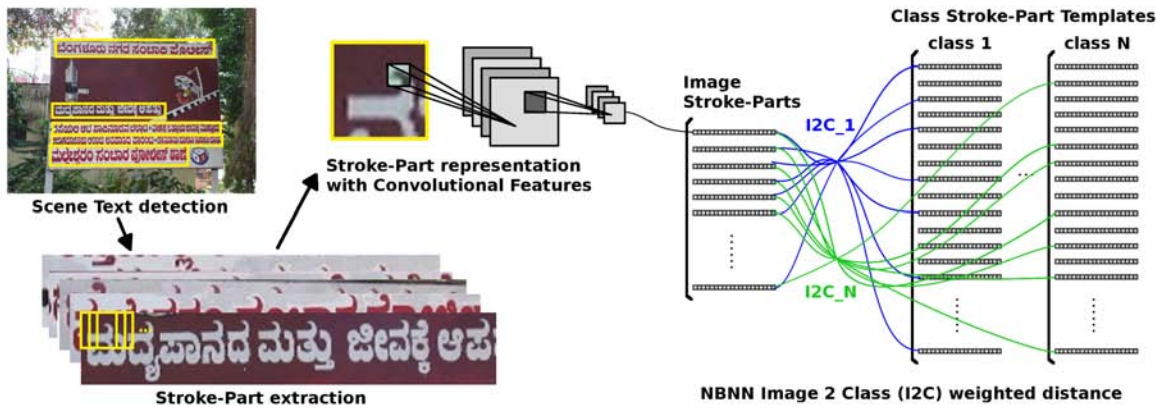


Figure 8.4: Method deploy pipeline: Text lines provided by a text detection algorithm are resized to a fixed height, image patches are extracted with a sliding window and fed into a single layer Convolutional Neural Network (CNN). This way, each text line is represented by a variable number of patch descriptors, that are used to calculate image to class (I2C) distances and classify the input text line using the Naive Bayes Nearest Neighbor (NBNN) classifier.

Figure 8.5: Convolution kernels of our single layer network learned with k-means.

discriminative descriptors while not requiring the large amount of training resources typically needed by deeper networks. The weights of the convolutional layer can be efficiently learned using the K-means algorithm [18].

We adopt a similar design for our network as the one presented in [17]. We set the number of convolutional kernels to 256, the receptive field size to 8×8, and we adopt the same non-linear activation function as in [17]. After the convolutional layer we stack a spatial average pooling layer to reduce the dimensionality of our representation to 2304 ($3 \times 3 \times 256$). The number of convolutional kernels and kernel sizes of the convolution and pooling layers have been set experimentally, by cross-validation through a number of typical possible values for single-layer networks.

To train the network we first resize all train images to a fixed height, while retaining the original aspect ratio. Then we extract random patches with size equal to the receptive field size, and perform contrast normalization and ZCA whitening [63]. Finally we apply the K-means algorithm to the pre-processed patches in order to learn the $K = 256$ convolutional kernels of the CNN. Figure 8.5 depicts a subset of the learned convolutional kernels where it can be appreciated their resemblance to small elementary stroke-parts.

Once the network is trained, we can use it to extract convolutional feature representations of 32×32 image patches, after contrast normalization and ZCA whitening.

A key difference of our work with [17], and in general with the typical use of CNN feature representations, is that we do not aim at representing the whole input image with a single feature vector, but instead we extract a set of convolutional features from small parts in a dense fashion. The number of features per image vary according to its aspect ratio. Notice that the typical use of a CNN, resizing the input images to a fixed aspect ratio, is not appealing in our case because it may produce a significant distortion of the discriminative parts of the image that are characteristic of its class.

8.1.2 *Naive-Bayes Nearest Neighbor*

The Naive-Bayes Nearest Neighbor (NBNN) classifier [8] is a natural choice in our pipeline because it computes direct Image-to-Class (I2C) distances without any intermediate descriptor quantization. Thus, there is no loss in the discriminative power of the image patches representations. Moreover, having classes with large diversity encourages the use of I2C distances instead of measuring

Image-to-Image similarities.

All the feature representations of image patches extracted from the training set images provide the templates that populate the NBNN search space.

In NBNN the I2C distance $d_{I2C}(I, C)$ is computed as:

$$d_{I2C}(I, C) = \sum_{i=1}^{n} \|d_i - NN_C(d_i)\|^2 \tag{8.1}$$

where d_i is the i-th descriptor of the query image I, and $NN_C(d_i)$ is the Nearest Neighbor of d_i in class C. Then the NBNN classifies the query image to the class $\hat{C}$ with lower I2C distance, i.e. $\hat{C} = argmin_C \; d_{I2C}(I, C)$. Figure 8.4 shows how computation of I2C distances in our pipeline reduces to $N \times n$ Nearest Neighbor searches, where N is the number of classes and n is the number of descriptors in the query image. To efficiently search for the $NN_C(d_i)$ we make use of the Fast Approximate Nearest Neighbor kd-tree algorithm described in [92].

8.1.3 *Weighting per class image patch templates by their importance*

When measuring the I2C distance $d_{I2C}(I, C)$ it is possible to use a weighted distance function which weights each template in the train dataset accounting for its discriminative power. The weighted I2C is then computed as:

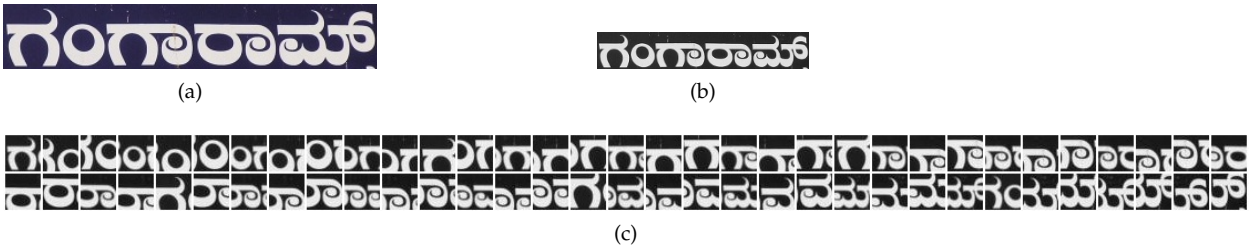
$$d_{I2C}(I, C, w) = \sum_{i=1}^{n} (1 - w_{NN_C(d_i)}) \|d_i - NN_C(d_i)\|^2 \tag{8.2}$$

where $w_{NN_C(d_i)}$ is the weight of the Nearest Neighbor of d_i of class C. The weight assigned to each template reflects the ability to discriminate against the class that the template can discriminate best.

We learn the weights associated to each template as follows. First, for each template we search for the maximum distance to any of its Nearest Neighbors in all classes except their own class, then we normalize these values in the range $[0, 1]$ dividing by the largest distance encountered over all templates. This way, templates that are important in discriminating one class against, at least, one other class have lower contribution to the I2C distance when they are matched as NN_C of one of the query image's parts.

8.2 *Ensembles of conjoined deep networks*

In this section we further develop our patch-based method by extending it two different ways: we use deep CNN architectures for the individual local descriptors; and propose a novel learning methodology to jointly learn the patch representations and their importance (contribution) in a global image to class probabilistic measure.



(a) (b) (c)

Figure 8.6: The original scene text images (a) are converted to greyscale and resized to a fixed height (b) in order to extract small local patches with a dense sampling strategy (c).

8.2.1 *Convolutional Neural Network for stroke-parts classification*

Given an input scene text image (i.e. a pre-segmented word or text line) we first resize it to a fixed height of 40 pixels, but retaining its original aspect ratio. Then, we densely extract patches at two different scales, 32×32 and 40×40, by sliding a window with a step of 8 pixels. The choice of these two particular scales is justified as follows: the 40×40 patch, covering the full height of the resized image, is a natural choice in our system because it provides the largest squared region we can crop; the 32×32 patches are conceived for better scale invariance of the CNN model, similarly as the random crops typically used for data augmentation in CNN-based image classification [67]. Figure 8.6 shows the patches extracted from a given example image. This way we build a large dataset of image patches that take the same label as the image they were extracted from. With this dataset of patches we train a CNN classifier for the task of individual image patch classification.

We use a Deep Convolutional Neural Network to build the expressive image patch representations needed in our method. For the design of our network we start from the CNN architecture proposed in [117] as it is known to work well for script identification. We then iteratively do random search to optimize the following CNN hyper-parameters: number of convolutional and fully connected layers, number of filters per layer, kernel sizes, and feature map normalization schemes. The CNN architecture providing better performance in our experiments is shown in Figure 8.7. Our CNN consists in three convolutional+pooling stages followed by an extra convolution and three fully connected layers. Details about the specific configuration and parameters are given in section 8.2.3.

At testing time, given a query scene text image the trained CNN model is applied to image patches following the same sampling strategy described before. Then, the individual CNN responses for each

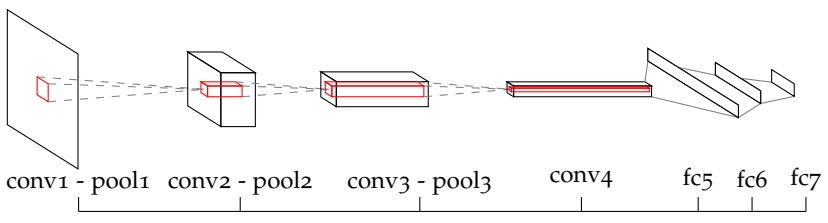


Figure 8.7: Network architecture of the CNN trained to classify individual image patches. The network has three convolutional+pooling stages followed by an extra convolution and three fully connected layers.

image patch can be fed into the global classification rule in order to make a single labeling decision for the query image.

8.2.2 *Training with an Ensemble of Conjoined Networks*

Since the output of the CNN for an individual image patch is a probability distribution over class labels, a simple global decision rule would be just to average the responses of the CNN for all patches in a given query image:

$$y^{(I)} = \frac{1}{n_I} \sum_{i=1}^{n_I} CNN(x_i) \tag{8.3}$$

where an image I takes the label with more probability in the averaged softmax responses ($y^{(I)}$) of their n_I individual patches $\{x_1, ..., x_{n_I}\}$ outputs on the CNN.

The problem with this global classification rule is that the CNN weights have been trained to solve a problem (individual patch classification) that is different from the final goal (i.e. classifying the whole query image). Besides, it is based in a simplistic voting strategy for which all patches are assumed to weight equally, i.e. no patches are more or less discriminative than others. To overcome this we propose the use of an Ensemble of Conjoined Nets in order to train the CNN for a task that resembles more the final classification goal.

An Ensemble of Conjoined Nets (ECN), depicted in Figure 8.8, consists in a set of identical networks that are joined at their outputs in order to provide a unique classification response. At training time the ECN is presented with a set of N image patches extracted from the same image, thus sharing the same label, and produces a single output for all them. Thus, to train an ECN we must build a new training dataset where each sample consists in a set of N patches with the same label (extracted from the same image).

ECNs take inspiration from Siamese Networks [10] but, instead of trying to learn a metric space with a distance-based loss function, the individual networks in the ECN are joined at their last fully connected layer (fc7 in our case), which has the same number of neurons as the number of classes, with a simple element-wise sum operation and thus we can use the standard cross-entropy classification loss.

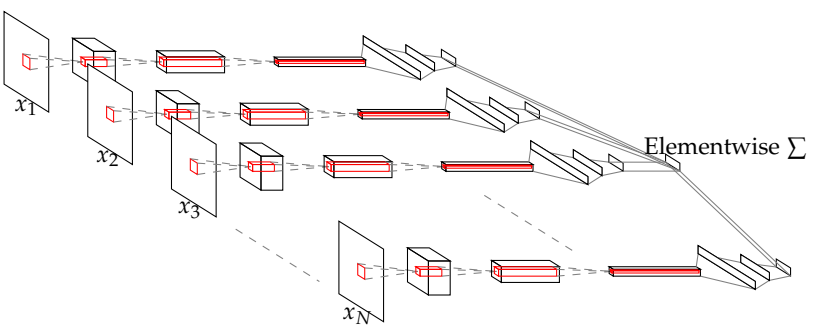


Figure 8.8: An Ensemble of Conjoined Nets consist in a set of identical networks that are joined at their outputs in order to provide a unique classification response.

This way, the cross-entropy classification loss function of the ECN can be written in terms of the N individual patch responses as follows:

$$E = \frac{-1}{M}\sum_{m=1}^{M}\log(\hat{p}_{m,l_m}),$$
$$\hat{p}_{m,k} = \exp(\sum_{n=1}^{N} x_{mnk}) / \left[\sum_{k'=1}^{K}\exp(\sum_{n=1}^{N} x_{mnk'})\right] \tag{8.4}$$

where M is the number of input samples in a mini-batch, $\hat{p}_m$ is the probability distribution over classes provided by the softmax function, l_m is the label of the m'th sample, N is the number of conjoined networks in the ensemble, K is the number of classes, and $x_{mnk} \in [-\infty, +\infty]$ indicates the response (score) of the k'th neuron in the n'th network for the m'th sample.

As can be appreciated in equation 8.4, in an ECN network a single input patch contributes to the backpropagation error in terms of a global goal function for which it is not the only patch responsible. For example, even when a single patch is correctly scored in the last fully connected layer it may be penalized, and induced to produce a larger activation, if the other patches in its same sample contribute to a wrong classification at the ensemble output.

At test time, the CNN model trained in this way is applied to all image patches in the query image and the global classification rule is defined as:

$$y^{(I)} = \sum_{i=1}^{n_I} CNN_{fc7}(x_i) \tag{8.5}$$

where an image I takes the label with the highest score in the sum ($y^{(I)}$) of the fc7 layer responses of the n_I individual patches $\{x_1, ..., x_{n_I}\}$. This is the same as in Equation 8.3 but using the fc7 layer responses instead of the output softmax responses of the CNN.

Notice that still the task for which the ECN network has been trained is not exactly the same defined by this global classification rule, as the number of patches n_I is variable for each image and usually different than the number of conjoined networks N. However, it certainly resembles more the true final classification goal. The number of conjoined networks N is an hyper-parameter of the method that is largely dependent on the task to be solved and is discussed in the experimental section.

8.2.3 *Implementation details*

In this section we detail the architectures of the network models used in this paper, as well as the different hyper-parameter setups that can be used to reproduce the results provided in following sections. In all our experiments we have used the open source Caffe [54] framework for deep learning running on commodity GPUs.

The basic CNN model for individual image patch classification described in section 8.2.1 and Figure 8.7 has the following per layer configuration:

- Input layer: single channel 32×32 image patch.
- conv1 layer: 96 filters with size 5×5. Stride=1, pad=0.
- pool1 layer: kernel size=3, stride=2, pad=1.
- conv2 layer: 256 filters with size 3×3. Stride=1, pad=0.
- pool2 layer: kernel size=3, stride=2, pad=1.
- conv3 layer: 384 filters with size 3×3. Stride=1, pad=0.
- pool3 layer: kernel size=3, stride=2, pad=1.
- conv4 layer: 512 filters with size 1×1. Stride=1, pad=0.
- fc5 layer: 4096 neurons.
- fc6 layer: 1024 neurons.
- fc7 layer: N neurons, where N is the number of classes.
- SoftMax layer: Output a probability distribution over class labels.

The total number of parameters of the network is $\approx$ 24M for the $N = 13$ case in the SIW-13 dataset. All convolution and fully connected layers use Rectified Linear Units (ReLU). In conv1 and conv2 layers we perform normalization over input regions using Local Response Normalization (LRN) [53]. At training time, we use dropout [125] (with a 0.5 ratio) in fc5 and fc6 layers.

To train the basic network model we use Stochastic Gradient Descent (SGD) with momentum and $L2$ regularization. We use mini-batches of 64 images. The base learning rate is set to 0.01 and is decreased by a factor of $\times 10$ every 100k iterations. The momentum weight parameter is set to 0.9, and the weight decay regularization parameter to 5×10^{-4}.

When training for individual patch classification, we build a dataset of small patches extracted by dense sampling the original training set images, as explained in section 8.2.1. Notice that this produces a large set of patch samples, e.g. in the SIW-13 dataset the number of training samples is close to half million. With these numbers the network converges after 250k iterations.

In the case of the Ensemble of Conjoined Networks the basic network detailed above is replicated N times, and all replicas are tied at their fc7 outputs with an element-wise sum layer which is connected to a single output SoftMax layer. All networks in the ECN share the same parameters values.

Training the ECN requires a dataset where each input sample is composed by N image patches. We generate this dataset as follows: given an input image we extract patches the same way as for the

simple network, then we generate random $N-$combinations of the image patches, allowing repetitions if the number of patches is $< N$. Notice that this way the number of samples can be increased up to very large-scale numbers because the number of possible different $N-$combinations is $\binom{M}{N}$ when the number of patches in a given image M is larger that the number of conjoined nets N, which is the usual case. This is an important aspect of ECNs, as the training dataset generation process becomes a data augmentation technique in itself. We can see this data augmentation process as generating new small text instances that are composed from randomly chosen parts of their original generators.

However, it is obviously non-practical to use all possible combinations for training; thus, in order to get a manageable number of samples, we have used the simple rule of generating $2 \times M$ samples per input, which for example in the SIW-13 dataset would produce around one million samples.

In terms of computational training complexity, the ECN has an important drawback compared to the simple network model: the number of computations is multiplied by N in each forward pass, similarly the amount of memory needed is linearly increased by N. To overcome this limitation, we use a fine-tuning approach to train ECNs. First, we train the simple network model, and then we do fine-tuning on the ECN parameters starting from the values learned using the simple net. When fine-tuning, we have found that starting from a fully converged network in the single-patch classification task we reach a local minimum of the global task, thus providing zero loss in most (if not all) the iterations and not allowing the network to learn anything new. In order to avoid this local minima situation we start the fine-tuning from a non-converged network (more or less at about 90/95% of the attainable individual patch classification accuracy).

Using fine-tuning with a base learning rate of 0.001 (decreasing $\times 10$ every 10k iterations) the ECN converges much faster, in the order of 35k iterations. All other learning parameters are set the same as in the simple network training setup.

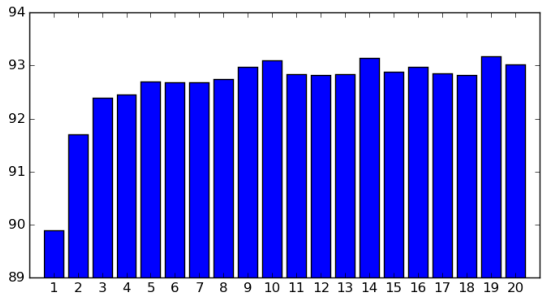


Figure 8.9: Validation accuracy for various number of networks N in the ensemble of conjoined networks model.

The number of nets N in the ensemble can be seen as an extra hyper-parameter in the ECN learning algorithm. Intuitively a dataset with larger text sequences would benefit from larger N values, while on the contrary in the extreme case of classifying small squared images (i.e. each image is represented by a single patch) any value of

$N > 1$ does not make sense. Since our datasets contain text instances with variable length a possible procedure to select the optimal value of N is by using a validation set. We have done experiments in the SIW-13 dataset by dividing the provided train set and keeping 10% for validation. Classification accuracy on the validation set for various N values are shown in Figure 8.9. As can be appreciated the positive impact of training with an ensemble of networks is evident for small values of N, and mostly saturated for values $N > 9$. In the following we use a value of $N = 10$ for all the remaining experiments.

8.3 *Experiments*

In this section we study the performance of the methods proposed in Section 8.2 and Section 8.1 for the tasks of script identification in pre-segmented text lines, joint text detection and script identification in scene images, and cross-domain performance. All reported experiments were conducted over three different datasets, namely the CVSI-2015, SIW-13, and MLe2e datasets.

8.3.1 *Script identification in pre-segmented text lines*

Table 8.1 shows the overall performance comparison of our methods with the state-of-the-art in all datasets for script identification in pre-segmented text lines. We also provide comparison with three well known image recognition pipelines using Scale Invariant Features [77] (SIFT) in three different encodings: Fisher Vectors, Vector of Locally Aggregated Descriptors (VLAD), and Bag of Words (BoW); and a linear SVM classifier. In all baselines we extract SIFT features at four different scales in sliding window with a step of 8 pixels. For the Fisher vectors we use a 256 visual words GMM, for VLAD a 256 vector quantized visual words, and for BoW 2,048 vector quantized visual words histograms. The step size and number of visual words were set to similar values to our method when possible in order to offer a fair evaluation. These three pipelines have been implemented with the VLFeat [132] and liblinear [30] open source libraries.

The method presented in Section 8.2 participated as a competing entry in the ICDAR2015 Competition on Video Script Identification (CVSI-2015) reaching the third place (CVC-2 entry in Table 8.1). While, as can be appreciated in Table 8.1, the winner of the competition [146] has better accuracy, our proposal was credited as a very close competitor in the competition report [115], and outperformed other entries with a noticeable margin. After the competition we find that with a more dense sampling strategy (using 8 pixels step for sliding window instead of 16) our results can be even better (CNN+NBNN entry in Table 8.1).

However, Table 8.1 indicates a clear weakness of this method when evaluated in the more challenging datasets of scene text (SIW-13 and MLe2e). This weakness motivated the extensions of our framework

Method	SIW-13	MLe2e	CVSI
This Chapter - Ensemble of Conjoined Nets	**94.8**	**94.4**	97.2
This Chapter - CNN (Avg.)	92.8	93.1	96.7
This Chapter - CNN + NBNN	76.9	91.12	97.91
Shi *et al.* [116]	89.4	-	94.3
HUST [117, 115]	88.0	-	96.69
Google [115]	-	-	**98.91**
Nicolaou *et al.* [103]	83.7	-	98.18
Singh *et al.* [120]	-	-	96.70
CVC-2 [44, 115]	-	88.16	96.0
SRS-LBP + KNN [104]	-	82.71	94.20
C-DAC [115]	-	-	84.66
CUK [115]	-	-	74.06
Baseline SIFT + Fisher Vectors + SVM	90.7	88.63	94.11
Baseline SIFT + VLAD + SVM	89.2	90.19	93.92
Baseline SIFT + Bag of Words + SVM	83.4	86.45	84.38

Table 8.1: Overall classification performance comparison with state-of-the-art in three different datasets: SIW-13 [116], MLe2e, and CVSI [115].

that we have detailed in Section 8.2.

As shown in Table 8.1 the method presented in 8.2 (CNN-ECN in the table) outperforms state of the art and all baseline methods in the SIW-13 and MLe2e scene text datasets, while performing competitively in the case of CVSI video overlay text dataset. In the SIW-13 dataset the proposed method significantly outperforms the best performing method known up to date by more than 4 percent points.

The contribution of training with ensembles of conjoined nets is significant in SIW-13 and MLe2e datasets, as can be appreciated by comparing the first two columns of Table 8.1 which correspond to the nets trained with the ensemble (first column) and the simple model (second column).

Our interpretation of the results in CVSI dataset in comparison with the ones obtained in SIW-13 and MLe2e relates to its distinct nature. CVSI overlaid-text variability and clutter is rather limited compared with that found in the scene text of MLe2e and SIW-13. As can be appreciated in Figure 8.10 overlaid-text is usually bi-level without much clutter. Figure 8.11 shows another important characteristic of CVSI dataset: since cropped words in the dataset belong to very long sentences of overlay text in videos, e.g. from rotating headlines, it is common to find a few dozens of samples sharing exactly the same font and background both in the train and test sets. This particularity makes the ECN network not really helpful in the case of CVSI, as the data augmentation by image patches recombination is somehow already implicit on the dataset.

Figure 8.10: Overlaid-text samples (top row) variability and clutter is rather limited compared with that found in the scene text images (bottom row).

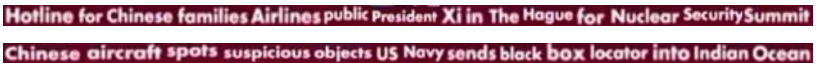


Figure 8.11: Cropped words in the CVSI dataset belong to very long sentences of overlay text in videos. It is common to find several samples sharing exactly the same font and background both in the train (top row) and test (bottom row) sets.

Furthermore, the CVSI-2015 competition winner (Google) makes use of a deep convolutional network but applies a binarization pre-processing to the input images. In our opinion this binarization may not be a realistic pre-processing in general for scene text images. As an example of this argument one can easily see in Figure 8.10 that binarization of scene text instances is not trivial as in overlay text. Similar justification applies to other methods performing better than ours in CVSI. In particular the LBP features used in [103], as well as the patch-level whitening used in our CNN-NBNN method, may potentially take advantage of the simpler, bi-level, nature of text instances in CVSI dataset. It is important to notice here that these two algorithms, have close numbers to the Google ones in CVSI-2015 (see Table 8.1) but perform quite bad in SIW-13.

Figure 8.12 shows the confusion matrices for the method presented in Section 8.2 in all three datasets with detailed per class classification results.

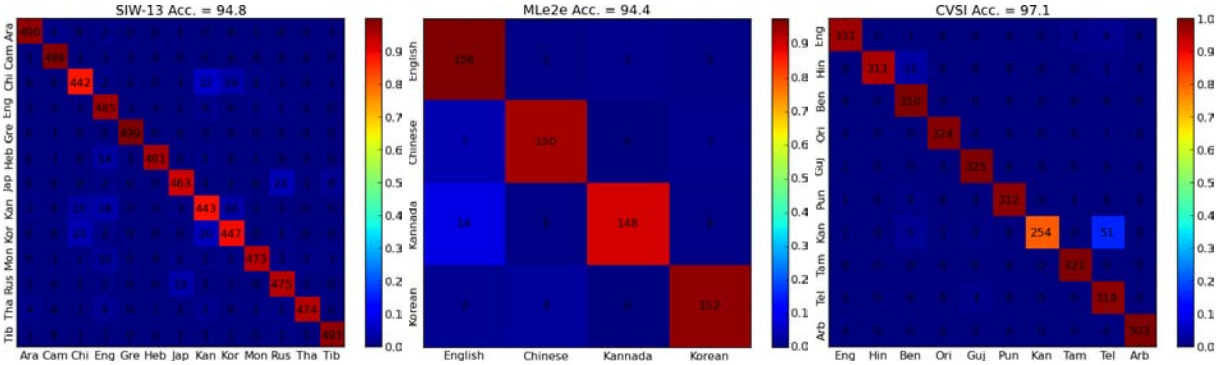
8.3.2 *Joint text detection and script identification in scene images*

In this experiment we evaluate the performance of a complete pipeline for detection and script identification in its joint ability to detect text lines in natural scene images and properly recognizing their scripts. The key interest of this experiment is to study the performance of the proposed script identification algorithm when realistic, non-perfect, text localization is available.

The experiments is performed over the new MLe2e dataset, in which script identification is performed at the text line level, because segmentation into words is largely script-dependent, and not meaningful in Chinese/Korean scripts. Notice however that in some cases, by the intrinsic nature of scene text, a text line provided by the text detection module may correspond to a single word, so we must deal with a large variability in the length of provided text lines.

We use the script-agnostic text detection method of Chapter 5 and

Figure 8.12: Confusion matrices with per class classification accuracy of the method presented in Section 8.2 in SIW-13, MLe2e, and CVSI datasets.



apply the script identification methods presented in this Chapter to the bounding boxes of detected regions.

For evaluation of the joint text detection and script identification task in the MLe2e dataset we propose the use of a simple two-stage evaluation framework. First, localization is assessed based on the Intersection-over-Union (IoU) metric between detected and ground-truth regions, as commonly used in object detection tasks [28] and the recent ICDAR 2015 Robust Reading Competition[1] [57]. Second, the predicted script is verified against the ground-truth. A detected bounding box is thus considered correct if it has a IoU > 0.5 with a bounding box in the ground-truth and the predicted script is correct.

[1] http://rrc.cvc.uab.es

The localization-only performance, corresponding to the first stage of the evaluation, yields an F-score of 0.63 (Precision of 0.57 and Recall of 0.69). This defines the upper-bound for the joint task. The two stage evaluation, including script identification, of the proposed method compared with our previous work is shown in Table 8.2.

Method	Correct	Wrong	Missing	Precision	Recall	F-score
This Chapter - ECN	**395**	**376**	**245**	**0.51**	**0.62**	**0.56**
This Chapter - CNN + NBNN	364	407	278	0.47	0.57	0.52

Table 8.2: Text detection and script identification performance in the MLe2e dataset.

Intuitively the proposed method for script identification is effective even when the text region is badly localized, as long as part of the text area is within the localized region. To support this argument we have performed an additional experiment where our algorithm is applied to cropped regions from pre-segmented text images. For this, we take the SIW-13 original images and calculate the performance of our method when applied to cropped regions of variable length, up to the minimum size possible (40×40 pixels). As can be appreciated in Figure 8.14 the experiment demonstrates that the proposed method is effective even when small parts of the text lines are provided. Such a behavior is to be expected, due to the way our method treats local information to decide on a script class. In the case of the pipeline for joint detection and script identification, this extends to regions that did not pass the 0.5 IoU threshold, but had their script correctly identified. This opens the possibility to make use of script identification to inform and / or improve the text localization process. The information of the identified script can be used to refine the detections.

Figure 8.13: Sample results of our method for the joint task of text detection and script recognition in natural scenes.

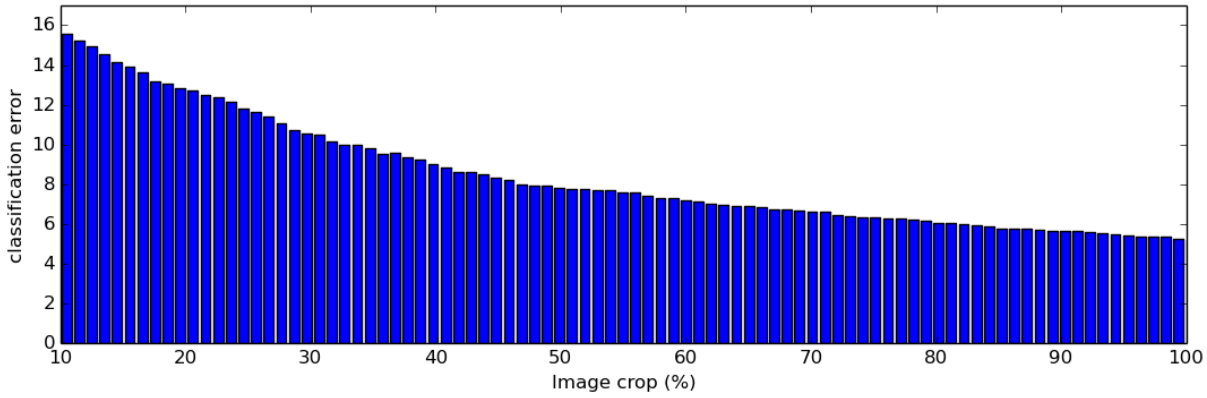


Figure 8.14: Classification error of our method CNN-ECN when applied to variable length cropped regions of SIW-13 images, up to the minimum size possible (40×40 pixels).

8.3.3 *Cross-domain performance and confusion in single-language datasets*

In this experiment we evaluate the cross-domain performance of learned CNN-ECN weights from one dataset to the other. For example, we evaluate on the MLe2e and CVSI test sets using the network trained with the SIW-13 train set, by measuring classification accuracy only for their common script classes: Arabic, English, and Kannada in CVSI; Chinese, English, Kannada, and Korean in MLe2e. Finally, we evaluate the misclassification error of our method (trained in different datasets) over two single-script datasets. For this experiment we use the ICDAR2013 [60] and ALIF [147] datasets, which provide cropped word images of English scene text and Arabic video overlaid text respectively. Table 8.3 shows the results of these experiments.

Method	SIW-13	MLe2e	CVSI	ICDAR	ALIF
ECN CNN (SIW-13)	94.8	86.8	90.6	74.7	100
ECN CNN (MLe2e)	90.8	94.4	98.3	95.3	-
ECN CNN (CVSI)	42.3	43.5	97.2	65.2	91.8

Table 8.3: Cross-domain performance of our method measured by training/testing in different datasets.

Notice that results in Table 8.3 are not directly comparable among rows because each classifier has been trained with a different number of classes, thus having different rates for a random choice classification. However, the experiment serves as a validation of how good a given classifier is in performing with data that is distinct in nature to the one used for training. In this sense, the obtained results show a clear weakness when the model is trained on the video overlaid text of CVSI and subsequently applied to scene text images (SIW-13, MLe2e, and ICDAR). On the contrary, models trained on scene text datasets are quite stable in other scene text data, as well as in video overlaid text (CVSI and ALIF).

In fact, this is an expected result, because the domain of video overlay text can be seen as a subdomain of the scene text domain. Since the scene text datasets are richer in text variability, e.g. in

terms of perspective distortion, physical appearance, variable illumination, and typeface designs, script identification on these datasets is a more difficult problem, and their data is more indicated if one wants to learn effective cross-domain models. This demonstrates that our method is able to learn discriminative stroke-part representations that are not dataset-specific, and provides evidence to the claims made in section 8.3.1 when interpreting the obtained results in CVSI dataset comparing with other methods that may be more engineered to the specific CVSI data but not generalizing well in scene text datasets.

8.4 *Conclusion*

In this Chapter we have presented two different patch-based methods for script identification in natural scene images. One of the methods combines the expressive representation of convolutional features and the fine-grained classification characteristics of the Naive-Bayes Nearest Neighbor classifier. The other is based on the use of ensembles of conjoined convolutional networks to jointly learn discriminative stroke-parts representations and their relative importance in a patch-based classification scheme.

Experiments done in three different datasets exhibit state of the art accuracy rates in comparison to a number methods, including the participants in the CVSI-2015 competition and standard image recognition pipelines.

Chapter 9
Applications

In this Chapter we present two end-to-end pipelines that are build on top the the methods developed thorough this thesis. Althought beign primarly an engineering work, because we make use of existing solutions, their results serve as an argument to further demonstrate the value of the contributions made in this thesis.

In Section 9.1 we build a multi-lingual end-to-end reading system by combining the scene text detection method presented in Chapter 5 with the script identification methods of Chapter 8 and two different off-the-shelf OCR engines. Second, in Section 9.2 we apply the text specific object proposals explained in Chapter 5 with two state of the art whole-word recognition methods in order to build an end-to-end word spoting system.

Finally, in Section 9.3 we give detail of all public releases of code, data, and models developed as part of the work in this thesis. This includes some parts of the recently developed OpenCV text module.

9.1 *Unconstrained Text Recognition with off-the-shelf OCR engines*

It is a generally accepted fact that Off-the-shelf OCR engines do not perform well in unconstrained scenarios like natural scene imagery, where text appears among the clutter of the scene.

A typical experiment frequently repeated to demonstrate the need of specific techniques for scene text detection and recognition is to attempt to process a raw scene image with a conventional OCR engine. This normally produces a bunch of garbage on the recognition output. Obviously this is not the task for which OCR engines have been designed and the recognition may be much better if we provide it with a pixel level segmentation of the text. Figure 9.1 show the output of the open source Tesseract[1] [121] OCR engine for a raw scene image and for its binarized text mask obtained with a scene text extraction method.

[1] http://code.google.com/p/tesseract-ocr/

Recent work [84, 43, 85] indicates that a conventional shape-based OCR engine would be able to produce competitive results in the end-to-end scene text recognition task when provided with a con-

Figure 9.1: Off-the-shelf OCR engine recognition accuracy may increase to acceptable rates if we provide pixel level text segmentation instead of raw pixels.

veniently pre-processed image. In this section we confirm this finding with a set of experiments where an off-the-shelf OCR engine is combined with our scene text detection framework. The obtained results demonstrate that in such pipeline, conventional OCR solutions still perform competitively compared to other solutions specifically designed for scene text recognition.

While being this primarly an engineering work, we think there are two important aspects of it that can be of broad interest from a research perspective: one is about the question of whether pixel-level text segmentation is really useful for the final recognition, the other is about how stronger language models affect the final results.

9.1.1 *End-to-end Pipeline*

In order to build a multi-lingual end-to-end reading system we combine the scene text detection method presented in Chapter 5 with the script identification methods of Chapter 8 and an off-the-shelf OCR engine: the open source project Tesseract[2] [121].

[2] http://code.google.com/p/tesseract-ocr/

The setup of the OCR engine in our pipeline is minimal: given a text detection hypotheses from the detection module we set the recognition language to the one provided by the script identification module, and we set the OCR to interpret the input as a single text line. Apart from that we use the default Tesseract parameters.

The recognition output is filtered with a simple post-processing junk filter in order to eliminate garbage recognitions, i.e. sequences of identical characters like "IIii" that may appear as the results of trying to recognize repetitive patterns in the scene. Concretely, we discard the words in which more than half of their characters are recognized as one of "i", "l", "I", or other special characters like: punctuation marks, quotes, exclamation, etc. We also reject those detections for which the recognition confidence provided by the OCR engine is under a certain threshold.

9.1.2 *Experiments*

We conduct our experiments on the MLe2e and ICDAR2011 datasets respectively for multi-linugual and English-only scene text end-to-end recognition.

Figure 9.2: End-to-end recognition of text from images containing textual information in different scripts/languages.

Multi-lingual scene text end-to-end recognition

Table 9.1 shows a comparison of the proposed end-to-end pipeline by using different script identification modules: the two methods presented in Chapter 8, and Tesseract's built-in alternative. Figure 9.2 shows the output of our full end-to-end pipeline for some images in the MLe2e test set.

Tesseract method in Table 9.1 refers to the use of Tesseract's own script estimation algorithm [130]. We have found that Tesseract's algorithm is designed to work with large corpuses of text (e.g. full page documents) and does not work well for the case of single text lines.

Script identification	Correct	Wrong	Missing	Precision	Recall	F-score
Chapter 8 ECN	**96**	212	**503**	0.31	**0.16**	**0.21**
Chapter 8 CNN-NBNN	82	211	517	0.28	0.14	0.18
Tesseract	50	**93**	549	**0.35**	0.08	0.13

Table 9.1: End-to-end multi-lingual recognition performance in the MLe2e dataset.

Results in Table 9.1 demonstrate the direct correlation between having better script identification rates and better end-to-end recognition results.

The final multi-lingual recognition f-score obtained (0.21) is far from the state-of-the art in end-to-end recognition systems designed for English-only environments [7, 52, 102]. As a fair comparison, a very similar pipeline using the Tesseract OCR engine [43] achieves an f-score of 0.37 in the ICDAR English-only dataset (see next Section). The lower performance obtained in MLe2e dataset stems from a number of challenges that are specific to its multi-lingual nature. For example, in some scripts (e.g. Chinese and Kannada) glyphs many times not single-body regions, composed by (or complemented with) small strokes that in many cases are lost in the text segmenta-

tion stage. In such cases having a bad pixel-level segmentation of text would make it practically impossible for the OCR engine to produce a correct recognition.

Our pipeline results represent the first reference result for multilingual scene text recognition and a first benchmark from which better systems can be built, e.g. replacing the off-the-shelf OCR engine by other recognition modules better suited for scene text imagery.

English-only scene text end-to-end recognition

Table 9.2 shows the comparison of the best obtained results of the two evaluated OCR engines with current state of the art. In both cases our results outperform [98] in total f-score while, as stated before, our detection pipeline is a simplified implementation of that method. However, it is important to notice that our recall is in general lower that in the other methods, while it is our precision what makes the difference in f-score. A similar behaviour can be seen for the method of Milyaev *et al.*[84], which also uses a commercial OCR engine for the recognition. Such higher precision rates indicate that in general off-the-shelf OCR engines are doing very good in rejecting false detections.

Notice that Table 9.2 does not include the method in [7] because end-to-end results are not available. However, it would be expected to be in the top of the table if we take into account their cropped word recognition rates and their high recall detection strategy.

Method	Precision	Recall	F-score
Milyaev *et al.*[84]	66.0	46.0	54.0
Yao *et al.* [142]	49.2	44.0	45.4
Neumann and Matas [100]	44.8	45.4	45.2
OpenCV + Tesseract	52.9	32.4	40.2
Neumann and Matas [99]	37.8	39.4	38.6
Chapter 5 + Tesseract	48.1	30.7	37.5
Neumann and Matas [98]	37.1	37.2	36.5
Wang *et al.* [137] *	54.0	30.0	38.0

Table 9.2: End-to-end results in the ICDAR 2011 dataset comparing our proposed pipeline with other state-of-the-art methods. Methods marked with an asterisk evaluate on a slightly different dataset (ICDAR 2003).

Figure 9.3: Qualitative results on well recognized text using the CSER+Tesseract pipeline.

Figure 9.4: Common misspelling mistakes using the CSER+Tesseract pipeline are due to missed diacritics, rare font types, and/or poor segmentation of some characters.

Figures 9.3, 9.4, and 9.5 show qualitative results of the evaluated pipeline. The end-to-end system recognizes words correctly in a variety of different situations, including difficult cases, e.g. where text appears blurred, or with non standard fonts. Common misspelling mistakes are, most of the time, due to missed diacritics, rare font types, and/or poor segmentation of some characters. Also in many cases the OCR performs poorly because the text extraction algorithm is not able to produce a good segmentation, e.g. in challenging situations or in cases where characters are broken in several strokes.

9.1.3 *Discussion*

In this section we have seen that pixel level segmentation methods are still among the best solutions in state of the art end-to-end recognition for focused scene text. It is true however that scene text is not always binarizable and that in such cases other techniques must be employed. But current segmentation methods combined with existing OCR technologies may produce optimal results in many cases, by taking advantage of more than 40 years of research and development in automated reading systems, e.g. all the accumulated knowledge of shape-based classifiers, and the state of the art in language modelling for OCR.

Figure 9.5: Common errors when segmentation is particularly challenging or characters are broken in several strokes.

Figure 9.6: End-to-end text recognition results on the Street View Text dataset.

9.2 *End-to-end word spotting*

In this section we combine the text specific object proposals explained in Chapter 5 with two state of the art whole-word recognition methods [2, 51] in order to build an end-to-end word spoting system.

End-to-end text recognition and image retrieval results

Table 9.3 shows end-to-end recognition F-scores on SVT, ICDAR2003, and ICDAR2015 datasets.

	IC03-50	IC03-Full	IC03	SVT-50	SVT	IC15-50	IC15-Full	IC15
Wang *et al.*[135]	0.68	0.61	-	0.38	-	-	-	-
Wang and Wu [137]	0.72	0.67	-	0.46	-	-	-	-
Alsharif [3]	0.77	0.70	0.63*	0.48	-	-	-	-
Jaderberg *et al.*[52]	0.80	0.75	-	0.56	-	-	-	-
Jaderberg *et al.*[51]	0.90	0.86	0.78	0.76	**0.53**	**0.90****	-	0.76
StradVision-1	-	-	-	-	-	0.86	0.83	0.70
Deep2Text II-2	-	-	-	-	-	0.77	0.77	0.77
Chapter 6 + Almazan *et al.*[2]	0.82	0.73	-	0.67	-	-	-	-
Chapter 6 + Jaderberg *et al.*[51]	**0.92**	**0.90**	0.75	**0.85**	0.52	0.85	**0.84**	0.71

Table 9.3: Comparison of end-to-end word spotting F-scores on ICDAR2003, ICDAR2015, and SVT datasets.

9.2.1 *Discusion*

In this section we described a complete system for robust reading of text in natural scene images able to perform both end-to-end text spoting and image retrieval based on textual information. The system builds upon two holistic word recognition methods: one is based on a compact attribute-based word representation that permits to perform word recognition and retrieval in an unified and integrated way [2]; the other consists in a deep CNN model trained to recognize words by assigning a label among 90k possible classes. Word recognition is applied on a set of candidate text boxes returned by the selective search text localization approach presented in Chapter 6 that generates multiple word hypotheses. Results on the SVT and ICDAR datasets show competitive results both for text recognition and retrieval tasks.

9.3 Public releases

Open Source algorithm implementations are of great help for new researchers, they serve as reference baselines to compare against, and permit to focus research in a certain part of the text extraction pipeline, while using other "standard" implemented modules, and analyse the improvement of their work. We strongly believe that releasing open source implementations of state of the art methods would help improving the experience of the research community interested in the text extraction problem, as well as having a positive impact in the quality of new methods to come.

For a long time no such open framework has been made available but, fortunately things are changing. In the past four years code of a few text extraction methods has been released to the public (not necessarily with open source licenses). These include the work of Kai Wang[3], the work of Rodrigo Minnetto[4] [88], an implementation of the Stroke Width Transform [27] in the libccv project[5], and our own work in multi-script text extraction and script identification.

[3] http://vision.ucsd.edu/~kai/grocr/

[4] http://www.dainf.ct.utfpr.edu.br/~rminetto/projects/snoopertext/

[5] http://libccv.org/

Multi-script Text Extraction method

We have released the code of the text extraction method presented in Chapter4 and published in an ICDAR 2015 paper. The released code[6] allows the recreation of Table 4.1.

[6] http://github.com/lluisgomez/text_extraction

Text Objet Proposals

We have released the code of our text specific object proposals method detailed in Chapter6. The released code allows the recreation of Tables 6.2 and 6.3, as well as the plots in Figure 6.3.

Script identification code and CNN models

We have released the code of the two methods for script identification described in Chapter8. The released code and CNN models allow the recreation of the results shown in Tables 8.1 and 8.3.

OpenCV Text module

The author of this thesis has been allocated to the Google Summer of Code (GSoC) program[7] 2013 and 2014 editions. This has supposed an exceptional opportunity to work with members of the OpenCV[8] library development team in order to implement state-of-the-art scene text detection and recognition methods [98][42]. The project, that has produced already some code[9] and its development is still going on, would hopefully make available to the OpenCV community a good quality open source implementation of a complete scene text end-to-end understanding system.

[7] http://www.google-melange.com/gsoc/homepage/google/gsoc2014

[8] http://opencv.org

[9] http://docs.opencv.org/trunk/modules/objdetect/doc/erfilter.html

Chapter 10

Conclusion and Future Work

In this thesis we have contributed several methods to the state of the art of automatic scene text understanding in unconstrained conditions. Our contributions are mainly on the on multi-language and arbitrary-oriented text detection, tracking, and recognition in natural scene images and videos.

In Chapter 4 a new methodology for text extraction from scene images was presented, inspired by the human perception of textual content, largely based on perceptual organisation. The proposed method requires practically no training as the perceptual organisation based analysis is parameter free. It is totally independent of the language and script in which text appears, it can deal efficiently with any type of font and text size, while it makes no assumptions about the orientation of the text.

In Chapter 5 we have detailed a scene text extraction method in which the exploitation of the hierarchical structure of text plays an integral part. We have shown that the algorithm can efficiently detect text groups whith arbitrary orientation in a single clustering process that involves: a learned optimal clustering feature space for text region grouping, novel discriminative and probabilistic stopping rules, and a new set of features for text group classification that can be efficiently calculated in an incremental way.

In Chapter 6 we have evaluated the performance of generic Object Proposals in the task of detecting text words in natural scenes. We have presented a text specific method that is able to outperform generic methods in many cases, or to show competitive numbers in others. Moreover, the proposed algorithm is parameter free and fits well the multi-script and arbitrary oriented text scenario.

In Chapter 7 we have presented a method for detection and tracking of scene text able to work in real-time on low-resource mobile devices. Although far from being a final solution, the proposed method goes beyond the full-detection approaches in terms of time performance optimization. The combination of text detection with a tracker, provides considerable stability, allowing the system to provide predicted estimates in cases where the detection module itself is not capable of returning a valid response. The use of MSER-tracking as an alternative, fast technique to provide simulated text detections

for the frames that are not processed by the full frame text detector proves to be an adequate solution, providing the system with enough information to continue tracking until the text detector returns updated positions.

In Chapter 8 a patch-based framework for script identification in natural scene images was presented. The two proposed methods are based on the intuition that effective script identification must leverage the discriminative power of certain small patches of the image. For this we rely on the use of ensembles of conjoined convolutional networks to jointly learn discriminative stroke-part representations and their relative importance in a patch-based classification scheme. Experiments performed in three different datasets exhibit state of the art accuracy rates in comparison to a number of methods, including three standard image classification pipelines. Our work demonstrates the viability of script identification in natural scene images, paving the road towards true multi-lingual end-to-end scene text understanding.

Future work

Improved text regions proposals. An interesting observation of our experiments in Chapter 6 is that while class-independent object proposals methods suffice with near a thousand proposals to achieve high recall rates for object detection, in the case of text we still need around 10000 in order achieve similar numbers. This indicates there is a large room for improvement in text specific Object Proposals methods. One possible direction would be to improve the quality of the proposals ranking with better classifiers while mantaining low time complexity. The perceptual organization approach presented in Chapter 4 opens up a number of possible paths for future research in object proposals methods, including the higher integration of the region decomposition stage with the perceptual organisation analysis, and further investigation on the computational modelling of perceptual organisation aspects such as masking, conflict and collaboration.

Integration of script-independent and script-specific approaches. In Chapter 8 we have seen that script identification is effective even when the text region is badly localized, as long as part of the text area is within the localized region. This opens the possibility to make use of script identification to inform and / or improve the text localization process. The information of the identified script can be used to refine the detections with an ad-hoc detection method specialized in a certain script. On the other hand, end-to-end word spotting systems like the one built in Chapter 9 may be extended to multi-linugual environments by training independent per-script whole word recognizers.

Bibliography

[1] Bogdan Alexe, Thomas Deselaers, and Vittorio Ferrari. Measuring the objectness of image windows. *TPAMI*, 2012.

[2] J. Almazán, A. Gordo, A. Fornés, and E. Valveny. Word spotting and recognition with embedded attributes. In *TPAMI*, 2014.

[3] Ouais Alsharif and Joelle Pineau. End-to-end text recognition with hybrid hmm maxout models. *arXiv preprint arXiv:1310.1811*, 2013.

[4] Pablo Arbelaez, Michael Maire, Charless Fowlkes, and Jitendra Malik. Contour detection and hierarchical image segmentation. *TPAMI*, 2011.

[5] Andrew D. Bagdanov, Alberto Del Bimbo, Fabrizio Dini, Giuseppe Lisanti, and Iacopo Masi. Posterity logging of face imagery for video surveillance. *IEEE Multimedia*, 2012.

[6] Keni Bernardin and Rainer Stiefelhagen. Evaluating multiple object tracking performance: The CLEAR MOT metrics. *EURASIP JIVP*, 2008.

[7] Alessandro Bissacco, Mark Cummins, Yuval Netzer, and Hartmut Neven. Photoocr: Reading text in uncontrolled conditions. In *ICCV*, 2013.

[8] Oren Boiman, Eli Shechtman, and Michal Irani. In defense of nearest-neighbor based image classification. In *CVPR*, 2008.

[9] Gary R Bradski. Computer vision face tracking for use in a perceptual user interface. In *Proc. WACV*, 1998.

[10] Jane Bromley, James W Bentz, Léon Bottou, Isabelle Guyon, Yann LeCun, Cliff Moore, Eduard Säckinger, and Roopak Shah. Signature verification using a "siamese" time delay neural network. *International Journal of Pattern Recognition and Artificial Intelligence*, 1993.

[11] Frédéric Cao, Julie Delon, Agnès Desolneux, Pablo Musé, and Frédéric Sur. An a contrario approach to hierarchical clustering validity assessment. Technical report, INRIA, 2004.

[12] Santanu Chaudhury and Rabindra Sheth. Trainable script identification strategies for indian languages. In *Document Analysis and Recognition, 1999. ICDAR'99. Proceedings of the Fifth International Conference on*, 1999.

[13] H. Chen, S.S. Tsai, G. Schroth, D.M. Chen, R. Grzeszczuk, and B. Girod. Robust text detection in natural images with edge-enhanced maximally stable extremal regions. In *Proc. ICIP*, 2011.

[14] Xiangrong Chen and A.L. Yuille. Detecting and reading text in natural scenes. In *Proc. CVPR*, 2004.

[15] Ming-Ming Cheng, Ziming Zhang, Wen-Yan Lin, and Philip Torr. Bing: Binarized normed gradients for objectness estimation at 300fps. In *CVPR*, 2014.

[16] Antonio Clavelli, Dimosthenis Karatzas, and Josep Lladós. A framework for the assessment of text extraction algorithms on complex colour images. In *Proceedings of the 9th IAPR International Workshop on Document Analysis Systems*, pages 19–26. ACM, 2010.

[17] A. Coates, B. Carpenter, C. Case, S. Satheesh, B. Suresh, Tao Wang, D.J. Wu, and A.Y. Ng. Text detection and character recognition in scene images with unsupervised feature learning. In *Proc. ICDAR*, 2011.

[18] Adam Coates, Andrew Y Ng, and Honglak Lee. An analysis of single-layer networks in unsupervised feature learning. In *AIStats*, 2011.

[19] David Crandall, Sameer Antani, and Rangachar Kasturi. Extraction of special effects caption text events from digital video. *IJDAR*, 2003.

[20] T. E. de Campos, B. R. Babu, and M. Varma. Character recognition in natural images. In *ICCVTA*, 2009.

[21] Agnès Desolneux, Lionel Moisan, and Jean-Michel Morel. A grouping principle and four applications. *IEEE Trans. PAMI*, 2003.

[22] Piotr Dollár and C Lawrence Zitnick. Structured forests for fast edge detection. In *ICCV*, 2013.

[23] Michael Donoser, Clemens Arth, and Horst Bischof. Detecting, tracking and recognizing license plates. In *ACCV*, 2007.

[24] Michael Donoser and Horst Bischof. Efficient maximally stable extremal region (mser) tracking. In *CVPR*, 2006.

[25] Michael Donoser and Horst Bischof. Real time appearance based hand tracking. In *ICPR*, 2008.

[26] Michael Donoser, Hayko Riemenschneider, and Horst Bischof. Shape guided maximally stable extremal region tracking. In *ICPR*, 2010.

[27] Boris Epshtein, Eyal Ofek, and Yonatan Wexler. Detecting text in natural scenes with stroke width transform. In *Proc. CVPR*, 2010.

[28] Mark Everingham, SM Ali Eslami, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes challenge: A retrospective. *IJCV*, 2014.

[29] Mark Everingham, SM Ali Eslami, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes challenge: A retrospective. *International Journal of Computer Vision*, 111(1):98–136, 2015.

[30] Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. LIBLINEAR: A library for large linear classification. *JMLR*, 2008.

[31] Pedro F Felzenszwalb and Daniel P Huttenlocher. Efficient graph-based image segmentation. *IJCV*, 2004.

[32] Miguel A Ferrer, Aythami Morales, and Umapada Pal. Lbp based line-wise script identification. In *Document Analysis and Recognition (ICDAR), 2013 12th International Conference on*, 2013.

[33] Jan Flusser and Tomas Suk. Pattern recognition by affine moment invariants. *Pattern Recognition*, 1993.

[34] Jan Flusser and Tomáš Suk. Affine moment invariants: a new tool for character recognition. *Pattern Recognition Letters*, 1994.

[35] Victor Fragoso, Steffen Gauglitz, Shane Zamora, Jim Kleban, and Matthew Turk. Translatar: A mobile augmented reality translator. In *WACV*, 2011.

[36] A.L.N. Fred and A.K. Jain. Combining multiple clusterings using evidence accumulation. *IEEE Trans. PAMI*, 2005.

[37] Debashis Ghosh, Tulika Dube, and Adamane P Shivaprasad. Script recognition—a review. *PAMI*, 2010.

[38] Suman K Ghosh, Lluis Gomez, Dimosthenis Karatzas, and Ernest Valveny. Efficient indexing for query by string text retrieval. In *Document Analysis and Recognition (ICDAR), 2015 13th International Conference on*, pages 1236–1240. IEEE, 2015.

[39] Julinda Gllavata and Bernd Freisleben. Script recognition in images with complex backgrounds. In *SPIT*, 2005.

[40] Julinda Gllavata, Ermir Qeli, and Bernd Freisleben. Detecting text in videos using fuzzy clustering ensembles. In *ISM*, 2006.

[41] Vibhor Goel, Anand Mishra, Karteek Alahari, and CV Jawahar. Whole is greater than sum of parts: Recognizing scene text words. In *ICDAR*, 2013.

[42] Lluis Gomez and Dimosthenis Karatzas. Multi-script text extraction from natural scenes. In *ICDAR*, 2013.

[43] Lluís Gómez and Dimosthenis Karatzas. Scene text recognition: No country for old men? In *Computer Vision-ACCV 2014 Workshops*, 2014.

[44] Lluis Gomez-Bigorda and Dimosthenis Karatzas. A fine-grained approach to scene text script identification. In *DAS*, 2016.

[45] Hideaki Goto. Redefining the dct-based feature for scene text detection. *IJDAR*, 2008.

[46] Hideaki Goto and Makoto Tanaka. Text-tracking wearable camera system for the blind. In *ICDAR*, 2009.

[47] Chunhui Gu, Joseph J Lim, Pablo Arbeláez, and Jitendra Malik. Recognition using regions. In *CVPR*, 2009.

[48] Ismail Haritaoglu. Scene text extraction and translation for handheld devices. In *CVPR*, 2001.

[49] Jan Hosang, Rodrigo Benenson, and Bernt Schiele. How good are detection proposals, really? In *BMVC*, 2014.

[50] Ming-Kuei Hu. Visual pattern recognition by moment invariants. *Trans. on IRE*, 1962.

[51] Max Jaderberg, Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Reading text in the wild with convolutional neural networks. *arXiv preprint arXiv:1412.1842*, 2014.

[52] Max Jaderberg, Andrea Vedaldi, and Andrew Zisserman. Deep features for text spotting. In *ECCV*, 2014.

[53] Kevin Jarrett, Koray Kavukcuoglu, Marc'Aurelio Ranzato, and Yann LeCun. What is the best multi-stage architecture for object recognition? In *Computer Vision, 2009 IEEE 12th International Conference on*, 2009.

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

[55] H Judith, K Patrick, T Timothy, et al. Automatic script identification from document images using cluster-based templates. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1995.

[56] Keechul Jung, Kwang In Kim, and Anil K. Jain. Text information extraction in images and video: a survey. *Pattern Recognition*, 2004.

[57] Dimosthenis Karatzas, Lluis Gomez-Bigorda, Anguelos Nicolaou, Suman Ghosh, Andrew Bagdanov, Masakazu Iwamura, Jiri Matas, Lukas Neumann, Vijay Ramaseshan Chandrasekhar, Shijian Lu, et al. Icdar 2015 competition on robust reading. In *Document Analysis and Recognition (ICDAR), 2015 13th International Conference on*, 2015.

[58] Dimosthenis Karatzas, S Robles Mestre, Joan Mas, Farshad Nourbakhsh, and Partha Pratim Roy. Icdar 2011 robust reading competition-challenge 1: Reading text in born-digital images (web and email). In *ICDAR*, 2011.

[59] Dimosthenis Karatzas, Sergi Robles-Mestre, and Lluis Gomez. An on-line platform for ground truthing and performance evaluation of text extraction systems. In *DAS*, 2014.

[60] Dimosthenis Karatzas, Faisal Shafait, Seichi Uchida, Masakatzu Iwamura, Lluis Gomez, Sergi Robles Mestre, Joan Mas, David Fernandez, Jon Almazàn, and Lluis Pere de las Heras. ICDAR 2013 robust reading competition. In *ICDAR*, 2013.

[61] Rangachar Kasturi, Dmitry Goldgof, Padmanabhan Soundararajan, Vasant Manohar, John Garofolo, Rachel Bowers, Matthew Boonstra, Valentina Korzhova, and Jing Zhang. Framework for performance evaluation of face, text, and vehicle detection and tracking in video: Data, metrics, and protocol. *IEEE Trans. PAMI*, 2009.

[62] Bernardin Keni and Stiefelhagen Rainer. Evaluating multiple object tracking performance: the clear mot metrics. *EURASIP JIVP*, 2008.

[63] A. Kessy, A. Lewin, and K. Strimmer. Optimal whitening and decorrelation. *arXiv preprint arXiv:1512.00809*, 2015.

[64] Kwang In Kim, Keechul Jung, and Jin Hyung Kim. Texture-based approach for text detection in images using support vector machines and continuously adaptive mean shift algorithm. *IEEE Trans. PAMI*, 2003.

[65] Philipp Krähenbühl and Vladlen Koltun. Geodesic object proposals. In *ECCV*, 2014.

[66] Jonathan Krause, Timnit Gebru, Jia Deng, Li-Jia Li, and Li Fei-Fei. Learning features and parts for fine-grained recognition. In *ICPR*, 2014.

[67] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*. 2012.

[68] Deepak Kumar, MN Prasad, and AG Ramakrishnan. Multi-script robust reading competition in ICDAR 2013. In *ICDAR - MOCR Workshop*, 2013.

[69] Deepak Kumar and AG Ramakrishnan. Otcymist: Otsu-canny minimal spanning tree for born-digital images. In *Document Analysis Systems (DAS), 2012 10th IAPR International Workshop on*, pages 389–393. IEEE, 2012.

[70] DS Lee, Craig R Nohl, and Henry S Baird. Language identification in complex, unoriented, and degraded document images. *SERIES IN MACHINE PERCEPTION AND ARTIFICIAL INTELLIGENCE*, 1998.

[71] SeongHun Lee, Min Su Cho, Kyomin Jung, and Jin Hyung Kim. Scene text extraction with edge constraint and text collinearity. In *Proc. ICPR*, 2010.

[72] Huiping Li and David Doermann. Text enhancement in digital video using multiple frame integration. In *ICM*, 1999.

[73] Huiping Li, David Doermann, and Omid Kia. Automatic text detection and tracking in digital video. *IEEE Trans. IP*, 2000.

[74] Lin Li, Shengsheng Yu, Luo Zhong, and Xiaozhen Li. Multilingual text detection with nonlinear neural network. *Mathematical Problems in Engineering*, 2015.

[75] Jian Liang, David Doermann, and Huiping Li. Camera-based analysis of text and documents: a survey. *IJDAR*, 2005.

[76] David G. Lowe. *Perceptual Organization and Visual Recognition*. Kluwer Academic Publishers, 1985.

[77] David G Lowe. Object recognition from local scale-invariant features. In *ICCV*, 1999.

[78] Simon M. Lucas, Alex Panaretos, Luis Sosa, Anthony Tang, Shirley Wong, and Robert Young. ICDAR 2003 robust reading competitions: entries, results, and future directions. *IJDAR*, 2005.

[79] Santiago Manen, Matthieu Guillaumin, and Luc Van Gool. Prime object proposals with randomized prim's algorithm. In *ICCV*, 2013.

[80] J Matas, O Chum, M Urban, and T Pajdla. Robust wide-baseline stereo from maximally stable extremal regions. *Image and Vision Computing*, 2004.

[81] Jiri Matas and Karel Zimmermann. A new class of learnable detectors for categorisation. In *Image Analysis*, 2005.

[82] Carlos Merino and Majid Mirmehdi. A framework towards realtime detection and tracking of text. In *CBDAR*, 2007.

[83] C. Merino-Gracia, K. Lenc, and M. Mirmehdi. A head-mounted device for recognizing text in natural scenes. *Proc. of Int. Workshop on Camera-based Document Analysis and Recognition*, pages 27–32, September 2011.

[84] Sergey Milyaev, Olga Barinova, Tatiana Novikova, Pushmeet Kohli, and Victor Lempitsky. Image binarization for end-to-end text understanding in natural images. In *Document Analysis and Recognition (ICDAR), 2013 12th International Conference on*, pages 128–132. IEEE, 2013.

[85] Sergey Milyaev, Olga Barinova, Tatiana Novikova, Pushmeet Kohli, and Victor Lempitsky. Fast and accurate scene text understanding with image binarization and off-the-shelf ocr. *International Journal on Document Analysis and Recognition (IJDAR)*, 2015.

[86] Rodrigo Minetto, Nicolas Thome, Matthieu Cord, Jonathan Fabrizio, and Beatriz Marcotegui. Snoopertext: A multiresolution system for text detection in complex visual scenes. In *ICIP*, 2010.

[87] Rodrigo Minetto, Nicolas Thome, Matthieu Cord, Neucimar J Leite, and Jorge Stolfi. Text detection and tracking for outdoor videos. In *ICIP*, 2011.

[88] Rodrigo Minetto, Nicolas Thome, Matthieu Cord, Neucimar J Leite, and Jorge Stolfi. Snoopertext: A text detection system for automatic indexing of urban scenes. *Computer Vision and Image Understanding*, 2013.

[89] Rodrigo Minetto, Nicolas Thome, Matthieu Cord, Neucimar J Leite, and Jorge Stolfi. T-hog: An effective gradient-based descriptor for single line text regions. *Pattern recognition*, 2013.

[90] A. Mishra, K. Alahari, and C.V. Jawahar. Top-down and bottom-up cues for scene text recognition. In *Proc. CVPR*, 2012.

[91] Anand Mishra, Karteek Alahari, and CV Jawahar. Image retrieval using textual cues. In *ICCV*, 2013.

[92] Marius Muja and David G Lowe. Fast approximate nearest neighbors with automatic algorithm configuration. *VISAPP*, 2009.

[93] Gregory K Myers and Brian Burns. A robust method for tracking scene text in video imagery. *CBDAR*, 2005.

[94] Kate Nation. Form-meaning links in the development of visual word recognition. *Philosophical transactions of the Royal Society of London. Series B, Biological sciences*, 364(1536):3665–74, December 2009.

[95] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y. Ng. Reading digits in natural images with unsupervised feature learning. *NIPS Workshop on Deep Learning and Unsupervised Feature Learning*, 2011.

[96] Lukáš Neumann and Jiří Matas. A method for text localization and recognition in real-world images. In *Proc. ACCV*, 2010.

[97] Lukáš Neumann and Jiří Matas. Text localization in real-world images using efficiently pruned exhaustive search. In *Proc. ICDAR*, 2011.

[98] Lukáš Neumann and Jiří Matas. Real-time scene text localization and recognition. In *Proc. CVPR*, 2012.

[99] Lukáš Neumann and Jiří Matas. On combining multiple segmentations in scene text recognition. In *Document Analysis and Recognition (ICDAR), 2013 12th International Conference on*, pages 523–527. IEEE, 2013.

[100] Lukáš Neumann and Jiří Matas. Scene text localization and recognition with oriented stroke detection. In *Computer Vision (ICCV), 2013 IEEE International Conference on*, pages 97–104. IEEE, 2013.

[101] Lukáš Neumann and Jiří Matas. Iwrr keynote talk: Text reading in the wild – how to make it useful? In *Computer Vision-ACCV 2014 Workshops*, 2014.

[102] Lukáš Neumann and Jiří Matas. Real-time lexicon-free scene text localization and recognition. *IEEE Trans. PAMI*, 2015.

[103] Anguelos Nicolaou, Andrew D Bagdanov, Lluis Gomez-Bigorda, and Dimosthenis Karatzas. Visual script and language recognition. In *DAS*, 2016.

[104] Anguelos Nicolaou, Andrew D Bagdanov, Marcus Liwicki, and Dimosthenis Karatzas. Sparse radial sampling lbp for writer identification. *ICDAR*, 2015.

[105] Tatiana Novikova, Olga Barinova, Pushmeet Kohli, and Victor Lempitsky. Large-lexicon attribute-consistent text recognition in natural images. In *Proc. ECCV*, 2012.

[106] WM Pan, Ching Y Suen, and Tien D Bui. Script identification using steerable gabor filters. In *Document Analysis and Recognition, 2005. Proceedings. Eighth International Conference on*, 2005.

[107] Yi-Feng Pan, Xinwen Hou, and Cheng-Lin Liu. Text localization in natural scene images based on conditional random field. In *Proc. ICDAR*, 2009.

[108] Federico Perazzi, Philipp Krähenbühl, Yael Pritch, and Alexander Hornung. Saliency filters: Contrast based filtering for salient region detection. In *CVPR*, 2012.

[109] Marc Petter, Victor Fragoso, Matthew Turk, and Charles Baur. Automatic text detection for mobile augmented reality translation. In *ICCV W.*, 2011.

[110] Trung Quy Phan, Palaiahnakote Shivakumara, Zhang Ding, Shijian Lu, and Chew Lim Tan. Video script identification based on text lines. In *ICDAR*, 2011.

[111] Robert E Schapire and Yoram Singer. Improved boosting algorithms using confidence-rated predictions. *Machine learning*, 37(3):297–336, 1999.

[112] Asif Shahab, Faisal Shafait, and Andreas Dengel. ICDAR 2011 robust reading competition challenge 2: Reading text in scene images. *ICDAR*, 2011.

[113] Nabin Sharma, Sukalpa Chanda, Umapada Pal, and Michael Blumenstein. Word-wise script identification from video frames. In *ICDAR*, 2013.

[114] Nabin Sharma, Ranju Mandal, Rabi Sharma, Umapada Pal, and Michael Blumenstein. Bag-of-visual words for word-wise video script identification: A study. In *Neural Networks (IJCNN), 2015 International Joint Conference on*, 2015.

[115] Nabin Sharmai, Ranju Mandal, Michael Blumenstein, and Umapada Pal. ICDAR 2015 competition on video script identification (CVSI-2015). *ICDAR*, 2015.

[116] Baoguang Shi, Xiang Bai, and Cong Yao. Script identification in the wild via discriminative convolutional neural network. *Pattern Recognition*, 2015.

[117] Baoguang Shi, Cong Yao, Chengquan Zhang, Xiaowei Guo, Feiyue Huang, and Xiang Bai. Automatic script identification in the wild. *ICDAR*, 2015.

[118] Palaiahnakote Shivakumara, Nabin Sharma, Umapada Pal, Michael Blumenstein, and Chew Lim Tan. Gradient-angular-features for word-wise video script identification. In *ICPR*, 2014.

[119] Palaiahnakote Shivakumara, Zehuan Yuan, Danni Zhao, Tong Lu, and Chew Lim Tan. New gradient-spatial-structural features for video script identification. *CVIU*, 2015.

[120] Ajeet Kumar Singh, Anand Mishra, Pranav Dabral, and CV Jawahar. A simple and effective solution for script identification in the wild. In *DAS*, 2016.

[121] Ray Smith. An overview of the tesseract OCR engine. In *icdar*, pages 629–633. IEEE, 2007.

[122] Ray Smith. Limits on the application of frequency-based language models to OCR. In *Document Analysis and Recognition (ICDAR), 2011 International Conference on*, pages 538–542. IEEE, 2011.

[123] A Lawrence Spitz. Determination of the script and language content of document images. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 1997.

[124] A Lawrence Spitz and Masaharu Ozaki. Palace: A multilingual document recognition system. In *Document Analysis Systems*, 1995.

[125] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 2014.

[126] Kazutaka Takeda, Koichi Kise, and Masakazu Iwamura. Real-time document image retrieval on a smartphone. In *DAS*, 2012.

[127] TN Tan. Rotation invariant texture features and their use in automatic script identification. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 1998.

[128] Jasper RR Uijlings, Koen EA van de Sande, Theo Gevers, and Arnold WM Smeulders. Selective search for object recognition. *IJCV*, 2013.

[129] JRR Uijlings, KEA van de Sande, T Gevers, and AWM Smeulders. Selective search for object recognition. *International journal of computer vision*, 104(2):154–171, 2013.

[130] Ranjith Unnikrishnan and Ray Smith. Combined script and page orientation estimation using the tesseract OCR engine. In *MOCR*, 2009.

[131] Koen EA Van de Sande, Jasper RR Uijlings, Theo Gevers, and Arnold WM Smeulders. Segmentation as selective search for object recognition. In *Computer Vision (ICCV), 2011 IEEE International Conference on*. IEEE, 2011.

[132] A. Vedaldi and B. Fulkerson. VLFeat: An open and portable library of computer vision algorithms. http://www.vlfeat.org/, 2008.

[133] Paul Viola and Michael Jones. Rapid object detection using a boosted cascade of simple features. In *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, volume 1, pages I–511. IEEE, 2001.

[134] B Waked, S Bergler, CY Suen, and S Khoury. Skew detection, page segmentation, and script classification of printed document images. In *Systems, Man, and Cybernetics, 1998. 1998 IEEE International Conference on*, 1998.

[135] Kai Wang, B. Babenko, and S. Belongie. End-to-end scene text recognition. In *Proc. ICCV*, 2011.

[136] Kai Wang and Serge Belongie. Word spotting in the wild. In *Proc. ECCV*, 2010.

[137] Tao Wang, David J. Wu, Adam Coates, and Andrew Y. Ng. End-to-end text recognition with convolutional neural networks. In *Proc. ICPR*, 2012.

[138] Christian Wolf and Jean-Michel Jolion. Object count/area graphs for the evaluation of object detection and segmentation algorithms. *IJDAR*, 2006.

[139] Sally L Wood, Xiaozhong Yao, Kanthimathi Krishnamurthi, and Laurence Dang. Language identification for printed text independent of segmentation. In *Image Processing, 1995. Proceedings., International Conference on*, 1995.

[140] Yi Wu, Jongwoo Lim, and Ming-Hsuan Yang. Online object tracking: A benchmark. In *CVPR*, 2013.

[141] Bangpeng Yao, Gary Bradski, and Li Fei-Fei. A codebook-free and annotation-free approach for fine-grained image categorization. In *CVPR*, 2012.

[142] C. Yao, X. Bai, and W. Liu. A unified framework for multi-oriented text detection and recognition. *TIP*, 2014.

[143] Cong Yao, Xiang Bai, Wenyu Liu, Yi Ma, and Zhuowen Tu. Detecting texts of arbitrary orientations in natural images. In *CVPR*, 2012.

[144] Cong Yao, Xiang Bai, Wenyu Liu, Yi Ma, and Zhuowen Tu. Detecting texts of arbitrary orientations in natural images. In *Proc. CVPR*, 2012.

[145] Qixiang Ye and David Doermann. Text detection and recognition in imagery: A survey. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 37(7):1480–1500, 2015.

[146] X. Yin, X. Yin, K. Huang, and H. Hao. Robust text detection in natural scene images. *IEEE Trans. PAMI*, 2013.

[147] Sonia Yousfi, Sid-Ahmed Berrani, and Christophe Garcia. Alif: A dataset for arabic embedded text recognition in tv broadcast. In *Document Analysis and Recognition (ICDAR), 2015 13th International Conference on*, 2015.

[148] Ali Zandifar and Antoine Chahine. A video based interface to textual information for the visually impaired. In *Proceedings of the 4th IEEE international Conference on Multimodal interfaces*, page 325. IEEE Computer Society, 2002.

[149] Dengsheng Zhang and Guojun Lu. Evaluation of mpeg-7 shape descriptors against other shape descriptors. *Multimedia Systems*, 2003.

[150] Jing Zhang and Rangachar Kasturi. Extraction of text objects in video documents: Recent progress. In *DAS*, 2008.

[151] C Lawrence Zitnick and Piotr Dollár. Edge boxes: Locating object proposals from edges. In *ECCV*, 2014.