Republic of Tunisia, Ministry of Higher Education and Scientific Research
University of Sfax, Faculty of Economics and Management of Sfax

DETECTION AND RECOGNITION OF ARTIFICIAL TEXT IN ARABIC NEWS VIDEOS

THESIS

presented to the Economic, Management and Computer Sciences Doctorate School of Sfax

in consideration for the award of the academic grade of

Doctor of Philosophy

SPECIALTY: Computer Science

by

Oussama Zayene

Master degree in Intelligent and Communicating Systems, ENISo

Laboratory of Advanced Technology and Intelligent Systems (LATIS), National Engineering School of Sousse (ENISo), Tunisia

Committee in charge:

PRESIDENT:	**Mr Abdelmajid Ben Hamadou**	Prof., University of Sfax
EXAMINER:	**Mr Slim Kanoun**	Prof., University of Sfax
REVIEWER:	**Mme Yosra Ben Jemaa**	Prof., University of Sfax
REVIEWER:	**Mr Abdel Belaid**	Prof., University of Nancy II
CO-SUPERVISOR:	**Mr Jean Hennebert**	Prof., Hes-So//Fribourg
CO-SUPERVISOR:	**Mr Rolf Ingold**	Prof., University of Fribourg
THESIS DIRECTOR:	**Mme Najoua Essoukri Ben Amara**	Prof., University of Sousse

2017-2018

Acknowledgements

This thesis is a part of a join collaboration between the Laboratory of Advanced Technology and Intelligent Systems (LATIS) from the National Engineering School of Sousse, The research Institute for Complex Systems (ICoSys) from the Hes-So//Fribourg and the Document, Image and Voice Analysis (DIVA) Group from the University of Fribourg.

This thesis owes its existence to the help and support of many people.

First of all, I would like to express my appreciation and gratitude to my supervisor Prof. Najoua Essoukri Ben Amara, head of the LATIS lab. I am very grateful for her valuable advice and permanent guidance throughout my PhD journey.

I am also very thankful to Prof. Rolf Ingold, head of the DIVA Group, and Prof. Jean Hennebert, co-director of the ICoSys, for their hospitality and support during my research internships in their labs and for providing me excellent opportunities to excel in several aspects.

I would also like to thank Dr. Sameh Masmoudi Touj for her helpful suggestions and feedback on my PhD work.

I would like to thank Mr Abdelmajid Ben Hamadou, professor at the university of Sfax, for honoring me by presiding my jury. I would also thank the board of reviewers who took the time to evaluate my thesis, Mrs Yousra Ben Jemaa, professor at the university of Sfax, and Mr Abdel Belaid, professor at the university of Nancy 2. Their comments were very useful. My thanks go as well to Mr Slim Kanoun, professor at the university of Sfax, for accepting to be a member of my thesis committee as an examiner of this work.

I also would like to thank the Tunisian Television (TT) "التلفَزَة التونسية", in particular Mrs. Kaouthar Hamed (Vice-director of the Broadcast Archive department), for the valuable discussions and for making their archive available to us.

In the past few years, I had many discussions with colleagues from LATIS and DIVA and got a lot of help from them; in particular Fouad Slimane, Mathias Seuret, Soulayman Chouri and Soumaya Essefi, with whom I closely collaborated. Other colleagues also helped me in different aspects of the research. They are Ines Baccouche, Micheal Baechler, Kai Chen, Andreas Fischer and Mourad Khayati. I also enjoyed other activities with them, including lunches and hikings. I am thankful as well to secretaries of LATIS and UNIFR for their kind support.

Furthermore, I would like to thank Mrs. Neila Letaïef Harbi, Director of Student Affairs at University of Sousse, for her support and encouragement throughout my academic career.

Last but not least, I wish to express my profound gratitude to my family. Words cannot describe how grateful I am to my dearest mother and father. Thanks for your unconditional love, support and understanding. I am also very grateful to my sisters and to my friends Hamza and Alaeddine for their help and encouragement during this long journey.

Sousse, Tunisia

Oussama Zayene أسَامة زيَان

الملخص

يُعد النص المعروض في فيديوهات الاخبار من المعلومات المهمة الواصفة لمحتوى الفيديو. يعطي هذا النص، مثلا، معلومات موجزة ودقيقة حول هوية المتكلم في الشاشة أو حول مكان وزمان وقوع أحداث معينة.

لذا، بمجرد إستخراج الصُور النصية (Text image) والتعرف عليها، يمكننا استخدامها ككلمات مفاتيح أثناء البحث في مخازن الفيديوهات الالكترونية، أو ككلمات دالة تساعد على فهرسة وأرشفة النشرات الإخبارية بصورة شبه آلية في قواعد البيانات الخاصة بها. غير أن عملية استخراج النص من مقاطع الفيديو ليس بالمهمة السهلة، نظرا لوجود العديد من التحديات مثل اختلاف النصوص وتنوعها من حيث الحجم والخط واللون . كما تتميز صُور الفيديو بدقتها المنخفضة (Low resolution)، وباحتوائها على العديد من الاجسام المتشابهة مع شكل النص، كالأسوار والنوافذ.

نظراً لأهمية هذه النصوص قام العديد من الباحثين، على مدى العقدين الماضيين، باقتراح طرق وخوارزميات مختلفة للتحقق من وجود النصوص داخل الملف (صورة أو فيديو) و إستخراجها ثم التعرف عليها بإعتماد نظام القارئ الآلي للحروف (OCR).

لقد ركزت جل الطرق المقترحة على لغات قليلة مثل اللاتينية والصينية، أما فيما يخص اللغة العربية المستخدمة من قبل مليار ونصف شخص حول العالم، يعتبر عدد الأبحاث المنجزة حتى الآن قليل جدا على الرغم من وجود العديد من القنوات الإخبارية العربية وحاجتها الماسة لهذه الخوارزميات للتعامل بطريقة فعّالة مع مخازنها الإلكترونية التي لا ينفك حجمها عن الزيادة كل يوم.

تساهم هذه الاطروحة في هذا المجال من البحوث من خلال تطوير طرق جديدة لاكتشاف أماكن النصوص العربية المضمنة في مقاطع الفيديوهات الإخبارية والتعرف عليها آلياً.

نقدم في المرحلة الأولى من الأطروحة طريقة مبتكرة لتحديد أماكن تواجد النصوص العربية في الصور المستخرجة من الفيديوهات الاخبارية. تتكون هذه الطريقة من جزأين رئيسيين. يعتمد الأول على خوارزمية متكونة من ثلاثة مراحل رئيسية : استخلاص مرشحات النص باعتماد تقنية (SWT) وتصفيتها ثم تجميعها عن طريق تطبيق مجموعة من القيود الهندسية. أما الثاني فيعتمد بالأساس على نوعية خاصة من الشبكات العصبية الاصطناعية (Convolutional Auto-Encoders) لاستخراج الواصفات (Features) بطريقة أوتوماتيكية، ثم استخدامها كمدخلات للمصنف (SVM) لتمييز النص عن بقية المحتوى.

في المرحلة الثانية من هذه الاطروحة، نعرض النظام المقترح للتعرف الآلي على الصور النصية المستخرجة، قصد تحويلها إلى نصوص إلكترونية قابله للتعديل. نعتمد في هذا العمل على نوعية حديثة من الشبكات العصبية الاصطناعية ذات التعلم العميق (Deep LSTM) مما يمكننا من تجنب تقسيم (Segmentation) الصور النصية إلى كلمات وأجزاء الكلمات، الامر الذي عادة ما يمثل تحدي للباحثين، ذلك لطبيعة النصوص العربية المعتمدة على الاجزاء المتصلة، مع إمكانية وجود إلتصاقات (Ligatures) بين الأجزاء المختلفة المكونة للنص، مثال تقاطع حرفين

أو أكثر، و فراغات داخل الجزء المتصل بذاته.
لتدريب الخوارزميات المقترحة وتقييم فعاليتها قمنا بإنشاء قاعدة بيانات تحمل إسم AcTiV. تتكون هذه الاخيرة من 189 فيديو إخباري وقع تسجيلهم بطريقة دورية من أربعة قنوات تلفزية عربية على مدة 3 سنوات، مع الحرص على احتواء هذه الفيديوهات لعدد كبير من النصوص العربية بمختلف أشكالها وميزاتها. تحتوي AcTiV أيضا على مجموع 10415 صورة لنصوص عربية وقع استخراجها من الفيديوهات المسجلة. وقد تم تطوير نظام أو إطار عمل مصغر شبه تلقائي (Semi-automatic annotation Framework) لمساعدتنا في إستخراج الصور النصية مع ما يقابلها من النص (Ground-truth) بأقل وقت ممكن وكفاءة عالية. كما قمنا أيضا بتطوير نظام (Evaluation tool) لقياس نسب الدقة (precision) والاعادة (recall) في الانظمة المقترحة لاكتشاف مواقع النصوص.
يجدر الإشارة إلى أن AcTiV وقع استعمالها في مسابقتين عالميتين سنتي 2016 و2017 لاختبار نجاعة الأنظمة المشاركة ومقارنتها ببعضها.

الكلمات المفتاحية: إستخراج النصوص العربية من الفيديوهات، التعرف التلقائي على النصوص العربية، الشبكات العصبية الاصطناعية ذات التعلم العميق، قاعدة البيانات AcTiV.

Abstract

TV news are important sources of information for most people. They allow a better understanding of the social and political events punctuating our everyday life. Today, we can save big amounts of digital news videos thanks to the availability of low-cost mass storage technology. As video archives are growing rapidly, making manual video annotation impractical, the need for efficient indexing and retrieval systems is evident. Text displayed in news video is one of the most important high-level information of video content. Actually, it can be used as powerful semantic clues for automatic broadcast annotation. Nevertheless, extracting text from videos is a non-trivial task due to many challenges like the complexity of backgrounds and the variability of text regions in scale, font, color and position. Over the past two decades, interest in this area of research has led to a plethora of text detection and recognition methods. So far, these methods have focused only on few languages such as Latin and Chinese. For a language like Arabic, which is used by more than one billion people around the world, the literature is limited to very few studies.

This thesis aims to contribute to the current research in the field of Video Optical Character Recognition (OCR) by developing novel approaches that automatically detect and recognize embedded Arabic text in news videos.
We introduce a two-stage method for Arabic text detection in video frames. In the first stage, which represents the CC-based detection part of this method, text candidates are firstly extracted, then filtered and grouped by respectively applying the Stroke Width Transform (SWT) algorithm, a set of heuristic rules and a proposed textline formation technique. In the second stage, which represents the machine-learning verification part, we make use of Convolutional Auto-Encoders (CAE) and Support Vector Machines (SVM) for text/non-text classification.

For text recognition, we adopt a segmentation-free methodology using multidimensional Recurrent Neural Networks (MDRNN) coupled with a Connectionist Temporal Classification (CTC) decoding layer. This system includes also a new preprocessing step and a compact representation of character models. We aim in this thesis to stand out from the dominant methodology that relies on hand-crafted features by using different deep learning methods, i.e. CAE and MDRNNs to automatically produce features.

Initially, there has been no publicly available dataset for artificially embedded text in Arabic news videos. Therefore, creating one is unquestionable. The proposed dataset, namely AcTiV, contains 189 video clips recorded from a DBS system to serve as a raw material for creating 4,063 text frames for detection tasks and 10,415 cropped text-line images for recognition purposes. AcTiV is freely available for the scientific community. It is worth noting that the dataset was used as a benchmark for two international competitions in conjunction with the ICPR 2016 and ICDAR 2017 conferences, respectively.

Keywords: AcTiV dataset, Arabic Video Text Detection, SWT, Auto-Encoders, Arabic Video Text Recognition, MDRNN, CTC layer, OCR

Contents

Chapter 1

Introduction

Among the pattern recognition fields, automatic text recognition, known as Optical Character Recognition (OCR) has been widely studied for its prominent position in our everyday life. The aim of this research area is to design a system that converts text images to readable text codes. OCR has a long history of research that started from printed character recognition, extended to handwriting character recognition, and later to printed document recognition, and finally evolved to handwriting document recognition. By the 1960's, OCR technology had found applications for automated data processing in several industries including government, banking and mail. In the middle of the 80s, Toshiba commercialized the first world's OCR technology able to read Chinese characters. Thus, great progress has been made in processing printed/handwriting text against clean background. Today, OCR is considered as a mature technology. There are several commercial products like ABBYY [1], OmniPage [2] and Tesseract [3], which have demonstrated successes in large-scale book scanning.

Recently, embedded texts in videos and natural scenes have received increasing attention as they often give crucial information about the media content. For instance, text captions in news videos can provide concise information about the 'when', 'where' and 'who' elements in relation to the current content. Sometimes this information is not present in the audio or it cannot be acquired through other video understanding methods. Detecting and recognizing text in videos, often called Video OCR, is an essential task in a lot of applications like content-based multimedia retrieval. Actually, broadcast news and public-affairs programs represent a prominent source of information that provides an overview of what is happening at local and world levels. The analysis of public newscast by national and foreign news TV channels is of capital importance for media analysts in several domains such as politics and security. Nowadays, TV newscasters archive a tremendous number of news videos thanks to the rapid progress in the mass storage technology. As the archive size grows rapidly, the manual annotation of huge multimedia databases becomes impractical. This situation creates an urgent need for efficient indexing and retrieval algorithms.

Text displayed in news videos is one of the most important high-level information of video content. Most videos contain two kinds of text. The first type is caption text, which is

[1] http://finereader.abbyy.com/
[2] https://nuance.com/print-capture-and-pdf-solutions/optical-character-recognition/omnipage.html
[3] http://code.google.com/p/tesseract-ocr/

Figure 1.1: Frame samples from different TV channels depicting typical characteristics of artificial text.

artificially superimposed on the video during the editing process, as shown in Figure 1.1, where the video frames include caption/artificial/superimposed text. This text can provide a brief and direct description of video content (e.g. subtitles, location, event information, sports scores, etc) and hereby suitable for indexing and retrieval. The second type is scene text, which is naturally recorded as a part of scene during video capturing, such as traffic signs, shop names, and text on T-shirts. As depicted in Figure 1.2, scene text mostly appears accidentally and is seldom intended.

Figure 1.2: Examples of scene text video frames.

It is evident that applying a conventional OCR system for video frames leads to poor recognition rates due to the limitations of such systems and the nature of video content [CO05]. Compared to scanned documents, text detection and recognition in video frames are more challenging. The major challenges are:

- Text pattern variability: Text in videos mostly has an unknown font-size and font-family, various positions and differ in color and alignment even within the same TV channel.

- Background complexity: Backgrounds are cluttered with noise and blur. There are objects that have a similar appearance with video text, such as bricks and foliage, and some objects that own similar texture characteristics with video text like fences or stripes of clothes.

- Video quality: The acquisition conditions of videos like compression artifacts, low resolution, distortions and degradation make the task harder.

The recognition of Arabic text for indexing Arabic documents has recently become a compelling research domain. Widely used, Arabic represents the official language of 22 countries, the native language of over 280 million people residing in the Arab World, and the liturgical language of over 1.5 billion Muslims around the world. Behind this huge population, more images and videos are being collected and stored than ever before, especially with the significant changes and big events during the last seven years of the "Arab Spring", referred to as a revolutionary wave of both violent and non-violent protests, coups and civil wars in North Africa and the Middle East, which began on December 2010 in Tunisia. Subsequently, the contemporary mass of multimedia documents resulting from the widespread use of digital cameras and video recorders is increasing day after day at a rapid pace, and the various amount of visible text has the potential to surpass all previous scanned book sources. Thus, the ability to automate the interpretation of graphically-embedded Arabic texts will have a broad range of benefits. Compared to Latin text, the Arabic one has special characteristics:

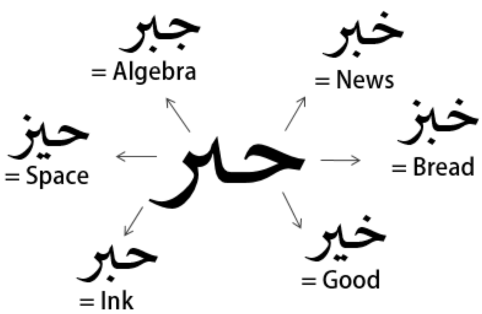


Figure 1.3: Impact of dots on a basic form of an Arabic word: A sample word that leads to six different ones.

- It is cursive with high connectivity between characters; i.e., most of them have a right and/or left connection point linked to the baseline.
- In the Arabic alphabet, 22 out of the 28 letters have four shapes each (word-initial, -medial, -final and -isolated), and six have two shapes each (final and isolated).
- Arabic characters may have exactly the same shape and are distinguished from each other only by a diacritic mark, which may appear above or below the main character such as letters *Baa* (ب), *Taaa* (ت) and *Thaa* (ث). These diacritics are normally a dot, a group of dots, a *Hamza* (ء) or a *Tild* (˜). It is worth noting that any deletion or erosion of these diacritic marks results in a misrepresentation of the character. Hence, any binarization algorithm needs to efficiently deal with these dots so as not to change the identity of the character. A typical example is illustrated in Figure 1.3.
- The spaces between parts of Arabic words are not uniform and vary in size, making ambiguities to distinguish between stroke ends or word ends in the segmentation phase.
- Arabic has several standard ligatures formed by combining two or more letters; e.g., *LaamAlif* (لأ), a combination of *Laam* (ل) and *Alif* (ا).

The first published work on Arabic OCR dates back to 1975, and was by Nazif [Naz75]. Since then, several techniques have been proposed for printed and handwriting Arabic text

recognition, and have acquired great improvement [AB96, LG06, MEA12, KEBE15, KEBE16]. A lot of progress of such methods has been triggered thanks to the availability of benchmarking databases [PMM+02, SIK+09, MKKEA12, MAAK+14] and the organization of international competitions [MEA08, EAMKA09, KTA+11, SKEA+11, SAM+14].
Although more than three decades have passed, there has been a lack in the analysis and recognition of Arabic video text. Despite the presence of several Arabic news channels with very high viewing rates in the Arabic world and outside of it, there have been only very few attempts to develop detection and recognition systems for overlaid text in Arabic news videos [HAV+12, YBG14, SWTF16]. So far, most of these systems have been tested on private datasets with non-uniform evaluation protocols, which makes objective comparison and scientific benchmarking rather impractical. In this thesis, we aim to fill the aforementioned gap by providing a standard dataset and accurate methods for detecting and recognizing Arabic video text. Technically speaking, the goal of text detection is to identify candidate text regions in a video frame by filtering out non-text objects. The target of text recognition is to transform already detected text regions (i.e. pixels) into readable text codes.

Contributions:
The main contributions of this work are as follows:

1. Detailed study about text detection and recognition in natural scenes and videos in terms of existing datasets and proposed methods.

2. Development of a hybrid system for Arabic video text detection based on a modified version of the Stroke Width Transform (SWT) [EOW10] for text component extraction, a new grouping procedure for textline construction, a Deep Convolutional Auto-Encoders for unsupervised feature-learning and an SVM classifier for text/non-text discrimination.

3. Elaboration of a semi-automatic framework for Arabic text annotation in news video. The framework includes two different levels of annotation: a global (manual) level, which concerns the entire video clip and a local (automatic) level for any specific frame extracted from that video.

4. Development of an evaluation tool for text detection tasks. The tool takes as input a video file, a frame or a set of frames to assess the performance of a detection system in terms of precision recall and F-score metrics.

5. Development of an innovative text recognition system based on the combination of a new preprocessing step, a compact representation of character models, and the use of Multidimensional Recurrent Neural Networks (MDRNNs) coupled with a Connectionist Temporal Classification (CTC) layer to recognize textlines without any prior segmentation or binarization step.

6. Design of a large dataset, namely AcTiV, of news videos, hand-selected text frames and cropped textline images. These multimedia documents are collected from two sources, a satellite receiver and the YouTube website (7%). The video clips are from four TV

news channels and are in three different resolutions. AcTiV represents the first publicly available annotated dataset for Arabic Video OCR systems and was used as a benchmark in two previous international competitions.

Figure 1.4 illustrates the timeline of our research work.

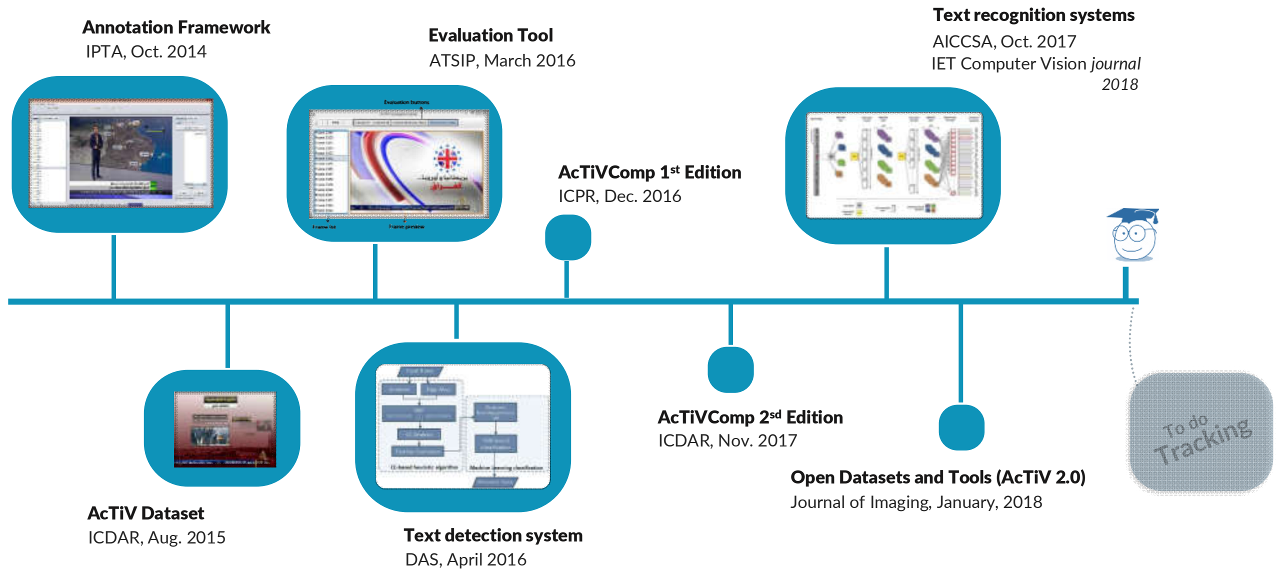


Figure 1.4: Timeline of the present thesis

Report outline:

The remaining parts of this thesis are structured as follows:

Chapter 2 (State of the Art in Text Detection and Recognition) gives a survey of text detection in videos and images in terms of method category, underlying steps, and used features / classifiers, followed by a review of some related work on text recognition with a focus on video and scene Arabic text.

Chapter 3 (Proposed Dataset and Experimental Settings) presents a short survey about the existing databases dedicated to scene and video text analysis, followed by a detailed description of the proposed dataset in terms of characteristics, statistics and annotation guidelines. This chapter also sums up the proposed annotation and evaluation tools and defines the suggested evaluation protocols.

Chapter 4 (Text Detection by SWT and Auto-encoders) describes the proposed text detection schemes. Furthermore, a detailed performance evaluation is discussed, and the obtained results are compared to other recently published studies.

Chapter 5 (Text Recognition by MDLSTM Networks) presents the suggested system for Arabic video text recognition. Several experiments are meant to analyze the impact of the proposed preprocessing step and the effect of the model sets' choice. The chapter presents also a comparison of our results with some state-of-the-art methods.

Chapter 6 (Conclusions and Future Work) includes the concluding remarks of the proposed thesis. In particular, it summarizes the main contributions and outlines the potential research directions in the future.

Chapter 2

State of the Art in Text Detection and Recognition

2.1 Introduction

Since the 80's research in OCR systems has been an active domain in computer vision and pattern recognition communities. Prior studies have mainly focused on systems operating on scanned documents. Recently, a great progress has been made in other fields of research such as text recognition in historical documents, in scene images and in videos (Figure 2.1).

Embedded text in videos represents a rich source of information for automatic video analysis and indexing. However, this kind of text is more difficult to extract and recognize than the one in scanned documents. This is due to many challenges like the complexity of backgrounds and the variability of text patterns (e.g., size, color, font and position). Hence, several approaches have been proposed to tackle these problems. This chapter presents an overview of text detection and recognition methods in images and videos.

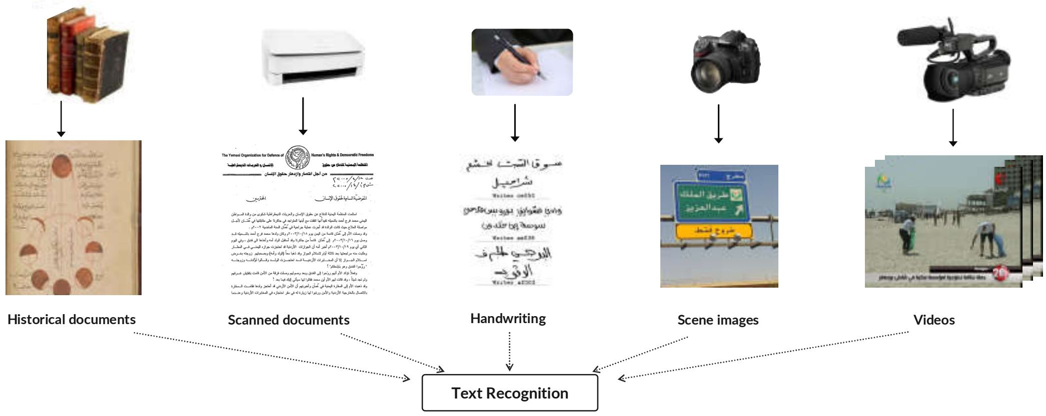


Figure 2.1: Some examples of text recognition tasks

A Video OCR system is generally composed of two main phases: text detection, which may include the localization and tracking of video text regions, and text recognition, which may include the extraction, segmentation and recognition of already detected text regions. As depicted in Figure 2.2, the two first tasks consist in locating text regions in video frames

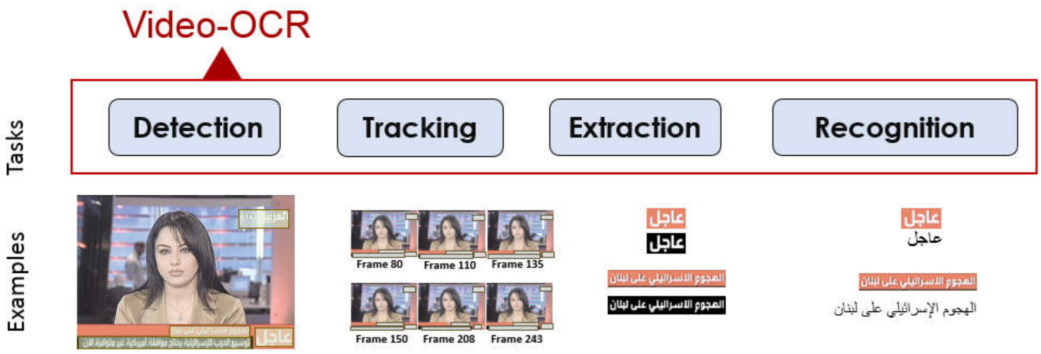


Figure 2.2: Main steps of a Video OCR system

and generating the bounding boxes of text lines as an output. Text extraction operates to extract text pixels and remove background ones. The recognition task converts image regions into text strings.

Decomposing the Video OCR problem into text detection and text recognition dates back to the 90s. Researchers have subsequently worked solely on text detection [LPTL14] or text recognition [RSR+15], or on combining both of them in an end-to-end system [YD15].
These fundamental tasks have been differently referred to in the literature according to the category of the processed text. For instance, several methods have included detection of artificial text in videos [WJW04], and recognition of video captions [SKH+99, TGLZ02], which narrows the focus to superimposed video text analysis. Some others have included scene text detection, and scene text recognition in the wild [ZYB16], which mainly work on natural scene images. This choice has been also driven by the targeted application; e.g., the recognition of video captions has enhanced multimedia retrieval systems, and the recognition

of text on maps and houses has been applied in assistive navigation and automatic geocoding systems [MTC+14].
Despite these differences, these methodologies have shared similar points in terms of used techniques, features and classifiers. Thus, we survey in the following previous studies in relation to text detection and recognition in multimedia documents in general, with a focus on Arabic video/scene text.

2.2 Text detection in images and videos

Recently, several approaches have been proposed to detect text in videos and natural scene images [LPTL14, YD15, YZTL16]. These approaches are generally grouped into texture-based methods, connected component (CC)-based methods or a combination of them, namely hybrid methods.

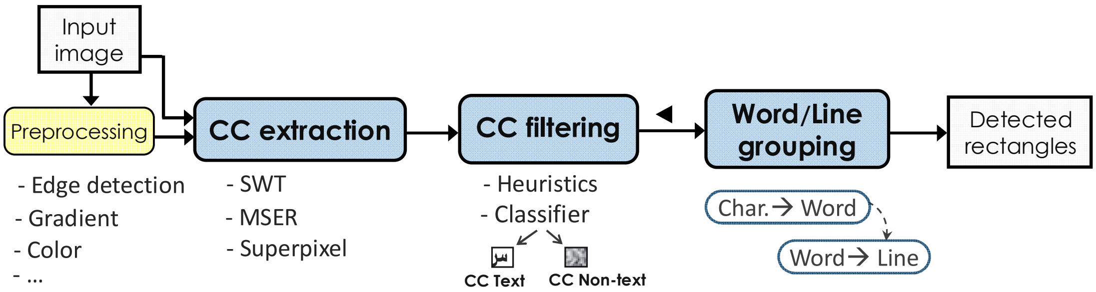


Figure 2.3: Flowchart of a typical CC-based text detection method. Yellow rectangles correspond to optional stages. The '◄' symbol indicates that the order of these two steps can be reversed.

2.2.1 Connected component-based methods

These methods work in a bottom-up manner by grouping neighboring pixels into successively larger components through a variety of ways such as color clustering, edge-based analysis and gradient-based analysis. Non-text components are then filtered out using heuristic rules [CTS+11, XXS14, ZL15, SWTF16] or trained classifiers [HLYW13, THA15, WFCL17]. In other words, these methods focus on the following problems:

- Problem (A): Text-like components extraction.
- Problem (B): CC analysis and linking.
- Problem (C): Non-text CC filtering.

Figure 2.3 depicts the flowchart of a typical CC-based method. It is worth noting that several researchers infer text lines from CCs before performing the filtering stage. Some others have applied the filtering stage before and after the grouping of CCs; that is, they filter out twice the non-text objects at the component level and at the word/line level, respectively, by using trained classifiers or heuristic checks. For example, Yao *et al.* [YBL+12] proposed a two-level filtering scheme for scene text detection. The first filter employed a set of geometric rules and the second one ran a Random Forest (RF) classifier on a set of component-level features. After linking character candidates, the RF classifier was again used with 11 chain-level features to reject false positive lines.

To better understand this category of methods, an outline of its main steps (Figure 2.3) is presented in the following paragraphs.

Preprocessing

As mentioned before, the video environment has many problems to deal with in regards to background complexity, low contrast, color bleeding, etc. Therefore, several researchers have applied a preprocessing step prior to CC extraction in order to enhance the input image quality.
To cope with blurred edges, Chen *et al.* [CTS+11] proposed to remove MSER [1] pixels, located outside the boundary of Canny edges. This was done by pruning MSER along the gradient directions, calculated from the input gray-scale image, (blue arrows in Figure 2.4).

Figure 2.4: Edge-enhanced MSER, from [CTS+11]. (a) Detected MSER for blurred text. Canny edges are shown in red lines, and blue arrows indicate gradient directions. (b) MSER after pruning along the gradient.

To address the above problems, Tsai *et al.* [TPB+] performed a combination of judicious parameter selection and a computationally efficient multi-scale analysis of MSER regions. In the same context, Li *et al.* [LJSvdH14] put forward an edge-preserving algorithm. Given an intensity image I smoothed by the guided filter [HST13], a new image I^* was computed based on its normalized gradient amplitude map (GAM), denoted by ∇I (Equation (2.1)).

$$I^* = I \pm 0.5\text{x}\nabla I \tag{2.1}$$

where $+$ was chosen to detect dark characters on light background, and $-$ was for detecting light characters in dark background. Zhuge and Lu [ZL15] exploited the GAM to overcome the problems of color bleeding and fuzzy boundaries. Furthermore, they applied a top/bottom-hat morphological filtering, prior to MSER treatment, to avoid background noise and enhance the contrast between text and background.

Ghanei and Faez [GF15] exploited the weighted median filtering (WMF) as a nonlinear

[1] MSER, for Maximally Stable Extremal Regions, is basically a method for blob detection in images, and has been shown suitable for detecting perceptually homogeneous characters in scene images and videos. The next subsection presents more details about it.

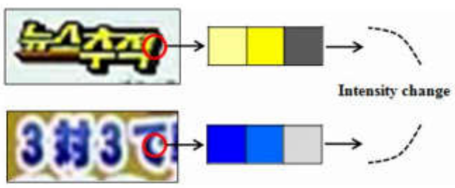
(a) (b) (c)

Figure 2.5: Color-to-gray conversion for a low luminance contrast image, from[GF15]. (a) RGB Image, (b) intensity part of the HSI space and (c) color-contrast preserving decolorization.

edge-preserving smoothing filter and then the color Contrast Preserving Decolorization (CPD) [LXJ12] to make the text detection system more robust for low luminance contrast and poor quality text (see Figure 2.5 for an illustration).
Stroke Width Transform (SWT), as noted and introduced by Epshtein *et al.* [EOW10], is a smart operator that calculates for each pixel the width of the most likely stroke containing the pixel. This algorithm can achieve high precision and recall rates with a very short processing time. However, it is sensitive to the defection of edges. Thus, before performing SWT, several researchers [EOW10, KP10, YQS12] applied a Gaussian smooth filter (Equation 2.2) to increase robustness against fine noise.

$$G(x,y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{2.2}$$

where σ is the standard deviation of the distribution. Felhi *et al.* [FTB12] performed an anisotropic diffusion filtering that smooths away textures whilst retaining sharp edges. Taking advantages from the geometric features revealed by the bandlet transform [MP07], a novel bandlet-based edge detector was introduced by Mosleh *et al.* [MBH13] to enhance the accuracy of SWT that originally uses the Canny edge detector. Xu *et al.* [XXS14] exploited the complementary properties of the gradient-based and smoothness-based edge information for generating high quality edge images and exploited various edge cues in CC analysis to overcome inter/intra-character errors. In [SK17], Shahzad and Khurshid proposed to preprocess the input frame using the YUV color space conversion and an edge sharpening filter.



(a)

Intensity of three consecutive pixels at the boundary

Figure 2.6: Change of intensities in transition region (from [KK09]).

Connected-component extraction

Several techniques have been suggested to extract CCs from video frames and scene images, making use of text characteristics such as color uniformity and gradient distribution. Kim *et al.* [KK09] proposed to detect text in videos by means of a background-text transition map. The idea was to firstly compute the intensity changes for each three consecutive pixels, as shown in Figure 2.6. If the difference between two changes is larger than a predefined threshold, the central pixel will be labeled as a transition pixel. Next, the small gaps between transition pixels were filled to generate CCs. This method may encounter difficulties when the text is multicolored or textured. Yi and Tian [YT11] introduced a color-based partition scheme, which applied a weighted mean-shift clustering in the RGB space to separate text from background pixels, and subsequently generate candidate character components.

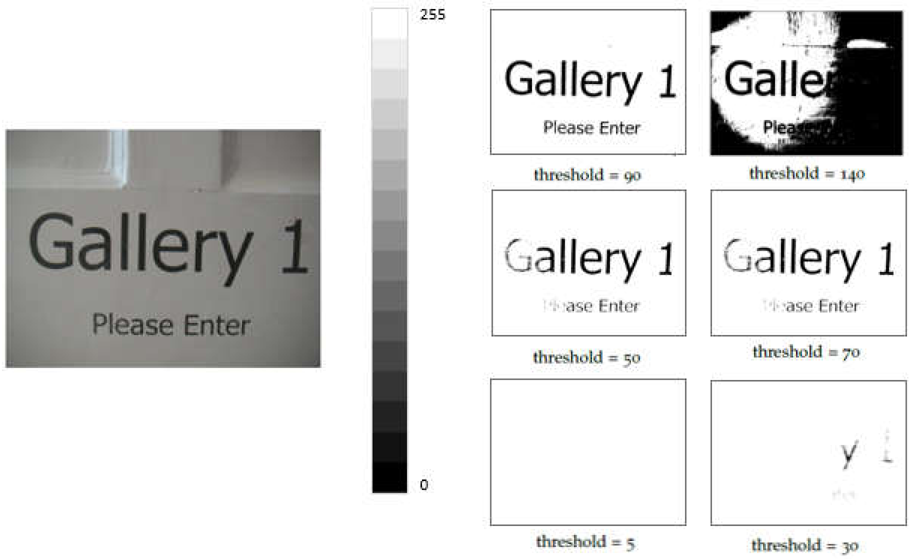


Figure 2.7: MSER detection process. All pixels with an intensity value less than the threshold g are assigned a black color. Note that for g=5, there are no pixels with an intensity value less than five. Subsequently, when g increases, black regions will start to appear. CC region '1' remains constant from g = 50 until g = 90. Such regions will be classified as ER and those ERs with minimal change in area over the range of thresholds are known as MSERs.

Among the recently published CC-based methods we can observe an increasing use of MSER for character candidate extraction. This technique was first introduced by Matas *et al.* [MCUP04] as a blob detection tool for stereo matching. MSERs are regions that are stable across a wide range of thresholds and that are either brighter or darker than all the pixels on their outer boundary. Figure 2.7 explains the process of MSER detection. The grayscale image is thresholded at multiple increasing thresholds. Each thresholded image consists of several CCs that are called an Extremal Region (ER). ERs in images of different thresholds form a parent-child relationship where child-regions are nested in parent regions. Hence, a component-tree is built. For each ER, R_i within the tree, a stability value Ψ is defined as

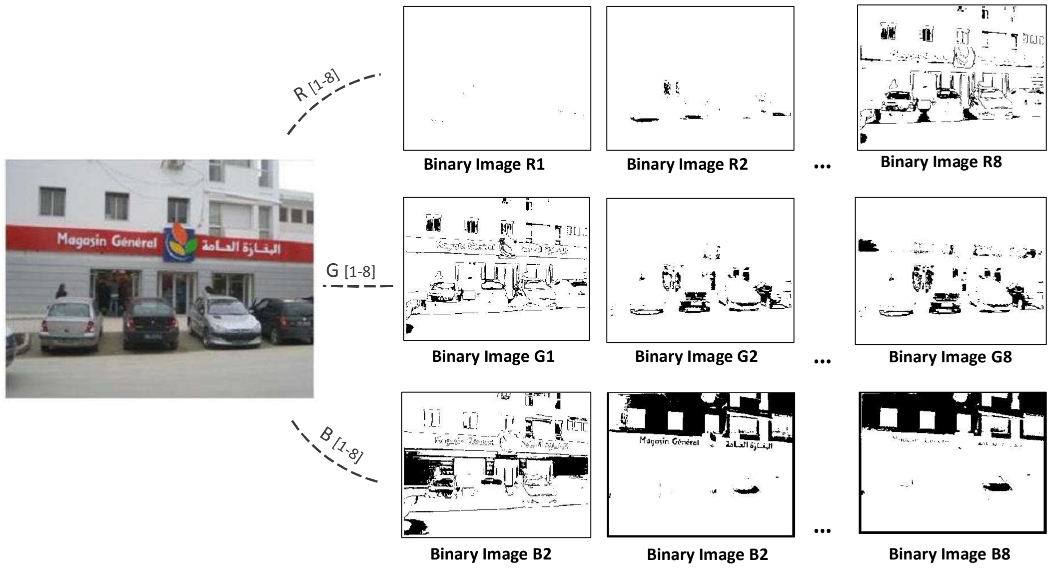


Figure 2.8: Binary map generation. The binary map is generated by assigning the value '1' for the pixels belonging to the interval while fixing the remaining pixel values to '0'.

follows:

$$\Psi(R_i^g) = \frac{|R_j^{g-\triangle}| - |R_k^{g+\triangle}|}{|R_i^g|} \tag{2.3}$$

where $|.|$ represents cardinality, R_i^g is a region obtained by thresholding at a gray value g, and $\triangle$ is a stability range parameter. $R_j^{g-\triangle}$ (respectively $R_k^{g+\triangle}$) is an ER obtained by moving upwards (respectively downwards) in the component-tree from region R_i^g until reaching a region with gray value $g - \triangle$ (respectively $g + \triangle$). ERs that have local minima of Ψ are defined as MSERs.

Neumann *et al.* [NM10] were the first to introduce MSER into the field of text detection. They proposed to extract MSERs from the original image as potential candidate regions, and eliminate invalid candidates using a trained classifier. Chen *et al.* [CTS+11] employed edge-enhanced MSERs to find letter candidates, and geometric filtering as well as stroke width information were used to exclude non-text objects. A similar method was recently proposed by Mansouri *et al.* [MCZ18] but using a baseline estimation technique and some morphological operations to filter out non-text objects from Arabic video frames.

The ICDAR 2013 competition winning approach [YYHH14] utilized a pruning algorithm to select appropriate MSERs as character candidates and a superior AdaBoost classifier to validate true candidates. The effectiveness of MSER was also exploited for video text detection by Jain *et al.* [JPZ+14] and Zhuge *et al.* [ZL15], among others, while Huang *et al.* [HQT14] introduced a novel framework, which exploited geometric grouping over MSER regions and classified the regions using Convolutional Neural Networks (CNN). Despite their success in recent years, the MSER-based methods have several open problems that need to be dealt with. First, these methods are difficult to achieve high text detection accuracies due to their requirement for maximum stability. Second, some text objects are not ERs, whose pixels have either higher or low intensity than their outer boundary pixels, and cannot be extracted by the MSER operator directly. Whereas the second problem is an intrinsic limitation of ER-based approaches, the first one has been addressed by some researchers. In [SHJC15], Sun *et al.* proposed a generalized color-enhanced Contrasting ER (CER), and in [HHQY16], He *et al.* put forward Contrast-Enhancement MSERs (CE-MSERs). Moreover, Cho *et al.* [CSJ16] employed efficient and effective ER tree pruning techniques.

Gaddour *et al.* [GKV16] proposed a region representation derived from MSER to detect Arabic scene text. Instead of relying on a range of unique thresholds, this approach calculated a range of pairs of thresholds for each channel in the RGB color space using the k-means algorithm. This range constructed a set of binary images, each belonging to a color interval [S_i, S_j] (see Figure 2.8). For all CCs of each binary map a two-stage filtering was applied to eliminate non-text CCs. At a later stage, the remained candidates were grouped into textlines through a series of connection rules.

Figure 2.9: Stroke Width Transform. (a) Scene text detection examples from Epshtein's work [EOW10]. (b) Example of SWT computation [YT12].

The stroke serves as a basic element to construct text characters. It is defined as a contiguous zone of text that forms a band of approximately constant width. Therefore, in addition to color uniformity and character alignment, stroke width consistency represents a significant characteristic of text. Based on this observation, a regional stroke width distribution can be utilized to check whether the localized areas contain text or not [SNDC07, DCC+07].

Being one representative approach of the CC-based category, specifically, the Epshtein's SWT method [EOW10] has shown particular effectiveness and computational efficiency in scene text detection. The SWT operator uses the information of stroke edges to extract candidate text components from the input image. The key insight is that letters have roughly parallel sides. The stroke width (SW) is calculated as a distance between two edge pixels with similar gradient magnitudes and opposite gradient directions (Figure 2.9(b)). SWT labels the pixels located inside the torso of a stroke by its width and transforms the input image into a width image (called hereafter SWT map). This algorithm has constituted the basis of a lot of subsequent work [YBL+12, MBH12, HLYW13, KJM13, IP13, XXS14, JJ+15, SX15, FSZ16] in the field of text detection. However, the SWT has its own pros and cons. As mentioned above, the original SWT and CC labelling algorithms were sensitive to edge noise. Furthermore, they are usually insufficient for complicated scenarios like low resolution, cluttered background and cursive and/or interfered text. In order to recover the mismatch between edge points, Huang *et al.* [HLYW13] proposed a new operator based on the SWT, called Stroke Feature Width (SFT). This operator exploited the color consistency and constraint relations of local edge points, yielding to a better component extraction result. In [KJM13], a learning-based framework was proposed to obtain text attention maps from multiple bottom-up saliency features. These text attention maps helped prune the search space for the SWT algorithm and limited its processing only closer to text edges, resulting in more accurate detection performances and efficient computation time. However, this approach was unable to handle regions where the text blended with the background. Su *et al.* [SX15] introduced seed-based SWT for scene text detection. First, seed stroke segments were extracted from the SWT map based on a set of heuristic rules. After that, the defective strokes were recovered with the help of the obtained seed stroke segments, and the initial inaccurate stroke width was consequently rectified. Some other researches [IP13, WPW16, FSZ16] extended the SWT by incorporating color cues of text pixels to achieve a better detection performance even when the Canny detector fails. For example, Feng *et al.* [FSZ16] introduced a multi-scale SWT technique. The multi-scale mechanism first computed five color channels (RGB, Hue and Intensity) from the input image, and then built a scale pyramid for each channel by successively smoothing and downsampling the image with a scale factor of 2/3. SWT was next performed on each level of the pyramid.

CC extraction algorithms may also include local binarization methods [PHL11, CYHL15, SWTF16]. In [SWTF16], Iwata *et al.* introduced an approach for Arabic news text detection. They utilized the Otsu's discriminant analysis technique to binarize the input frame and extract text candidate components. Recently, superpixel-based methods have achieved remarkable success in object detection problems [YYZ+15]. Superpixels, as introduced by Ren and Malik [RM03], are groups of connected and perceptually homogeneous pixels, obtained by over-segmenting the original image. Wang *et al.* [WFCL17] proposed a new superpixel segmentation algorithm based on color and edge information and a single-link clustering algorithm to extract character candidates in scene images.

It is to note that some edge-based detection methods [YT12, YQS12, YSM+15, FSZ16], in which edge components instead of regions are treated as text candidates, can also be categorized as a CC-based method. For instance, in [YT12], a color-pair boundary clustering was firstly performed based on Gaussian Mixture Model (GMM) and Expectation-Maximization (EM) algorithms. As a result, character edges with similar colors were grouped into identical boundary layers. Afterwards, a structural analysis took place by combining a stroke boundary and color assignment to extract character candidates in each boundary layer.

Connected-component filtering

Many components extracted at the previous stage are not part of text. Thus, most of text detection systems use classifiers or perform a set of geometric checks on each CC to filter out non-text objects. In the literature, we find a large number of statistical and geometrical rules based on different extrovert characteristics of text components such as stroke width similarity and color uniformity. In what follows, we present some frequently-used heuristic rules.

1. *Component size:* CCs whose sizes are too large or too small to be readable text should be rejected.
2. *Component position:* Objects located at the border of the image are discarded.
3. *Aspect Ratio (AR):* The ratio $\frac{h(c)}{w(c)}$ of a text CC's bounding box (BB) should be located in a reasonable range. Else, it is regarded as background noise.
4. *Occupation Ratio (OR):* The ratio $\frac{q}{w(c) \cdot h(c)}$ of foreground pixels (e.g. black pixels in the SWT map) to the total pixels of a given CC should be greater than a predefined threshold.
5. *Stroke Width Variance (SWV):* A text candidate is discarded if its SWV, defined by $\frac{\sigma(c)}{\mu(c)}$, exceeds a fixed threshold range, where $\sigma(c)$ and $\mu(c)$ are respectively the mean and standard deviation of the stroke widths in the component c.
6. *Stroke Color Variance (SCV):* The SCV within a true positive component should be less than the half of the average stroke color.

In addition to the above constraints, other heuristics have been proposed to wipe off distinct false candidates. For instance, Chen *et al.* [CTS+11] removed objects that contained a large number of holes, because CCs with many holes were unlikely to be text candidates. To remove false lines from Arabic video frames, Iwata *et al.* [SWTF16] calculated the eccentricity $e = \frac{Perimeter(c)^2}{Area(c)}$ of all CCs in a textline. The line would be removed if the average of e was less than a predefined threshold ($= 30$). Furthermore, crossing counts (number of transitions from white to black pixels) above, in and below the baseline, denoted respectively by N_A, N_I and N_B, are computed and the line would be discarded if the following conditions held: $N_A < N_I \;\; or \;\; N_A < N_B$. Based on the SW consistency of text candidates, Xu *et al.* [XXS14] performed a set of simple rules including the *stroke count ratio*, which represented the number of rays within CC in the SWT process divided by the CC height. Feng *et al.* [FSZ16] designed two novel edge-based heuristics, namely the *stroke pair ratio* $SPR = \frac{N_{esp}}{N_{ep}}$ and the *edge density* $ED = N_{eInCc}$, where N_{esp} was the number of edge pixels which would find their stroke pairs in an edge component, N_{ep} was the number of pixels in an edge, and N_{eInCc} was the number of edge components inside the BB on the stroke map. An edge would be removed if its SPR was less than a predefined threshold T_1. Furthermore, edges inside a BB would be all removed if the CC's ED was greater than a threshold T_2.
Text components are rich of corners, which are uniformly distributed over text regions. Based on this observation, Zhuge *et al.* [ZL15] conducted a corner detection in BBs of binary images, and then counted the number of corners in each BB. If the ratio of this number to the area of the box was below a fixed threshold Thr, than the BB would be considered as a false alarm and discarded from the video frame. In [GKV16], a first filtering was applied for all CCs in the binary map according to a *stability criterion* of embedded text. This criterion tested the surface evolution of a component C_{0i} by varying the two extremities of the corresponding color interval $[S1, S2]$ (see Gaddour *et al.* [GKV16] in the previous subsection), either increasing or reducing it by a predefined factor α. The surface of a text component would remain relatively stable since the text is well contrasted compared to the rest of the image. Then, a second filtering process was performed via a baseline estimation technique and other syntactic rules about AR and ligature count considering the specificities of Arabic script.

These heuristics have proved to be both simple and fast. However, their flexibility to verify text candidates is limited. In other words, if the heuristics are strict, they may fail to preserve text that does not comply with all the rules. If the heuristics are weak conditions (i.e. relaxed), they may introduce false alarms.

On the other hand, several methods have built classifiers to distinguish text/non-text components by combining multiple features. In [YBL+12], the extracted components and lines were respectively verified by two sets of features and a two-level classification scheme based on the RF classifier. The proposed features were specially designed for capturing the intrinsic properties of text, such as stroke width uniformity at the character level, and characters' similarity, in terms of colors, sizes and orientations, at the line level. In [MBH12], an 8-element vector of hand-crafted features was fed to the k-means clustering to identify text components. This vector included the variance V_G of the gradient directions of all edge pixels in one CC, the contrast, the AR, the variance SWV and the median SWM of the stroke widths in one CC, and the skewness of the gradient directions $SK_G = \frac{\mu_3}{\sigma_3}$, where μ_3 and σ_3 denote the third moment about the mean and standard deviation of the gradient directions. In [SWX+13], 13-dimensional features of gradient, stroke width, regularity and occupation were combined to represent each MSER. Such CC-based features were input to the RF classifier to distinguish text and non-text regions. Koo and Kim [KK13] proposed to divide MSER regions into normalized squares, from which the following features were extracted and classified with a multilayer perceptron (MLP): the number of foreground pixels, the number of vertical white-black transitions, and the number of horizontal black-white transitions. The average of the classification responses from all sub-regions was subsequently used for text/non-text classification. Chen *et al.* [CYHL15] utilized a linear SVM trained with a set of 11 features for text/non-text classification in born-digital images. The features were extracted based on the contour, the area, the bounding box, the skeleton and the stroke width of one CC. The remaining CCs underwent textline merging and text/non-text line classification. The linear SVM classifier was again used at the line level. In [KJ11], the difference between middle and upper zones, the distance between two lines, the CC max/min heights and AR as well as other structural features were combined and trained with SVMs.
The standard deviations of colors and compactness, number of character candidates in a region, AR and SWM values of an MSER region were extracted as candidate region features in [YYHI12]. Using these features, an AdaBoost classifier was trained for distinguishing text and non-text regions. Huang *et al.* [HLYW13] applied two RF classifiers at component and line levels, sequentially, to discriminate text regions. The classifiers were built upon two Text Covariance Descriptors (TCDs) that would capture the inherent correlations between multiple features and encode the statistical characteristics of text strokes. Turki *et al.* [THA15] trained a Dynamic Time Warping (DTW) classifier using the HOG and SIFT features to classify MSER regions as text or non-text.

More recent efforts have focused on reducing the amount of hand-crafted features or human-defined heuristics by adopting deep CNN [PYKY16, HHQY16, ZLC+17, WFCL17] to design text detection systems. These deep learning-based approaches usually achieve superior performances over the conventional ones. A CNN model was utilized to filter out non-text components, which were generated by Edgebox detector in [JSVZ16]. Similarly, in [HHQY16], He *et al.* proposed a novel approach for text detection, which integrated an improved CE-MSER operator and a Text-Attentional CNN (TA-CNN) classifier. The CE-MSER detector worked in the front-end to extract text candidates, while the TA-CNN model was employed to correctly identify true text candidates. A similar method was also proposed by Sun *et al.* [SHJC15] but using a Fully Connected Network (FCN). In [PYKY16], a multi-information (multi-channels and multi-scales) MSER fusion algorithm was introduced to extract character candidates, which were then grouped and verified utilizing a hybrid filter with CNN, AdaBoost and Bayesian classifiers. Recently, Wang *et al.* [WFCL17] utilized a new architecture of deep CNNs and a double-threshold strategy to classify text/non-text components that were extracted using the above-mentioned superpixel technique.

Connected-component grouping and textline construction

The challenging question to answer in this stage is how to group adjacent CCs, detected and filtered in the previous steps, into separated meaningful words or lines. The existing methods for CC grouping can be divided into two categories: rule-based [CTS+11, YT11, LL12, BYL13, ZL15] and clustering-based [YYHH14, HYH+16] methods. Based on the assumption that characters in one line typically appear in a linear form and usually have a similar color, height and space between them, several heuristic rules have been commonly applied to connect text components. For instance, two characters were paired in [CTS+11] in case (1) the ratio of their SW medians was lower than a threshold *T1*, (2) their height ratio was lower than a threshold *T2* and (3) they were not very distant. Li *et al.* [LL12] calculated two maps, namely the distance map and the orientation map, by measuring the Euclidean distance D and the orientation angle θ between each pair of CC. When D was lower than a predefined threshold Max_{dist}, these two CCs were labeled as adjacent candidates. Next, θ was checked for each adjacent pair of CCs on the orientation map. Every pair of CCs satisfying this rule was finally checked by a set of similarity constraints concerning height, width, SW mean and intensity. According to their statistical analysis of text strings, Yi *et al.* [YT11] defined four geometrical constraints to decide whether two CCs can be considered as sibling of each other:

- For text strings aligned horizontally, the difference between y-coordinates of the CC centroids should not be greater than T_1 times the height of the higher one, i.e. $|coorY_1 - coorY_2| \leq T_1 \cdot max(h_1, h_2)$
- Two adjacent letters should not be too far from each other, so the distance between two CCs should not be greater than T_2 times the width of the wider one, i.e. $|coorX_1 - coorX_2| \leq T_2 \cdot max(w_1, w_2)$
- The centroid of a CC c_1 should be located between the upper-bound and lower-bound of the other candidate CC c_2, i.e. $coorY_1 > coorY_2 - h_2 \cdot T_3$ and $coorY_1 \leq coorY_2 + h_2 \cdot T_3$
- The color difference between them should be lower than a predefined threshold T_4, i.e. $|cl(c_1) - cl(c_2)| \leq T_4$

A graphical illustration of this grouping method is depicted in Figure 2.10. Similarly, Bai *et al.* [BYL13] applied four constraints based on the color, SW, distance and direction of a pair of characters :

Figure 2.10: (a) Sibling group of CC 'r' where 'B' and 'o' comes from left and right sibling sets respectively. (b) Merging of sibling groups to an adjacent character group (e.g., "Brolly?"). (c) Two detected adjacent character groups [YT11].

- $max((r_1 - r_2), (g_1 - b_2), (b_1 - b_2)) < t_1$
- $|sw_1 - sw_2| < t_2$
- $distance(c_1, c_2) < t_3 \cdot max(h_1, h_2)$
- $angle(c_1, c_2) < t_4$

where $(r, g, b), h, distance(.)$ and $angle(.)$ respectively denote the RGB values, height, distance between the centers of two CCs c_1 and c_2, and the directional angle (between the line connecting CC centers and the horizontal axis).
In [VTPEK16], a raycast-based textline grouping method was proposed, where a horizontal ray was cast to the right, starting from each character region until hitting an other character region (Figure 2.11). If this happens those regions would be grouped and the scan would continue till the ray would exit the image or hit another too faraway region, according to the criteria given by Equation (2.4).

$$v(A, B) = \frac{distance(A_c, B_c)}{min(A_w, B_w)} \tag{2.4}$$

where A_c and B_c are the corresponding region center, and A_w and B_w are the corresponding region widths.

Figure 2.11: Raycast-based text line grouping from [VTPEK16].

To construct a text line given the obtained character' pairs, the previous methods [CTS+11, LL12, BYL13, VTPEK16] were generally inspired by the observation that the centers of the

letters' BBs are usually in a straight line. This worked well for English characters, but did not perform so well for Arabic and Chinese ones, whose centers are not in a straight line. Furthermore, most of these methods assumed that a text character and its siblings have similar sizes and proper distances (see the work of [YT11] in Figure 2.10). This is not the case for cursive scripts like Arabic, which usually has a non-uniform inter/intra-word distance and a variable size of characters' BB.
Zhuge *et al.* [ZL15] suggested to form video textlines (Chinese/Latin) by using a more flexible method. Specifically, the Run Length Smoothing Algorithm (RLSA) was applied on CCs of a binary image, as follows:

$$CCsImg(i,j) = H_{rlsa}(i,j) * V_{rlsa}(i,j) \tag{2.5}$$

where $H_{rlsa}(i,j)$ was acquired by RLSA to merge CCs whose Euclidean distance was less than a fixed threshold in the horizontal direction, and $V_{rlsa}(i,j)$ was acquired by RLSA in the vertical direction.

On the other hand, the clustering-based method presented by Pan *et al.* [PHL11] constructed text lines by minimizing energy functions of a learned distance metric. To group multi-oriented text components, Yao *et al.* [YBL+12] made use of a greedy agglomerative clustering method, in which neighboring pairs would be grouped together if their average alignment was under a certain threshold. Yin *et al.* [YYHH14] proposed to group characters into text candidates by using the single-linkage clustering algorithm, where the distance weights and clustering thresholds were automatically learned by a self-training distance metric learning algorithm. The merging process was treated as an assignment problem on top of a character component graph in [HYH+16], and the best assignment without conflicts was chosen based on scores calculated using several textline features concerning the color histogram of both x and y regions (Equation 2.6), their height ratio, AR and SW ratio. Utilizing these features, a logistic regression was trained to determine whether two given regions are similar.

$$\sum_i x_p(i) * log(\frac{x_p(i)}{y_p(i)}) \tag{2.6}$$

The above rule-based techniques usually require hand-tuned parameters, while the clustering-based ones are complicated by the incorporation of a post-processing stage, where one has to specify a rather complicated energy model.

In summary, CC-based methods first extract candidate components through a variety of ways including SWT, MSER, color clustering and superpixel segmentation, and then filter out non-text components utilizing human-defined rules or automatically trained classifiers. The remaining CCs are finally grouped into textlines. Table 2.1 presents a selection of recently published methods under this category and summarizes for each work how information is preprocessed, extracted, classified and grouped. This table also gives, for each algorithm, a brief highlight in terms of used dataset and obtained F-score.

Table 2.1: A selection of CC-based methods proposed since 2010.

Method (Year)	Preprocessing	CC extraction	CC filtering	CC grouping	Highlights
Epshtein [EOW10]	Canny edge detector Sobel gradient	SWT	Geometric constraints (SW ratio, CC size, ...)	Heuristic checks	Introducing the SWT; Latin scene text, (F=0.66, ICDAR'05 dataset)
Mosleh [MBH13]	Bandlet-based edge detection	SWT	Hybrid feat.' + K-means	Heuristic checks	Introducing Bandlet-SWT; (F=0.71, ICDAR'05 dataset)
Yao [YBL+12]	Canny edge detector Sobel gradient	SWT	CC-level feat. + RF Chain-level feat. + RF	Greedy agglomerative clustering	Latin/ Chinese multi-oriented scene text, (F=0.6, MSRA-TD500 dataset)
Huang [HLYW13]	Canny edge detector Sobel gradient	SFT	TCD-C + RF TCD-T + RF	Heuristic checks	Introducing SFT and TCD; (F=0.72, ICDAR'05 dataset), (F=0.73, ICDAR'11 dataset)
Chen [CYHL15]	-	Local binarization	CC-level feat. + SVM Chain-level feat. + SVM	Heuristic checks	Latin born-digital text, (F=0.88, ICDAR'13 dataset)
Zhuge [ZL15]	GAM Top/bottom hat	MSER	Corner detection + Heuristics, Multi-frame verification	RLSA algorithm	Chinese artificial text, (F=0.97, private dataset)
Ghanei [GF15]	WMF CPD	SWSR MSER	Projection profiles via radon transform	Meanshift clustering	Arabic and Latin scene text, (F=0.718, ICDAR'11 dataset)
Toro [VTPEK16]	-	MSER	HSW feat. + SVM	Ray-casting	Detect text on traffic panels; (F=0.69, ATPD dataset)
Iwata [SWTF16]	-	Local binarization	Geometric constraints	V. projection profiles	Arabic artificial text, (Private dataset)
Gaddour [GKV16]	-	MSER-like algorithm	Geometric constraints	Heuristic checks	Arabic scene text, (Private dataset)
huang [HQT14]	-	MSER	CNN, NMS	Heuristic checks	Latin scene text, (F=0.78, ICDAR'11 dataset)
Wang [WFCL17]	-	Super-pixel	CNN, Double-threshold	Heuristic checks	Latin scene text, (F=0.82, ICDAR'11 dataset), (F=0.84, ICDAR'13 dataset)
Pei [PYKY16]	Pyramid image	MSER	CNN, AdaBoost, Bayesian classifier	Heuristic checks	Latin/ Chinese multi-oriented scene text, (F=0.72, MSRA-TD500 dataset)

2.2.2 Texture-based methods

In texture-based methods, known also as sliding window-based methods, the input image is scanned using multi-scale sliding windows to extract different texture proprieties and classify image regions as text or non-text based on texture-like features. Some widely utilized features include the Histograms of Oriented Gradients (HOGs) [HP09, ZLMZ11, GWX+13, KSR15], wavelets [SPT09, SPT10, SDTP14, GGJ16], Discrete Cosine Transform (DCT) [QLWS07, AK09, KAJK16], Fourier transform [SPT10, SPT11, RSDE13], Gabor filters [FGG05, YT11, RSDE13, MFSS18] and Local Binary Patterns (LBP) [ZLMZ11, GWX+13, YBG14].
This kind of methods has focused on the binary classification, text *versus* non-text, of a small image window. In other words, it has focused on the following problem:

- Problem (D): Determine whether a given window (block) is a part of a text region.

As shown in Figure 2.12, this methodology and the aforementioned one (CC-based) share some common steps, i.e. the preprocessing, merging and refinement phases, which have been already detailed in the previous section (2.2.1). Thus, in what follows, we mainly focus on the key steps of this category, namely those of feature extraction and classification.

In an earlier work [QLWS07], Qian *et al.* proposed a DCT[2] based method to find candidate text blocks in compressed videos. Firstly, 8 x 8 block-wise DCT was performed on each video

[2] A common use of the DCT is in JPEG and MPEG compression.

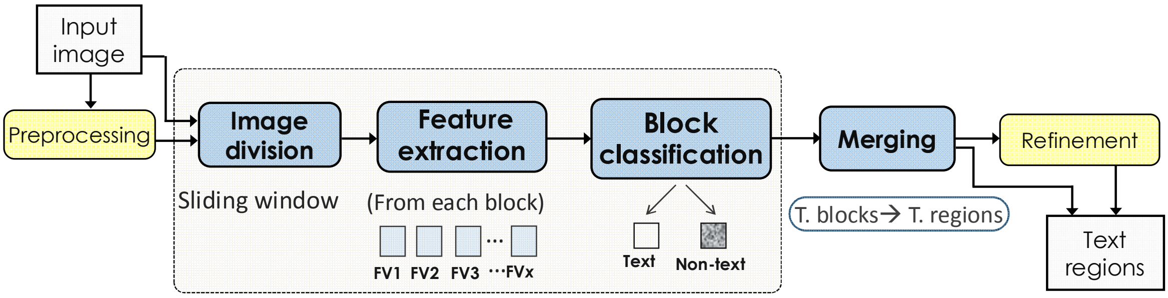


Figure 2.12: Flowchart of a typical sliding window-based text detection method. Yellow rectangles correspond to optional stages.

frame, producing a set of DCT coefficients (AC), given by Equation (2.7).

$$AC_{u,v} = \frac{1}{8} K_u K_v \sum_{x=0}^{7} \sum_{y=0}^{7} b(x, y) \cdot \cos \frac{(2x+1)\,\pi u}{16} \cdot \cos \frac{(2x+1)\,\pi v}{16} \tag{2.7}$$

where $b(x, y)$ represented the image block, u and v respectively denoted the horizontal and vertical frequencies, and K was a coefficient. If u or v was 0, then $K = 1/\sqrt{2}$. Otherwise, $K = 1$. Seven AC coefficients were subsequently selected to capture the horizontal, vertical, and diagonal textures and to represent the texture intensity of each image block. Next, two empirically chosen thresholds were utilized to determine whether or not each block contained text, based on the observation that the texture intensity of text blocks was higher than that of background ones. Morphological operators and projection profiles were finally used to bridge the gaps in the region of characters and to remove background noise. Other methods based on DCT features were proposed by Hsia *et al.* [HHL14] and Kim *et al.* [KAJK16], among others.

In [AK09], Angadi *et al.* firstly divided the input scene image into 8 x 8 blocks and then applied a DCT-based high-pass filter on every block to eliminate constant background. A set of 8 texture features (e.g., homogeneity and contrast) was then computed on every 50 x 50 block of the processed image, and a newly defined discriminant function was employed to classify potential text blocks. Finally, the survived blocks were incrementally merged and then refined by using a set of geometrical constraints.

Shivakumara *et al.* [SPT09] proposed to transform the input grayscale frame into three high-frequency subband images, LH, HL and HH, as shown in Figure 2.13. An 8 x 8 sliding window was next moved across each subband image to calculate a feature vector of 7 elements. The features, which included energy (Equation 2.8), entropy (Equation 2.9), inertia (Equation 2.10), local homogeneity (Equation 2.11), mean (Equation 2.12), second-order and third-order central moments (Equations 2.13 and 2.14) of subband images, were then fed to the K-means clustering for text/background discrimination. Finally, text regions were located by projection profiles.

$$Energy = \sum_{i,j} W^2(i, j) \tag{2.8}$$

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

Figure 2.13: 2D wavelet single level decomposition LH, HL and HH subbands, from [SPT09]. (a) Gray image, (b) Horizontal (LH), (c) Vertical (HL) and (D) Diagonal (HH).

$$Entropy = \sum_{i,j} W(i,j) \cdot \log W(i,j) \tag{2.9}$$

$$Inertia = \sum_{i,j} (i-j)^2 \; W(i,j) \tag{2.10}$$

$$Homogeneity = \sum_{i,j} \frac{1}{1+(i-j)^2} \; W(i,j) \tag{2.11}$$

$$Mean = \frac{1}{N^2} \sum_{i=1}^{N} \sum_{j=1}^{N} W(i,j) \tag{2.12}$$

$$\mu_2 = \frac{1}{N^2} \sum_{i=1}^{N} \sum_{j=1}^{N} (W(i,j) - Mean)^2 \tag{2.13}$$

$$\mu_3 = \frac{1}{N^2} \sum_{i=1}^{N} \sum_{j=1}^{N} (W(i,j) - Mean)^3 \tag{2.14}$$

where $W(i,j)$ is the subband image, at position (i,j) in the window of size N x N.

The same authors [SPT10] introduced a new Fourier Statistical Feature (FSF) in the RGB color space to detect text in video frames. As in [SPT09], a sliding window of size 8 x 8 was employed to extract texture features (Equations 2.8 - 2.14) from each subband frame (R, G, B). The k-means clustering was again utilized. In [SDTP10], the wavelet-median moment features were computed in each window (of size 4 x 4) and subjected to the K-means clustering to classify text pixels from background ones. In [JWS09], the entropy (Equation 2.9), μ_2 (Equation 2.13) and μ_3 (Equation 2.14) were also computed at the block level and fed to an SVM classifier for text/non-text discrimination in video frames. A set of Gray-Level Co-occurrence Matrix (GLCM) features was further utilized in this method. More particularly, the correlation and contrast were employed to represent the image texture.

Ghai *et al.* [GGJ16] proposed an unsupervised clustering technique similar to that described by Shivakumara *et al.* in [SPT09, SPT10] for the classification of multi-channel wavelet features. Firstly, the input image was decomposed into three R, G, B channel images and a 2D wavelet transform was then performed on each image. Next, two statistical features, namely the mean and standard deviation, were calculated from each overlapped sliding window for every high frequency subband (LH, HL and HH). After that, these features were fed to the K-means clustering to classify the image into text, simple background and complex background clusters. Finally, a voting decision process and an area-based filtering were used to locate text regions.

The method in [SPT11] looked for candidate text regions in a video frame using a proposed Fourier-Laplacian filtering followed by a block-wise feature extraction scheme and the K-means clustering. Finally, heuristics concerning straightness and edge density were employed for false positive elimination. The Maximum Gradient Difference (MGD) feature was employed in this method. It was particularly calculated for each pixel, as a difference between the maximal and the minimal values within a local $1 \times N$ window of the gradient image g, as described by Equation (2.15).

$$\begin{aligned} Max(x,y) &= max_{\forall t \in [\frac{-N}{2}, \frac{N}{2}]}\ g(x, y-t) \\ Min(x,y) &= min_{\forall t \in [\frac{-N}{2}, \frac{N}{2}]}\ g(x, y-t) \\ MGD(x,y) &= Max(x,y) - Min(x,y) \end{aligned} \tag{2.15}$$

Typically, the pixels of text regions have larger MGD values than those of background regions. This characteristic was exploited to capture potential text blocks.

In [SDTP14], a combination of wavelet and median moments was proposed to identify text candidates at the block level followed by an angle projection boundary growing method to deal with multi-oriented text in videos.

The Gabor filter has been widely utilized to model texture in text detection [YT11, LLL+11, RSDE13]. Actually, a 2-D Gabor filter is a Gaussian kernel modulated by a sinusoidal carrier wave (as expressed in Equation 2.16), which gives different responses for various positions in a window centered at (x, y).

$$\begin{gathered} g(x,y) = \exp\left(-\frac{x'^2 + y^2 y'^2}{2\sigma^2}\right) \cos\left(2\pi \frac{x'}{\rho} + \varphi\right) \\ x' = x\cos\theta + y\sin\theta, \quad y' = -x\sin\theta + y\cos\theta \end{gathered} \tag{2.16}$$

In [YT11], Yi and Tian applied a set of Gabor filters on character strokes to extract a new text descriptor, namely the Stroke Gabor Words (SGWs). Principal SGWs were then computed for each image window to describe its text strokes. Characteristic distributions generated by SGWs were finally used by the K-means algorithm to classify text and non-text windows.

Lee *et al.* [LLL+11] proposed to extract four different classes of texture features by means of multi-scale sliding windows, and use Modest AdaBoost [3] [VV05] to detect text in natural scenes. Specifically, they exploited the following types of features:

- Local energy of Gabor filters,

[3] Modest Adaboost is a variant of the well-known GentleBoost classifier.

- Mean and standard deviation of X and Y derivatives,
- Six statistical texture measures of image histogram,
- Four coefficients of *Deubechies* wavelets.

Hanif *et al.* [HP09] introduced a boosting framework that integrated feature and weak classifier selection based on computational complexity to generate efficient text detectors. The proposed scheme extracted a set of features from each block, including HOG, Standard Deviation (SD), and Mean Difference Feature (MDF). An MLP-based localizer was then applied as a refinement step.

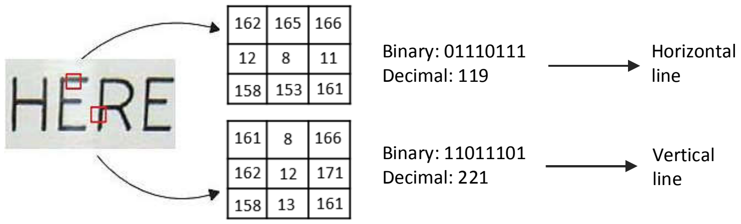


Figure 2.14: Illustration of used LBP in [ZLMZ11].

Zhou *et al.* [ZLMZ11] put forward a multilingual scene text detection method. The input image was firstly divided into 32 x 24 blocks using a sliding window. Three types of texture features including HOG, Mean of Gradients (MG) and LBP (Figure 2.14) were afterwards computed for each block. A cascade AdaBoost classifier was trained based on the extracted features to determine whether the block is part of a text region or not.

Gao *et al.* [GWX+13] suggested to jointly use transfer learning and cascade AdaBoost with weak learners of classification and regression to decide whether a sliding window (of size 32 x 16) contained text or not. The method employed a feature pool that included LBP, HOG, MDF, SD, SWV, SWM and histogram of intensity. A set of heuristic rules was then employed to group and refine the detected text blocks.

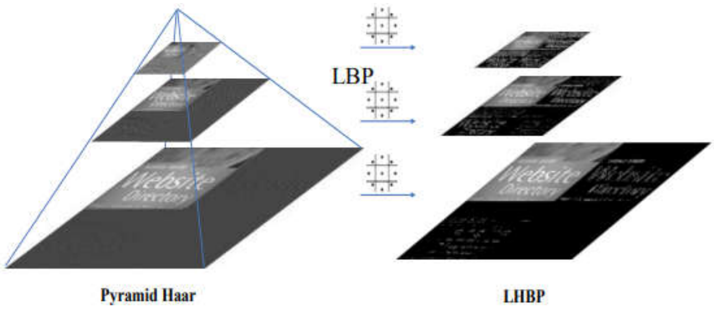
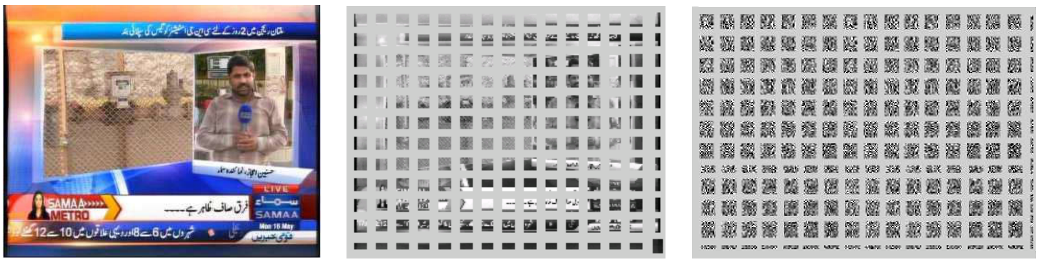


Figure 2.15: LHBP for multi-scale texture feature representation from [JXY+08].

Ji *et al.* [JXY+08] introduced a robust text characterization approach based on Local Haar Binary Pattern (LHBP). More specially, a threshold-restricted LBP was extracted from the high-frequency coefficients of pyramid Haar wavelet, calculated at various resolutions to represent multiscale texture information (Figure 2.15). Assuming that some occurrences between certain directions were notable, a directional correlation analysis (DCA) was subsequently applied to filter out non-directional LHBP regions and locate candidate text regions. Finally, using the LHBP histogram, an SVM classifier was trained to refine the preliminary detection results.

Raza *et al.* [RSDE13] presented a fully-heuristic method for multilingual text detection in video frames. The method relied on a cascade of transforms: Firstly, the Discrete Stationary Wavelet Transform (D-SWT) was exploited to capture the potential text edges in each sliding window (10 x 10). Next, the Gabor filters and the Fast Fourier Transform (FFT) were sequentially applied on the output of the D-SWT to suppress most of the background. A fixed-size sliding window was again used to compute some fractal dimension (FD) features, from the obtained FFT image, and compare their average to a predefined threshold. A final window-based validation step was performed using GLCM features including energy, contrast, correlation and homogeneity. A similar method was proposed in [RAS13]. The authors first divided the input frame into 50 x 50 blocks and then applied the DCT on each block, as shown in Figure 2.16. Two filtering stages of non-text blocks were subsequently carried out. At the first stage, the absolute sum of the DCT coefficients was compared to a predefined threshold. In the second one, the Discrete Fourier Transform (DFT) was applied, and then an empirically chosen threshold was used for the selection / rejection of the DFT coefficients to filter out non-textual information. A 2D Gabor filter was utilized as a final thresholding step followed by the application of morphological operations and projection profiles for refinement.



(a) (b) (c)

Figure 2.16: Conversion of input image into blocks [RAS13]. (a) Input image, (b) Conversion to blocks of 50 x 50, (c) DCT of each block.

Moradi *et al.* [MM13] put forward a method for detecting Farsi/Arabic text in video frames. Firstly, artificial corners were obtained with the help of edge extraction, and font size estimation was performed. Next, a texture intensity picture was created by combining DCT coefficients, and a new LBP picture was introduced to describe the acquired texture pattern. A set of features, including energy, entropy, homogeneity, inertia, and third-order central moment, was then computed on some macro blocks of the processed image and fed into an SVM classifier. Finally, the candidate text blocks underwent empirical rules and projection profile analysis for text refinement.

Yousfi *et al.* [YBG14] put forward two texture-based approaches to detect Arabic text in video frames. The core of the first method is a multiexit asymmetric boosting cascade (GentleBoost) running on multi-block LBP (mb-LBP) features. The second one relies on an AdaBoost classifier using Haar-like features. The mb-LBP can capture large scale structures like corners at different positions, compared to the original LBP (Figure 2.14). Haar-like features on the other hand are based on the difference value between the mean of intensities in contiguous rectangular regions. These features were extracted by sliding a fixed-size window through an input image at multiple scales (to detect text regions of different sizes).

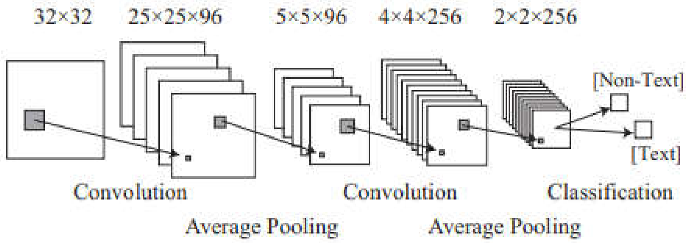


Figure 2.17: CNN for text detection from [WWCN12].

Other than traditional hand-crafted features and statistical models, recent deep learning-based methods have been also used within this sliding window-based methodology. More specifically, Coates *et al.* [CCC+11] proposed the use of an unsupervised feature-learning scheme to generate the features for character *vs* background classification and character recognition. They evaluated a single-layer CNN model on each possible window (32 x 32) of the input image at multiple scales.

Wang *et al.* [WWCN12] proposed to combine a multi-layer CNN with unsupervised feature learning to train character models for both text detection and recognition. They ran CNN for character classification utilizing a sliding window approach (Figure 2.17) and used the responses to localize candidate text regions.

Jaderberg *et al.* [JVZ14] put forward a new CNN architecture, which took a 24 x 24 image block and predicted a text / non-text score, a character class and a bi-gram class. The input image was scanned by the trained network in 16 scales, and a text saliency map was subsequently formed by taking the text / non-text output of the network. Given the saliency maps, word BBs were finally obtained by the RLSA algorithm.

A CNN-based method for Arabic video text detection was suggested in [YBG14]. The employed architecture was composed of six layers. It received training labeled images with a retina of 32 x 64 pixels, as shown in Figure 2.18. The first four layers performed feature extraction and combination and the last two ones represented a simple MLP used for classification.

Gupta *et al.* [GVZ16] proposed generating synthetic images and utilizing them to train a Fully-Convolutional Regression Network (FCRN), which performed text detection and bounding-box regression in multiple scales through all locations of an image. It is worth noting that the input image was first divided into a fixed number of blocks (14 x 14).

Tian *et al.* [THH+16] adapted the Region Proposal Networks (RPN) architecture [RHGS15]

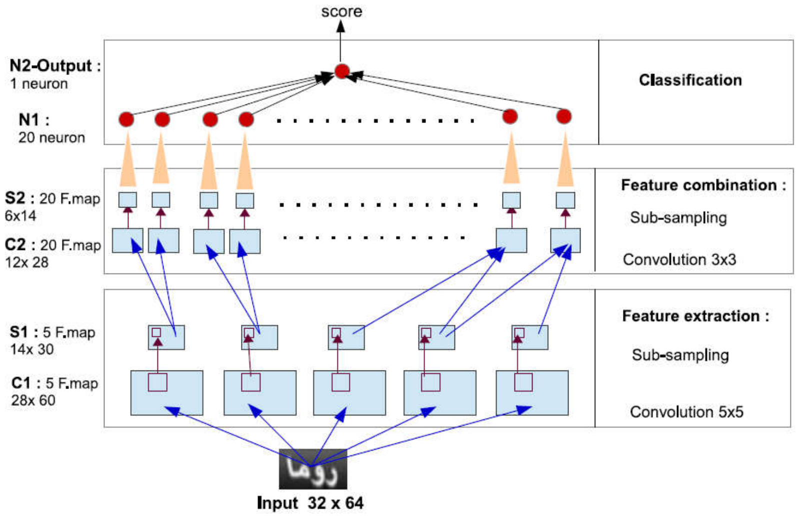


Figure 2.18: CNN-based architecture from [YBG14].

Table 2.2: A selection of texture-based methods proposed since 2008.

Method	Preprocessing / Segmentation	Features	Classification	Grouping/ Postprocessing	Highlights
Shiva [SPT09])	2D wavelet transform / 8x8 sliding window	Energy, entropy, inertia, 2^{sd} and 3^{rd} central moments, homogeneity,..	K-means	Projection profiles	Asian/Latin artificial text, (Private dataset), 15.2 sec/ 256 x 256 image
Ji [JWS09]		Entropy, 2^{sd} and 3^{rd} central moments, GLCM	SVM	Vote mechanism, Morphological filter	Scene/artificial text, (Private dataset)
Shiva [SPT11]	Fourier-Laplacian filtering/ 1xN sliding window	MGD features	K-means	Heuristic checks	Multi-oriented Chinese/ Latin text, (F=0.81, ICDAR'03 dataset), (F=0.77, private dataset)
Lee [LLL+11]	Multi-scale sliding windows	Local energy of Gabor filter, Statistical texture measures of image histogram, Coefficients of Deubechies wavelets,	AdaBoost	Color and gradient-based edge analysis	Latin scene text, (F=0.7, ICDAR'03 dataset), Several min/ image
Gao [GWX+13]	32x16 sliding window	LBP, HOG, MDF, SD, histogram of intensity, number of extended edges in the image	AdaBoost	Heuristic rules	Latin scene text, (F=0.7, ICDAR'11 dataset)
Raza [RAS13]	50x50 block/ cascade of transforms (DCT-based thresholding, DFT-based thresholding)	-	-	2D Gabor filtering, morphological op., projection profiles	Multilingual artificial text, (F=0.84, private dataset), 4.13 sec/ image
Yousfi [YBG14]	Multi-scale sliding windows	mb-LBP features; Haar-like features	GentleBoost; AdaBoost		Artificial Arabic text, (F=0.77, private dataset), 7.25 sec/ 576 x 1024 image
Hanif [HP09]		Mean, HOG, MDF, Standard deviation,	AdaBoost	MLP	Latin scene text, (F=0.49, ICDAR'03 dataset)
Ji [JXY+08]	Multi-scale sliding window (12x10)	LHBP histogram	SVM	DCA filtering	Latin scene text, (F=0.68, ICDAR'03 dataset)
Ghai [GGJ16]	R, G, B channel images + 2D wavelet transform	Mean, Standard deviation	K-means	Voting decision, geometrical filtering	Latin/ Hindi text, (F=0.99, private dataset), 20.1 sec / image
Coates [CCC+11]	Multi-scale sliding windows (32x32)	Unsupervised feature learning	CNN		Latin scene text
Jaderberg [JVZ14]	Multi-scale sliding windows (24x24)	CNN	CNN	RLSA algorithm	Trained on a large number of synthetic images; Latin scene text, (F=0.8, ICDAR'05 dataset), (F=0.56, SVT dataset)
Gupta [GVZ16]	Multi-scale sliding windows (14x14)	CNN	FCRN		Trained on a large number of synthetic images; Latin scene text, (F=0.84, ICDAR'13 dataset)

by sliding a 3-by-3 window across the last convolutional map of the popular VGG-16 architecture [SZ14] and applied a BRNN to jointly predict the text / non-text score, the y-axis coordinate and the anchor-side refinement.

In summary, sliding window-based methods first search for possible text blocks through out the whole image and then identify them using a trained classifier, which usually takes a set of texture features as an input. Table 2.2 presents some sliding window-based methods and summarizes for each method how information is preprocessed (and segmented into windows), classified and grouped. The table also highlights, for each algorithm, the used dataset and the obtained F-measure.

2.2.3 Hybrid methods

This category combines the advantages of both texture-based and CC-based methods for more accurate text detection. For instance, Pan *et al.* [PHL11] utilized an AdaBoost classifier with HOG features to detect text candidates and then extract CCs from multi-scale probability maps. A Conditional Random Field (CRF) model, combining unary component properties and binary contextual relationships, was then employed to discriminate text components from non-text ones.

In [FRSD+16], the image was first segmented using the Toggle Mapping Morphological Segmentation (TMMS) technique [FMC09] to extract potential CCs. A two-stage filtering scheme was then performed to eliminate non-text CCs. The first stage made use of the common geometric constraints (e.g., size and AR of CCs) with fixed thresholds, while the second one was based on the KNN algorithm. The remaining candidate components were then grouped into higher structures. An SVM classifier was finally trained on a set of texture features such as HOG, LBP and GLCM to validate previously grouped CCs. Figure 2.19 presents the pipeline of this method.

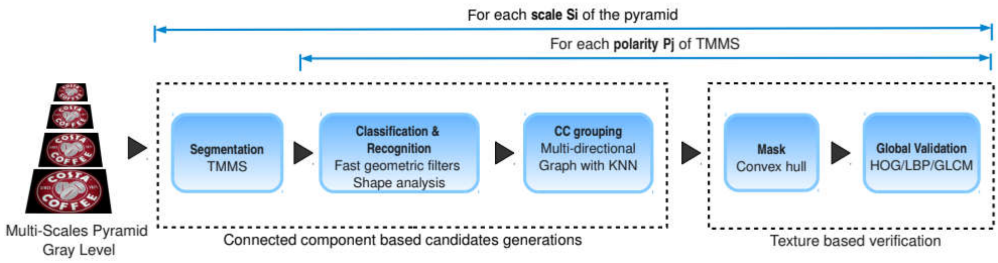


Figure 2.19: Typical example of a hybrid method for scene text detection from [FRSD+16].

Gonzalez *et al.* [GBYB12] proposed a hybrid text location method based on a combination of the complementary proprieties of the MSER technique and the locally adaptive thresholding algorithm for CC generation. Next, CCs were filtered based on geometric criteria such as compactness, AR, OR, solidity and SW ratio. After that, character candidates were grouped into lines, and each line was classified as text / non-text. For this purpose, the authors made use of an SVM classifier and three different types of texture features, namely MDF, standard deviation and HOG.

According to several papers and books [AGP13, LPTL14, YBG14], the hybrid approaches are also referred to as a mixture of heuristic-based and machine learning-based methods. In this way, a lot of presented work in the previous sections can be further classified as hybrid methods, like the suggested work of Chen *et al.* [CYHL15] for born-digital text detection, which consisted of two stages. The first localized text components with an efficient local contrast-based segmentation, while the second verified the previous results based on a linear SVM classifier trained on a set of statistical and structural features. Similarly, Huang *et al.* proposed a two-stage schema for scene text detection [HQT14]. Candidate text components were first determined using the MSER detector. Then a trained CNN model was utilized to give final text lines.

In [AGP10], Anthimopoulos *et al.* suggested combining an edge-based heuristic method with a texture-based machine learning solution for text detection in video frames and images. CCs were firstly detected through an edge map analysis. After that, dilation, opening and projection profiles were introduced to generate initial candidate text areas. In continuation, the results were refined using a sliding window and an SVM classifier trained on edge LBP (eLBP) features that described the local edge distribution .

Such approaches aim to combine the efficiency of heuristic methods with the accuracy and generalization of machine-learning solutions.

2.3 Text recognition in multimedia documents

Video text is usually displayed in different colors and with unknown scales and fonts, which makes it difficult to be recognized by means of a standard OCR engine. According to the literature, there have been essentially two ways to solve this problem, which are: i) Recognizing characters by separating text pixels from the background beforehand, and then applying an available OCR software [ZLL10, ZW13, HWL15, RSR$^+$15]. ii) Recognizing characters by using features and classifiers specially designed for video or/and natural scene text [EGMS12, SL14, JSVZ14, ZCLX16]. Figure 2.20 depicts the different categories of video and scene text recognition and the relation between their underlying modules.

The first category requires an appropriate preprocessing stage to obtain characters with well-defined shapes and a plain background. Therefore, several studies have made use of text extraction, binarization and enhancement techniques for this aim. The terms "extraction" and "binarization" are often used synonymously and operate to extract character pixels from the localized text regions prior to recognition.

2.3.1 Robust binarization for better recognition

Document image binarization represents an active area of research. It has achieved very good performances on scanned documents and could deal with certain degraded document images. However, this task is particularly challenging for scene and video text due to presence of various artifacts and complex backgrounds. To overcome these problems a variety of binarization methods have been proposed, which can be classified into two broad categories: classical thresholding-based and machine learning-based methods.

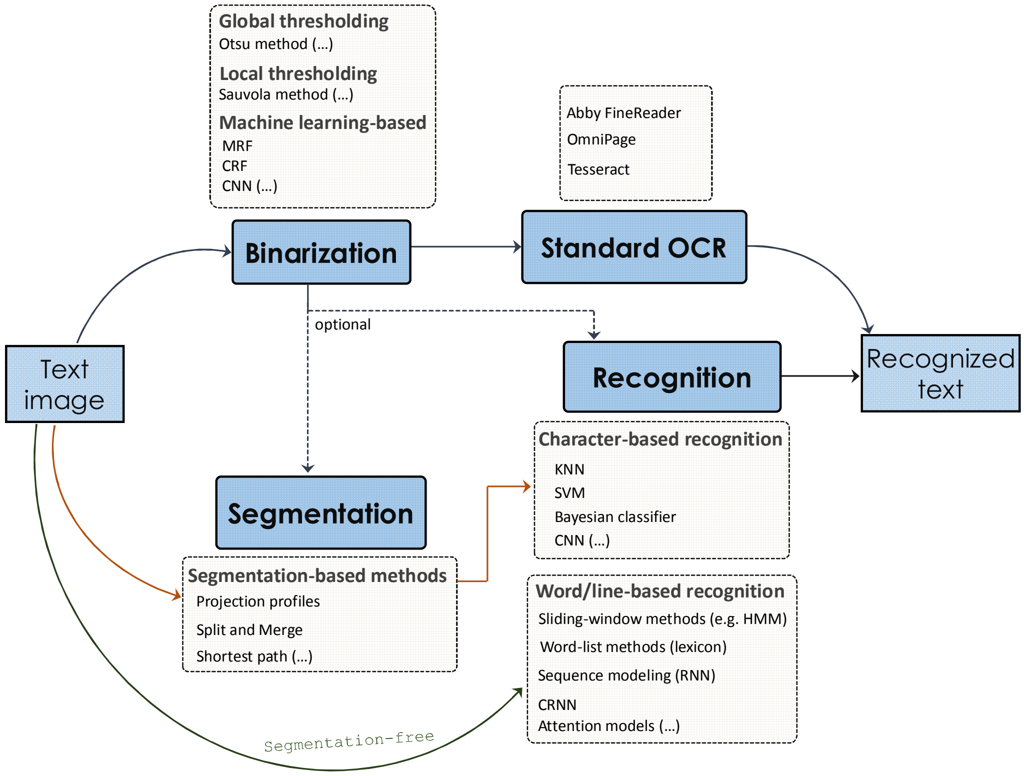


Figure 2.20: Text recognition methods in videos and images

Classical thresholding methods

This kind of binarization can be further categorized into global and local adaptive approaches. In the global approaches there is only one threshold for the whole text image. Otsu's method [Ots79] represents a typical example of this category. It assumes that an image has two classes of pixels, namely text and background, and employs an algorithm that looks for a global threshold maximizing the separability between the two classes. This works well on images with high contrast and clean background, which is not the case of scene and video text.

On the other hand, adaptive thresholding methods compute a local threshold T for each pixel p on the basis of the information contained within a neighborhood of p. Niblack [Nib85] and Sauvola [SP00] methods are perhaps the best known techniques in this class. The following formula $T = m + s.k$ was suggested by Niblack to compute the local threshold based on the mean m and standard deviation s of the gray values in a fixed-size window centered at p. Wolf *et al.* [WJ04] improved the Niblack algorithm and proposed formulating the decision of binarization by means of a local contrast instead of gray values. A new formula was consequently defined, by Equation (2.17), to compute T in multimedia documents.

$$T = (1 - \alpha)m + \alpha.G + \alpha\frac{s}{R}(m - G) \tag{2.17}$$

where α is a parameter controlling the incertitude related to m, and G is the minimal gray value of the entire image. Later, Zhou *et al.* [ZLL10] proposed a thresholding-based binarization for video text images. After performing Canny edge detection, the inner side of boundaries were identified by using a local threshold method. The contours were then filled up utilizing a modified flood-fill algorithm to form text regions. In [NGP11], the authors exploited lower and upper baselines of the text, the convex hull analysis and the stroke width information together with the adaptive thresholding for artificial text binarization in video frames. Although most of these methods have performed satisfactorily for many cases, they suffer from problems like the high sensitivity to the choice of parameters and the failure in handling images with noisy background and similar foreground-background texture or colors.

Machine learning methods

Other video/image text recognition methods perform binarization based on clustering or, in general, on machine learning techniques to solve the aforementioned problems. Saidane and Garcia [SG07] put forward an automatic binarization method for text areas in video frames based on a CNN model, which took color text patches as an input and learned to output the corresponding binary image. The CNN architecture was composed of four layers, namely convolutional, sub-sampling, up-sampling and inverse convolutional.
Cho *et al.* [CSLK11] made use of CRFs for text extraction by a superpixel representation of the input image. Character features concerning color, edge strength, stroke width and contextual feature were employed. Another method [ZLY+11] exploited also the CRFs for scene text extraction. The method relied specifically on the use of a two-step iterative CRF scheme along with an OCR module as a region filtering stage.
Zhang and Wang suggested a method [ZW13] for binarizing artificial text in video using the K-means algorithm in the RGB space to segment the input text image into k clusters, the Markov Random Field (MRF) model to get the binarization result, and the Log-Gabor filters as a refinement step. Other studies [SXWZ12, SLT12] proposed to divide the input image into three classes: foreground, background, and uncertain. Next, they classified those uncertain pixels by applying an MRF model.
Hu *et al.* [HWL15] put forward a binarization method for overlaid and scene text utilizing two confidence maps and the K-means clustering algorithm.
Milyaev *et al.* [MBN+13] proposed a scene text binarization technique, where they first obtained an initial estimate of binarization with the Niblack algorithm [Nib85]. After that, they performed the Laplacian operator on the image intensity to compute the unary terms of the energy function, followed by a global optimization using a graph cut algorithm.
Roy *et al.* [RSR+15] introduced a new method to binarize video text by means of a Bayesian classifier for text/non-text pixels discrimination and a connected component analysis for text information restoring.

Recently, Mishra *et al.* [MAJ17] suggested a method (Figure 2.21) that modeled the color and SW distributions of text and background using GMMs and computed unary and pairwise costs for every pixel. The problem was then solved by minimizing two energy functions E_c and E_{sw} to find the optimal binarization using an iterative graph cut algorithm.
A benchmark of several binarization methods on ICDAR 2003 (scene images) [LPS+05], IC-

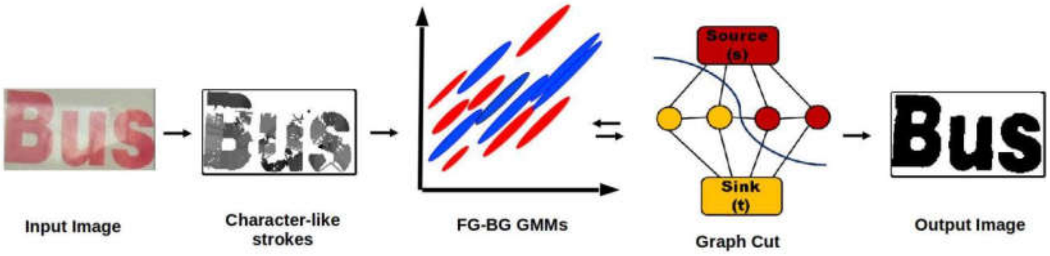


Figure 2.21: Overview of the binarization method by Mishra et al. [MAJ17]

DAR 2011 (born-digital images) [KMM+11] and SVT (street view images) [WB10] datasets was presented in [MAJ17]. Text recognition accuracy, pixel-level and atom-level [4] measures were used for the evaluation. As shown in Figure 2.22, the state-of-the-art results were obtained with energy minimization-based methods [MAJ17].

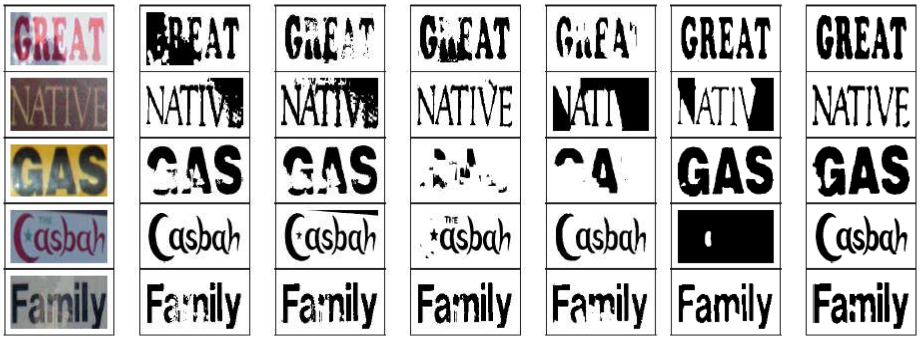


Figure 2.22: Comparison of different binarization methods. From left to right: input image, Otsu [Ots79], Wolf and Doerman [WJ04], Kasar et al. [KKR07], Milyaev et al. [MBN+13], Howe [How11] and Mishra et al. [MAJ17]. This figure was adapted from [MAJ17].

After obtaining the binarized text image, most of the previous methods made use of an available OCR engine like Google Tesseract [5] OmniPage, or ABBYY Finereader [6] for recognition, as depicted in Table 2.3.

2.3.2 Specific methods for text recognition in images and videos

In contrast to the previous methods, this category specifically uses classifiers directly on text regions mixed with background objects. Like in document-based OCR, the recognition module here can also be divided into segmentation-based and segmentation-free methods. The former, known as analytical approach, segments the lines, words or sub-words into smaller units (characters or graphemes) for recognition. The latter, also called global approach, takes the whole line or word image for recognition.

[4] An atom as introduced by Clavelli *et al.* [CKL10] is a CC, which typically corresponds to a single character, but can also correspond to a part of a character or to multiple characters, when characters are joint together.

[5] available at https://github.com/tesseract-ocr/

[6] available at: http://www.abbyy.com

Table 2.3: A selection of binarization-based text recognition methods.
WRR and I-11 respectively denote the Word Recognition Rate metric and the ICDAR'11 dataset.

Method (Ref)	Methodology	Preprocessing	Classification	Postprocessing	Highlights
Zhou [ZLL10]	Thresholding-based binarization	Canny edge	-	Local thresholding Flood-fill algorithm	Artificial Latin text, OmniPage, (Private dataset)
Ntirogiannis [NGP11]	Thresholding-based binarization	Baselines extraction, SW detection, ALLT binarization	-	Convex Hull analysis	Latin artificial text, ABBYY, (WRR= 88, private dataset)
Saidane [SG07]	Machine learn-based binarization	-	CNN	-	Latin artificial text, ABBYY, Tesseract, (Private dataset)
Cho [CSLK11]	Machine learn-based binarization	Superpixel representation	Feat.: color, edge, SW Classifier: CRF	-	Latin scene text
Zhang [ZW13]	Machine learn-based binarization	K-means	MRF	Log-Gabor filter	Chinese artificial text, Tesseract
Roy [RSR+15]	Machine learn-based binarization	Integration of color, wavelet and gradient sub-bands	Bayesian classifier	CC analysis	Multi-oriented text, Tesseract
Milyaev [MBN+13]	Machine learn-based binarization	Niblack algorithm, Laplacian operator	Graph cut algorithm	-	Latin scene text, (WRR=22.07, I-11, Tesseract), (WRR=22.57, I-11, ABBYY)
Mishra [MAJ17]	Machine learn-based binarization	Stroke map, GMM	Graph cut algorithm	-	Latin scene text, (WRR=62.57, I-11, Tesseract), (WRR=58.1, I-11, ABBYY)

Segmentation-based recognition

Text images, under this category, are segmented into individual characters before the recognition step; i.e., no recognition will be done till the text region is fully segmented. The projection-based method is among the simplest ways for dealing with the text segmentation [HAV+12]. It basically calculates the average gray value for each pixel column an then splits every blank region in the middle, making it vulnerable to disconnected structure (e.g., PAWs) and touching letters (e.g., cursive scripts like Arabic). Ben Halima *et al.* [HAV+12] adopted such a methodology to recognize Arabic text in news video frames. Textlines were first binarized and then segmented (Figure 2.23) using projection profiles. Characters were finally classified utilizing the fuzzy KNN algorithm applied on a set of hand-crafted features, and the best results were obtained for k=10. The used features include occlusions, projections, black-white transitions, number of components in the character and location of dots.

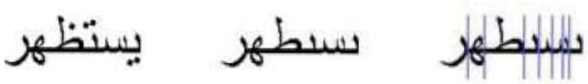


Figure 2.23: Segmentation step proposed in [HAV+12]

Other researches have exploited heuristic rules to further split and merge the obtained segments based on some assumptions about the width and height of characters [HMZ09]. However, these methods can lead to numerous segmentation errors particularly for video and scene images with complex backgrounds. Indeed, performing a sophisticated segmentation requires finding the optimal threshold related to the employed projection, which is very critical for such content. To address this problem, several methods have proposed to model the segmentation as a minimal cost path finding task. Yet, these methods have been only dedicated to few languages like Latin and Chinese. For instance, Phan *et al.* [PSST11] suggested a method based on the Gradient Vector Flow (GVF) for video character segmentation. The idea consisted in finding nonlinear splitting paths (rather than vertical splitting lines) that corresponded to candidate cut pixels. More specifically, a two-pass path finding process was applied where potential cuts were located in the forward pass. The backward pass served as a verification step to reject false cuts, as shown in Figure 2.24.

Shivakumara *et al.* [SPLT11] suggested a segmentation approach, which found least-cost cuts

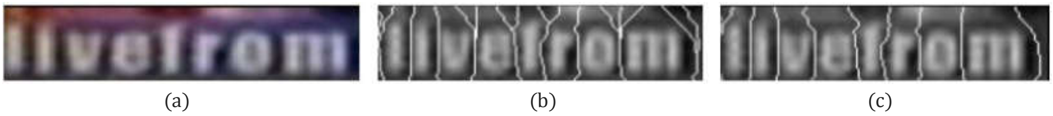


Figure 2.24: Character segmentation with GVF method proposed in [PSST11]. Forward pass (a) and backward pass (b) results of proposed path finding procedure.

by dynamic programming. Structural features were utilized for classification and recognition. Finally, a voting criterion was adopted to classify 62 character classes into various smaller ones.

Different from the previous methods, Saidane and Garcia [SG08] proposed to use a CNN model for the segmentation of video text lines. The input was three images, each of which corresponded to one RGB color channel of the text image. The output was a vector that classified the input columns into a border zone or a character one. As a side note, the network was trained in a supervised fashion using many synthetic text images where the exact positions of the characters were known. Although this method achieved high segmentation accuracies, it allowed only vertical cuts. Thus, handling overlapping cases (e.g., ligatures in Arabic script) was not guaranteed. As a continuation of this work, the authors suggested a segmentation-based method [SGD09] that made use of a CNN-based character recognizer. The network was trained to classify character images and then was applied on each segment. The classification was finally performed using the best path search algorithm.

In [EGMS14], the video text images were first processed utilizing a combination of intensity analysis and multi-frame integration to separate the text pixels from the background. Then, a shortest path algorithm [LLP96] was employed to perform segmentation. The recognition module was similar to the one presented in [SGD09].

Alsharif [AP13] proposed a lexicon-free segmentation-based approach for recognizing words in scene images. The authors opted for a hybrid model composed of a four-state HMM and a four-layer convolutional Maxout network to segment the words into characters. Afterwards, they trained a network, namely *segmentation correction Maxout* to reduce the amount of under and over-segmentation errors. Finally, a variant of the Viterbi algorithm was used for recognition.

Iwata *et al.* [IOWK16] put forward a segmentation-based method to recognize Arabic text in video frames. Text lines were first segmented into words utilizing a space detection algorithm. The character candidates were then over-segmented into primitive segments (Figure 2.25). The recognition was performed by the modified quadratic function (MQDF) classifier at the character level and by the dynamic programming at the word level.

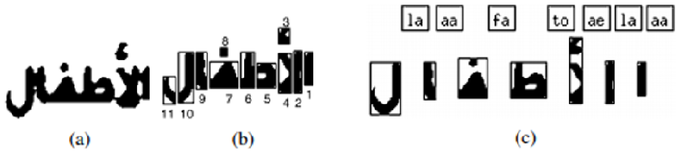


Figure 2.25: Result of word segmentation from [SWTF16]: (a) Input word image, (b) primitive segments and (c) character segmentation

As it can be inferred from the reviewed methods, most of them made use of projection profiles, heuristics or more sophisticated techniques, such as CNNs and path-finding algorithms, so as to explicitly segment the text image into words and then characters. The recognition phase consists in a character-based classification of each segment. The classification results are then concatenated in different manners to form the final transcription. However, with such approaches, the segmentation errors can propagate further and impact the recognition performance. For example, they may fail when the spaces between characters, PAWs or words in a textline are not uniform. Moreover, the segmentation annotations require additional resource consumption.

Segmentation-free recognition

This category regroups alternative methods that completely avoid the segmentation stage. This is generally achieved by means of a sliding-window procedure and an HMM classifier [RRS+13, BKR+17], or more recently by using deep networks such as RNNs [AJQ14], CNNs [JSVZ14, JSVZ16] or their combination [GCWL17, HHQ+16].
Roy *et al.* [RRS+13] put forward a sliding window-based method for multi-oriented text recognition in scene images. First, the input text image was binarized and the window trajectory was estimated by a polynomial function. Next, the Marti-Bunke [MB01] and local gradient histogram features were extracted from each sliding window, followed by character modeling via HMMs. Finally, the Viterbi algorithm was employed to find the best path through the model, which was the recognition result.

Bhunia *et al.* [BKR+17] suggested to perform a color channel selection procedure to avoid complex binarization for word recognition in scene/video images. The method consists in applying a multi-label SVM classifier trained on wavelet-based features to select the proper color channel that would provide a higher recognition performance. A set of PHOG features was then extracted on each sliding window (after converting it to the selected color channel) and fed as an input to HMMs for recognition.

An HMM-based scheme was suggested by Som *et al.* [SCS09] for the recognition of scrolling text in Turkish broadcast news. The authors employed Gaussian mixtures to represent the output distributions of HMM states. A set of synthetic character glyphs was used to construct the training data for different glyph models.

The great success of deep learning methods, specifically CNNs, in various computer vision tasks has enlightened researchers in the field of scene and video text recognition. A peculiar characteristic of text objects is that their length is not constant; e.g., an Arabic word may consist of 2 characters such as (من) or 10 characters such as (المترادفات). Consequently, most popular CNN models cannot be applied directly to text sequence prediction, since they operate on fixed-dimension inputs. Some studies have been made to address this problem for scene text recognition.

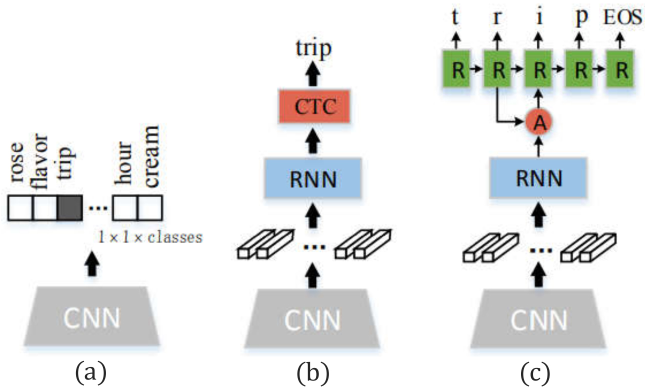


Figure 2.26: Three typical CRNN-based architectures for scene text recognition. (a) CNN + softmax. (b) CRNN + CTC and (c) CRNN + Attention. This figure was adapted from [GCWL17].

For example, Jaderberg *et al.* [JSVZ14, JSVZ16] proposed a CNN-based classifier to holistically recognize English words in natural scene images. More specifically, the recognition was formulated as an image classification problem where a class label was assigned to each word (see Figure 2.26(a)) in a predefined large lexicon $\tilde{9}0$k words, leading to a large trained model with a huge number of classes. This is a somewhat impractical solution for other texts such as Arabic, since their basic combinations can be greater than one million. Other methods (such as [WWCN12]) proposed detecting individual characters and then recognizing each of them with CNN models trained using labeled character images. Such methods can be affected by a large amount of inter/intra-character confusions. Therefore, they often rely on an accurate text detector.

RNNs represent another important branch of deep neural networks. They have the ability to model contextual information using their recurrent connections. Thus, RNNs are good at recognizing patterns occurring in time series like speech and text.

In this context, Zhai *et al.* [ZCLX16] adopted a bidirectional RNN (BRNN) architecture and a connectionist temporal classification (CTC) layer for Chinese news text recognition. The authors collected two million news titles from 13 TV channels to train the suggested model. Su and Lu [SL14] extracted sequences of HOG features as a sequential image representation and generated the corresponding character sequence with RNNs.
Naz *et al.* [NUA+17] suggested an RNN-based system for Urdu Nasta'liq [7] text recognition.

[7] Urdu is a derivative of the Arabic alphabet.

The input textline images were first normalized to a fixed height, and then transformed into a sequence of manually-designed features including projection, pixel distribution and GLCM features. An RNN was trained on these features in a frame-wise fashion, followed by a CTC decoding layer that transcribed the input data and finally provided the recognized sequence.

Recently, some published studies [EGMS12, YBG15b, SBY17, WGLZ17] have jointly used CNN and RNN models for recognizing text in natural scene images or/and videos. As shown in Figure 2.26(b), these methods are generally composed of two modules, one deep CNN for feature extraction and one BRNN for sequence modeling. In [EGMS12], video texts were first represented as sequences of learned features with a multi-scale scanning scheme. The sequence of features was then fed into a connectionist recurrent model, which would recognize text words without prior binarization or explicit character segmentation.
Shi *et al.* [SBY17] treated word recognition as a sequence labeling problem. CNNs and BRNNs were employed to respectively extract feature sequences from input images and generate sequential labels of arbitrary length. CTC was adopted to decode the sequence.
Wang *et al.* [WGLZ17] explored a GMM-HMM bootstrap model to align the frame sequence with the transcription. Next, the alignments were utilized as supervision for CNN training. Finally, BRNNs were used to model the text sequences.

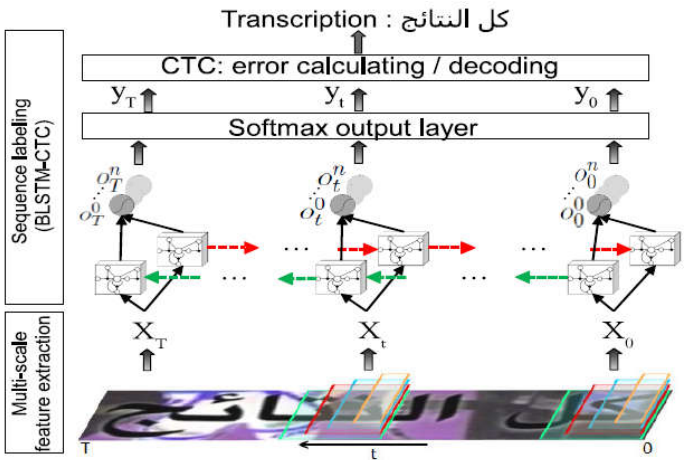


Figure 2.27: BLSTM-based video text recognition (from [YBG15b])

In [YBG15b], three RNN-based systems were proposed for embedded Arabic text recognition in video frames. These systems differed in their feature extraction scheme and had a common classifier. Firstly, a multi-scale sliding window was employed to extract features based on three different learning-feature models, where two of them were based on deep auto-encoders (AE) and the other one on CNN. A Bidirectional Long-Short Term Memory (BLSTM) network coupled with a CTC layer (Figure 2.27) was then utilized to predict correct characters from the associated sequence of features.

The effectiveness of the CNN-RNN paradigm was also exploited by He *et al.* [HHQ+16] and Qiang *et al.* [QDGJ16], among others, while Lee *et al.* [LO16] and Gao *et al.* [GCWL17] suggested incorporating an attention-based mechanism to weight the feature sequence and perform soft-feature selection, as shown in Figure 2.26(c). The attention mechanism plays a key role in the feature learning process. It allows the model to selectively focus on the most relevant parts of incoming features. More particularly, a residual attention module proposed by [WJQ+17] was employed in [LO16] to effectively suppress the response of background noise while enhancing the representation of foreground text.

Other text recognition methods not relying on neural networks have also brought novel representations and insightful ideas into this area of research. Rodriguez-Serrano *et al.* [RSPM13] and Almazan *et al.* [AGFV14] suggested embedding text strings and word images in a common Euclidean space. Word recognition was consequently converted into a retrieval problem. Yao *et al.* [YBSL14] put forward an alternative way for scene character representation, denoted as Strokelets, which was a combination of multi-scale mid-level features. While achieving promising performances on some standard datasets, these methods have been generally outperformed by the previously discussed methods based on deep networks.

Table 2.4: Text recognition methods using and avoiding character segmentation. WRR, I-03 and I-15 denote the Word Recognition Rate metric, ICDAR'03 dataset and ICDAR'15 dataset, respectively.

Method (Ref)	Methodology	Preprocessing/ Segmentation	Feature extraction	Recognition	Highlights
Shivakumara [SPLT11]	Segmentation-based recognition	Seg: Least-cost cuts by DP	Strucutral feat.	Voting-based char. classification	Latin artificial character
Ben Halima [HAV+12]	Segmentation-based recognition	Prep: Binarization/ Seg: Projection	Hand-crafted feat.	Fuzzy KNN-based char. classification	Arabic artificial text, (Private dataset)
Alsharif [AP13]	Segmentation-based recognition	Seg: HMM/Maxout model	-	Viterbi algorithm	Latin scene text, (WRR=85.1, I-03), (WRR=74.3, SVT)
Elagouni [EGMS14]	Segmentation-based recognition	Prep: Intensity analysis, MFI/ Seg: Shortest path algorithm	Multi-scale scanning	CNN-based char. recognizer Graph model	Latin artificial text, (WRR=85.8, private dataset1), (WRR=41.19, private dataset2)
Iwata [SWTF16]	Segmentation-based recognition	Prep: Binarization/ Seg: Over-segmentation	Chain code histogram	MQDF for char. classification + DP for word rec.	Arabic artificial text, (WRR=94, private dataset)
Roy [RRS+13]	Segmentation-free	Prep: Wavelet-Gradient based binarization	MB feat. + Local gradient histogram	HMM classifier	Multi-oriented scene text, (WRR=63.28, I-03), (WRR=58.41, MSRA-TD500)
Bhunia [BKR+17]	Segmentation-free	Prep: Color channel selection	PHOG	HMM classifier	Scene/artificial text, (WRR=78.44, I-03), (WRR=75.41, I-15 video)
Jaderberg [JSVZ14]	Segmentation-free	-	-	CNN	Lexicon-based (90k word); Latin scene text, (WRR=89.5, I-03), (WRR=68.0, SVT)
Su and Lu [SL14]	Segmentation-free	-	HOG	BRNN + CTC	Latin scene text, (WRR=82.0, I-03), (WRR=83.0, SVT)
Naz [NUA+17]	Segmentation-free	-	Hand-crafted feat.	MDRNN + CTC	Urdu text, (Private dataset)
Yousfi [YBG15b]	Segmentation-free	-	CNN	BRNN + CTC	Arabic artificial text , (WRR=71.26, ALIF dataset)
Qiang [QDGJ16]	Segmentation-free	-	CNN	BRNN + CTC	Street View house number, (WRR=91.0,SVHN dataset)
Lee [LO16]	Segmentation-free	-	CNN	Attention-based RNN	Latin scene text, (WRR=88.7, I-03), (WRR=80.7, SVT)
Gao [GCWL17]	Segmentation-free	-	CNN+Attention	CNN + CTC	Latin scene text, (WRR=83.0, SVT)

To sum up, this kind of methodology permits avoiding the problems of segmentation.

Nevertheless, it usually requires a large number of samples covering various text fonts and background to train the classifier. Table 2.4 presents a selection of methods for artificial/ scene text recognition using and avoiding character segmentation. The table highlights, for each each method, the employed techniques, features and classifiers, the used dataset and the obtained recognition rate.

End-to-end text recognition

Recent advances in computer technology as well as the progress made in implementing practical computation and memory capabilities have led to other text recognition architectures, namely end-to-end recognition. The latter can be seen as a unified framework for both text detection and recognition, which converts text regions in the entire image/frame into text strings. Considering a small lexicon, word spotting provides a straightforward way for designing end-to-end recognition where specific words are directly matched with image patches using character and word models [WB10]. Goel *et al.* [GMAJ13] put forward a word spotting approach, which looked for converting the lexicon into a collection of synthetic word images. The recognition task was then treated as a problem of retrieving the best match from the lexicon image set by means of a weighted DTW technique.
For an open lexicon, however, word spotting methodologies are relatively impracticable due to the large search space. In this case, a strong character representation and sophisticated optimization strategies are required to face both image and lexicon complexities. Neumann and Matas [NM10, NM13] were among the first researchers to suggest an unconstrained (do not require a word list) end-to-end text recognition approach. In [NM13], they integrated character detection and recognition based on oriented stroke features. The gradient image was first convolved using a set of oriented bar filters. Strokes with specific orientations in a relative position were then employed to detect and recognize character candidates. Dynamic programming was finally adopted to optimize the recognition results.

As it can be observed in Table 2.4, most of the methods rely on a feature extraction process. Yet, feature design is a challenging and time-consuming task due to its dependence on the domain knowledge and past experience of human experts [SZK$^+$12, CRC16]. On the other hand, there has been recent work proposing recognition systems that would perform automatic feature extraction inside a learning scheme operating directly on raw pixel data. Such systems have shown high performances on different OCR tasks [Gra12, PBKL14, ML15] and received considerable attention, especially with the resurgence of LSTM-RNN models. A comparison between the results of the work by [YSBS15, CRC16] on Arabic handwriting recognition demonstrates that a 1D LSTM network that operates on raw image pixels [YSBS15] has outperformed the same network trained using either handcrafted or learned features [CRC16]. Motivated by this observation, a novel method for Arabic video text recognition is proposed in Chapter 5, based on a simple and effective preprocessing step and a multidimensional LSTM network operating directly on raw pixel values.

2.4 Summary

This section summarizes the previously reviewed approaches and presents their strengths and limitations. For text detection, we have surveyed three categories of methods, namely CC-based, sliding window-based and hybrid methods.

CC-based methods first extract candidate components through a variety of ways including SWT, MSER and color clustering. Non-text CCs are then filtered by heuristic rules or classifiers. Finally, the remaining CCs are merged into a higher structure to form the final detection results. These methods are generally faster and more efficient compared to the sliding window-based ones, because the number of components to be processed is relatively small. Moreover, they are invariant to scale, rotation and font variation. Therefore, they have become the mainstream methodology in text detection. However, such methods are sensitive to image quality and cluttered backgrounds.

As opposed to CC-based methods, sliding window-based ones firstly search for possible text blocks over windows at multiple scales in an image. Next, text and non-text regions are distinguished by a trained classifier, which often uses traditional texture features such as HOG, LBP and FFT, or automatically learned features. A major drawback of such a category is its high computational cost as all locations and scales are exhaustively scanned.

The third category, hybrid methods, is a mixture of CC-based and texture-based methods. It is also referred to in the literature as a combination of heuristic-based and machine learning-based methods. The aim here is to benefit from the efficiency of human-defined heuristics and the accuracy of machine-learning algorithms to improve the detection performance.
Our work on text detection in Chapter 4 falls into this category as it combines two stages: a fully heuristic detection phase and a machine learning-based classification scheme.

In section 2.3, we have discussed two different methodologies for video and scene text recognition. The first one makes use of a binarization step, to extract text pixels and remove background ones, prior to recognition. A variety of video text binarization methods have been proposed, which can be classified into thresholding-based techniques and clustering-based ones. After obtaining the binarized text image, available OCR engines like Tesseract or OmniPage are generally employed for recognition. Yet, in this kind of methodology the recognition performance usually relies on the efficiency of binarization and may suffer from noise and distortion in complex backgrounds.

The second methodology recognizes characters by using features and classifiers specially designed for video or/and scene text. It can be further categorized into segmentation-based and segmentation-free methods. The former segments the line or word images into smaller units (sub-words, characters or graphemes) for recognition. With such methods, the segmentation errors may propagate further and impact the recognition performance. Whereas, the latter takes as an input the whole line or word for recognition. Such methods are based on classifiers like HMMs, RNNs, their combination or other sequence learning models.
Our work on text recognition in Chapter 5 falls into this category as it is based on a segmentation-free method, which relies on the use of RNN-LSTM networks.

2.5 Conclusion

Text detection and recognition in videos and images, as a major research branch of content-based information retrieval and indexing, continues to be a topic of interest for many researchers. In this chapter, we have reviewed a large panel of methods and techniques that have contributed to an impressive progress in such a field of research.
Several observations can be made based on this survey:

- A first remarkable point is that most of the existing methods that have tackled Video OCR problems focus on Latin and Chinese texts. Hence, they can not be directly replicated for Arabic text, which is the focus of our work. For instance, several merging procedures, in the detection task, assume that a text character and its siblings have similar sizes and proper distances. This is not the case for cursive scripts like Arabic, which usually has a non-uniform inter/intra-word distance and a variable size of character bounding boxes.

- A second important point that is related to the wide use of preprocessing and hand-crafted features in most existing methods for both detection and recognition tasks. This can limit the generalization ability of the proposed method to a certain type of data and challenges.

- A third point we have highlighted is the absence of standard and publicly available datasets to build and evaluate Arabic video text detection and recognition systems. This could explain the lower performances achieved for such a language compared to Latin.

These different points will be intensively investigated in this thesis. In the next chapters, we describe our detection and recognition methods. But first of all, we present in the following the proposed dataset, its annotation framework and the different used evaluation protocols and metrics.

Chapter 3

Proposed Dataset and Experimental Settings

3.1 Introduction

The number of standard datasets has been growing significantly for all scientific research fields over the last decade. This can be explained by a variety of fundamental requirements, ranging from the development and test of specific methods to the need of a systematic benchmark. Up to our knowledge, no attempt has been previously made on the development of standard datasets for embedded Arabic text in news videos, despite the important number of Arabic news TV channels. This was a major difficult at the beginning of our thesis. For this reason, we have developed a standard annotated dataset of video clips containing Arabic text. In this chapter, we first present a short survey of currently used datasets in text detection/recognition problems. Then we introduce the acquisition procedure of our collected data. Next, we describe the content of the suggested dataset in terms of characteristics, statistics, data organization and ground-truth annotation. Finally, we present the proposed evaluation protocols and metrics. It is worth note that the chapter also includes details about the proposed ground-truthing and evaluation tools, *AcTiV-GT* and *AcTiV-Eval*.

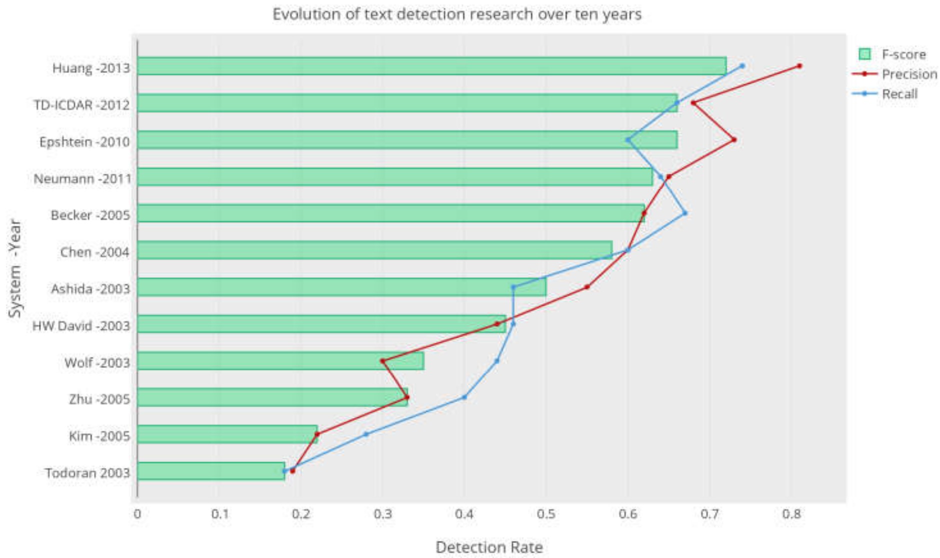


Figure 3.1: A selection of some text detection methods [Luc05, HLYW13, ZYB16] showing the evolution of this area of research over ten years

3.2 Related Work

As mentioned in the previous chapter, the detection and recognition of text in images and videos represent a very active domain nowadays [SSD11, KSU+13, LPTL14, YD15]. Much of the progress that has been made in this field is attributed to the availability of standard datasets. The most popular of these is the dataset of ICDAR 2003 "Robust Reading Competitions" (RRC) [LPS+05], prepared for scene text localization, character segmentation (removing background pixels) and word recognition. This dataset includes 509 text images in real environments captured with hand-held devices. 258 images from the database are used for training and the remaining 251 images constitute the test set. This dataset was also used in the ICDAR 2005 "Text Locating Competition" [Luc05]. Figure 3.1 shows the evolution of the Latin text detection research between 2003 and 2013 [Luc05, HLYW13, ZYB16] taking as a benchmark the ICDAR 2003 dataset. As it can be observed, the method of Huang *et al.* [HLYW13] outperformed other approaches by a large margin. This method enhanced the SWT algorithm by color information and introduced the TCD descriptors for text/non-text discrimination. In the word recognition task, the best accuracy of 93.1% was achieved by Jaderberg *et al.* [JSVZ16] using their proposed CNN model. Some examples of this dataset are depicted in Figure 3.2(a).
The dataset of ICDAR 2011 [SSD11] was inherited from the benchmark used in the previ-

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

Figure 3.2: Typical samples from ICDAR2003 (a), MSRA-TD500 (b), NEOCR (c) and KAIST (d) datasets.

ous ICDAR RRC (i.e., 2003 and 2005), but it underwent some extensions and modifications since there were some missing ground-truth information and several imprecise word bounding boxes. The final dataset consists of 485 full images for the localization task and 1,564 cropped word images for the recognition task. On this dataset, the detection method of Liao *et al.* [LSB+17] attained a state-of-the-art performance with an F-score of 82%. This method made use of the FCN networks followed by a standard non-maximum suppression process.

In ICDAR 2013 RRC [KSU+13], a new dataset for video text detection, tracking and recognition was proposed. It contains 28 short videos (10 seconds to 1 minute long) captured from real-life situations. An updated version of this dataset was provided in ICDAR 2015 [KGBN+15] including a training set of 25 videos and a test set of 24 videos. This database includes a variety of Latin text such as French, English and Spanish.

The MSRA-TD500 dataset [YBL+12] works on multi-oriented scene text detection. This dataset includes 500 images (300 for training and 200 for testing) with horizontal and skewed texts in complex natural scenes. Some samples are illustrated in Figure 3.2(b). The method of Liu *et al.* [LLQ+17] achieved a state-of-the-art performance on this database with an F-score of 75%. This method employed the MSER technique and the AdaBoost classifier to extract text candidates and filter out non-text objects, respectively.

The Street View Text (SVT) dataset [WB10] is used for scene text detection, segmentation and recognition in outdoor images. It includes 350 images with 904 word-level annotated bounding boxes. The method of Gao *et al.* [GCWL17] showed superiority, on this dataset, over existing techniques with 83% as a word recognition accuracy. This method was based on a CNN model with attention mechanism. In the segmentation task, the best F-score, 90%, was obtained by Mishra *et al.* [MAJ17]. Their algorithm was mainly based on two steps: a graph cut procedure and a GMM refinement.

The NEOCR dataset [NDMW11] contains 659 natural scene images with multi-oriented text of high variability (see Figure 3.2(c)). This database is intended for scene text recognition and provides multilingual evaluation environments, as it includes text in eight European languages.

The KAIST dataset [LCJK10] consists of 3,000 images taken in indoor and outdoor scenes

(see Figure 3.2(d)). This is a multilingual dataset, which includes English and Korean text. KAIST can be used for both detection and segmentation tasks, as it provides binary masks for each character in the image. The text segmentation algorithm of Zhu and Zhang [ZZ17] outperformed existing methods on this dataset with an F-score of 88%. The method was mainly based on a superpixel clustering.
In 2016, Veit *et al.* [VMN+16] proposed a dataset, called COCO-Text, for English scene text detection and recognition. The dataset is based on the Microsoft COCO dataset, which contains images of complex everyday scenes. The best F-score (67.16%) on this dataset was achieved by the winner of the ICDAR 2017 COCO-Text competition [GSG+17] (detection task).
Recently, Chng and Chan [CKCS17] introduced a new dataset, namely Total-text, for curved scene text detection and recognition problems. It contains 1,555 scene images and 9330 annotated words with three different text orientations.

As for Arabic script, major contributions have been made in the conventional field of printed and handwritten OCR systems [LG06, MEA12]. A lot of progress of such systems has been triggered thanks to the availability of public datasets. One of the most widely used for offline Arabic handwritten recognition is the IFN/ENIT [PMM+02] dataset. In the same context, another dataset called KHATT was proposed by Mahmoud *et al.* [MAAK+14] and used in the writer identification competition of ICFHR 2014 [SAM+14]. The APTI database works on printed word recognition in low resolution images [SIK+09] and has been used as a benchmark in several competitions, like ICDAR 2013 [SKEA+13]. In online context, the community has created several datasets, such as ADAB [KTA+11] and AltecOnDB [AAAB15].

However, handling Arabic text detection and recognition for camera-based documents are limited to very little work, and most of the existing methods have been tested on private datasets with non-uniform evaluation protocols, which makes direct comparison and scientific benchmarking rather impractical.
Table 3.1 presents commonly used datasets for text processing in images and videos, and summarizes their features in terms of text category, source, task, language, and information of training/test samples. As depicted in this table, publicly available datasets for Arabic Video OCR systems are limited to one work for the recognition task and are even non-existent for detection and tracking problems. In 2015, Youssfi *et al.* [YBG15a] put forward a dataset for superimposed text recognition, called ALIF. This dataset was composed of 6,532 cropped text images extracted from diverse Arabic TV channels and with about 12% extracted from web sources. ALIF was limited to the recognition task and offered only one image resolution.

Accordingly, we have developed a new dataset of news video sequences containing artificial Arabic text. It is named AcTiV, for Arabic-Text-in-Video. We mainly targeted Arabic text detection and recognition systems, which require text transcription and layout coordinates of all text regions as ground-truth data. AcTiV differs from the previous datasets in three key aspects. First, AcTiV contains various types of multimedia documents ranging from news video clips to cropped text images, and this is suitable for systems operating in a stepwise methodology as well as for those based on integrated methodology, i.e. end-to-end systems. Second, these multimedia documents are collected from two sources: a Direct Broadcast Satellite (DBS) system and a video-sharing website (YouTube), and are in three different

Table 3.1: Most important existing datasets for text analysis.
'D', 'S', 'T' and 'R' respectively denote 'Detection', 'Segmentation', 'Tracking' and 'Recognition'.

Dataset (Year)	**Category**	**Source**	**Task**	**# of images (train/test)**	**# of text (train/test)**	**Language**
DARPA (1997)	Printed	Weblogs, newspapers (600 dpi)	R	345 (pages)		Arabic
IFN/ENIT (2002)	Handwritten	411 writers (300 dpi)	R	2,200	26,459 (city words)	Arabic
ICDAR'03 (2003)	Scene text	Camera	D/R	509 (258/251)	2,276 (1,110/1,156)	English
APTI (2009)	Printed word	Synthetically generated (72 dpi)	R	113,284	45,313,600	Arabic
ADAB (2009)	On-line handwritten	165 writers	R	937	15,158	Arabic
KAIST (2010)	Scene text	Camera, mobile phone	D	3,000	>5,000	English, Korean
OHASD (2010)	On-line handwritten	48 writers Tablet PC	R	145 (paragraphs)	3,825 (words)	Arabic
SVT (2010)	Scene text	Google Street View	D/S/R	350 (100/250)	904 (257/647)	English
ICDAR'11 (2011)	Scene text Born-digital text	Camera Web	D/R D/R	485 522	1564 4501	English English
NEOCR (2011)	Scene text	Camera	D/R	659	5,238	Eight languages
MSRA-TD500 (2012)	scene text	Camera	D	500 (300/200)	–	English, Chinese
ICDAR'13 (2013)	Scene text Artificial text Video scene	Camera Web Camera	D/S/R D/S/R D/T/R	229/233 410/141 28 videos	848/1,095 3,564/1,439 –	English English Spanish, French, English
KHATT (2014)	Handwritten	1000 writers 200-600 dpi	R/S	2000 (paragraphs)	9327 (lines)	Arabic
ALIF (2015)	Artificial text	Video frames	R		6,532 (4,152/2,199)	Arabic
COCO-Text (2016)	Scene text	MS COCO dataset	D/R	63,000	173,000	English
Total-Text (2017)	Curved scene text	Web	D/R	1,555 (1,255/300)	9,330 (words)	English

resolutions. Third, AcTiV has a larger scale than other datasets for text detection and recognition. In particular, our dataset has roughly twice the number of text annotations than the related dataset ALIF.

In the following, we describe our suggested dataset and related tools and protocols.

3.3 Description of AcTiV dataset

AcTiV is a real-content database where video clips are recorded from a DBS system and then transcoded and segmented into frames. In this section, we present the specificities of this dataset in terms of data acquisition, characteristics and statistics, annotation guidelines and data organization.

3.3.1 Data acquisition

TV channels broadcast a wide variety of programs ranging from talk shows and interviews to documentaries and weather reports. News reports are specifically picked for the present thesis. In order to ensure maximum diversities of content and avoid repetition, recordings from the same channel are spaced by at least one week. The video stream is initially saved

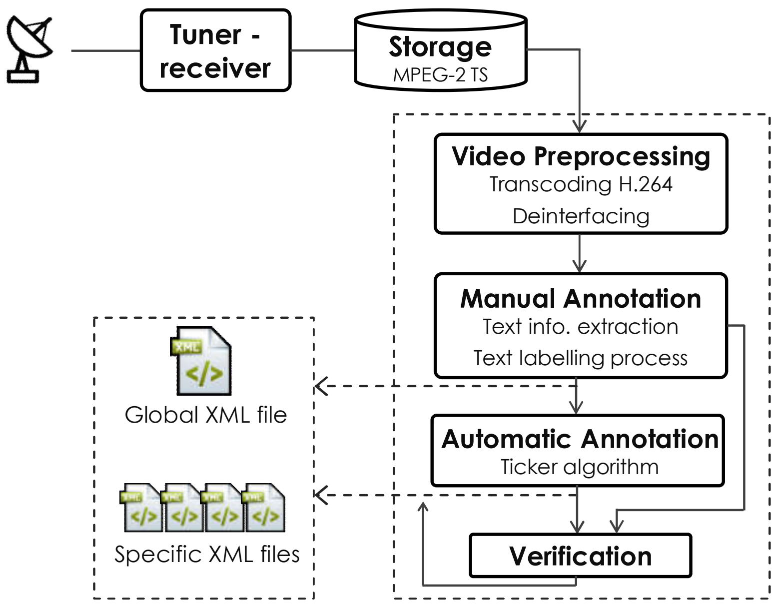


Figure 3.3: Data acquisition, video preprocessing and semi-automatic annotation of text regions

unaltered on the hard drive (MPEG-TS). After that, a transcoding process takes place to convert the interlaced (MPEG2/MPEG4) video to deinterlaced MPEG4-AVC using an x264-based encoder and applying a YADIF filter. The aim of this, is to prepare the video to a frame-by-frame analysis and to decrease the video bitrate without a perceived quality loss. In the present study, three different video-stream resolutions are chosen: HD (1920 x 1080, 25fps), SD (720 x 576, 25 fps) and SD (480 x 360, YouTube quality).
Figure 3.3 illustrates the data acquisition and video preprocessing steps as well as the annotation process, which will be detailed in section 3.3.3.

3.3.2 Characteristics and statistics

AcTiV was presented in the ICDAR 2015 conference [ZHT+15] as a first publicly accessible annotated dataset designed to support the development of new Arabic Video OCR systems. The challenges that have been addressed by AcTiV are the variability of text patterns (colors, fonts, sizes and positions) and the presence of complex background with various objects similar to text characters.

The AcTiV database is currently used by several research groups around the world. It enables users to test their system abilities to locate, track and recognize text in videos. The initial version of this dataset included 80 videos collected from four different Arabic news channels: TunisiaNat1 TV (ElWataniya1), France24 Arabe, RussiaToday Arabic and AljazeeraHD. This choice was based on the fact that it ensures maximum diversities of text areas in terms of font, size and background.
We have focused on text displayed as overlay in news videos, which can be classified into two categories, static and dynamic (or scrolling) text:

- Static text represents the type of text that does not undergo a change in its content, position, size, or color within its display interval, i.e. from a frame $_x$ to a frame $_{x+t}$. This category usually includes event information, names of people and places, and subtitles.

- Dynamic text targeted in our work refers to the horizontal scrolling text that usually resides in the lower third of the TV screen.

Figure 3.4: Samples of static texts (red rectangles) and dynamic texts (green rectangles) embedded in Arabic news video frames.

Figure 3.4 depicts samples of static and scrolling text from our dataset. Each video is around three to 12 minutes long. The maximal number of text blocks in one clip is 70. However, if we regard the same text block across multiple frames as separate text blocks, we have an average of 8,000 text blocks per clip.

Based on our preliminary experimental results and considering the AcTiV users' feedbacks, it was necessary to extend the content of data in terms of video clips and resolutions offering more training samples, particularly for deep learning-based methods.
The new version of AcTiV [ZTH+18] includes 189 video sequences, 10,415 text images and three video stream resolutions; i.e., the new added resolution is SD (480 x 360). Table 3.2 depicts the statistics of this dataset in terms of video clips per TV channel.

Table 3.2: Statistics of AcTiV dataset

Resolution	TV channel	# of video clips
HD (1920 x 1080)	AlJazeeraHD	43
SD (720 x 576)	France24 Arabe	47
	Russia Today Arabic	46
	TunisiaNat1	48
SD (480 x 360)	TunisiaNat1 YouTube	5

3.3.3 Annotation guidelines

Generally speaking, the creation of any standard database should undergo a manual or semi-automatic annotation process to generate ground-truth information for every text region. For instance, Siddiqi and Raza [SR12] proposed a semi-automatic text line labeling scheme for Urdu text localization in video images. The limitations of this method include the absence of ground-truth data to evaluate the performance of OCR systems, and the inability to annotate dynamic text. Several studies have made use of the freely available ground-truthing tool ViPER (Video Performance Evaluation Resource) [DM00] to annotate text objects in video content. However, this annotation methodology is dedicated to Latin and cannot be replicated for Arabic text. In our case, we use our own ground-truthing tool, AcTiV-GT [ZTH+14], to semi-automatically annotate the collected data.
The annotation process consists of two different levels: global and local.

- **The global annotation** process is performed manually through a user interface that includes a set of functionalities especially designed for annotating embedded text in video content. Figure 3.5 illustrates this user interface displaying a video frame being annotated. During the annotation process, we first load a video sequence. Then we draw a rectangular bounding-box for each textline that has a uniform size, alignment and spacing. Once a rectangle is selected, a new set of information will be created. It contains the following elements:

  - *position*: x, y, width and height.
  - *content*: text strings, text color, background color, background type (transparent, opaque)

  This set of information is saved in a meta file called global XML file. Our tool handles frame-by-frame playing, which allows the user to label the apparition and disapparition frame number for each textline. This information is stored in the global XML file as:

  - *aInterval*: apparition interval of the textline (Frame_S, Frame_E).

  As Arabic letters may have up to four shapes depending on their position in the word, and in order to have an easily accessible representation of Arabic text for future processing, it is transformed into a set of labels with a suffix that refers to the letter's position

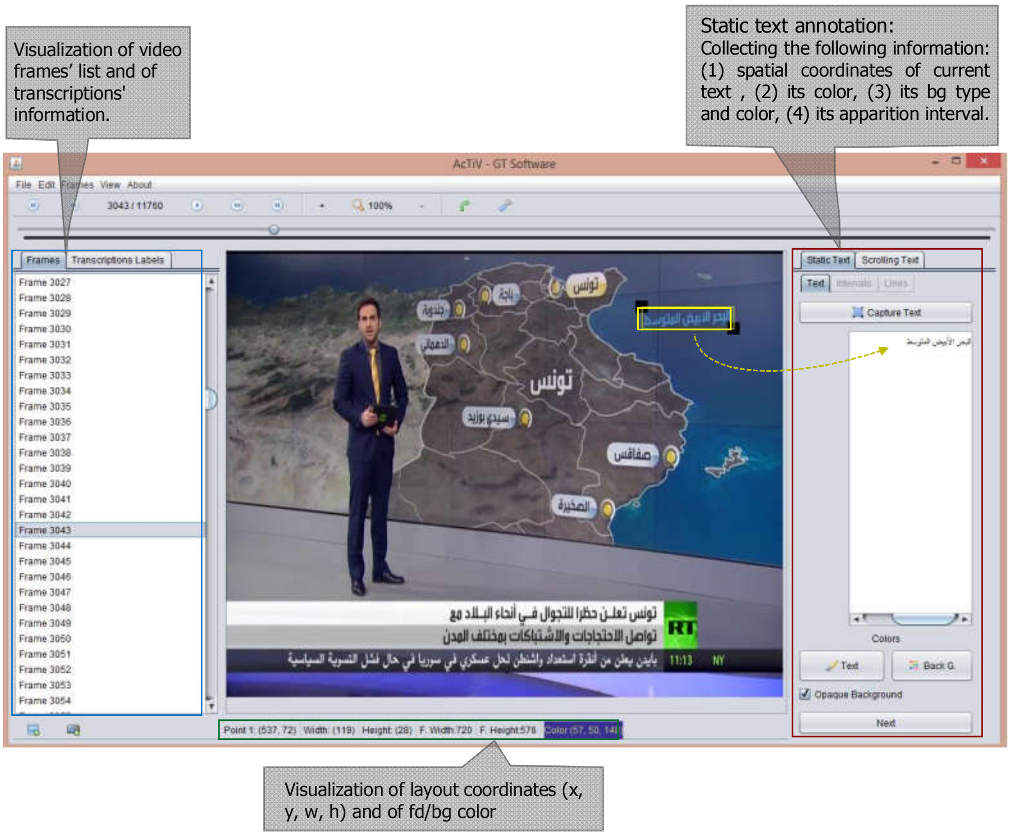


Figure 3.5: User interface of AcTiV-GT tool [ZTH+14] displaying labeled frame.

Figure 3.6: Arabic sentence and its corresponding labels.

in the word (B: Begin, M: Middle, E: End and I: Isolate). We adopt the same labels proposed in [SIK+09] to standardize the character labels for Arabic text. An example is depicted in Figure 3.6. A transcription label is generated for each appended Arabic text in the XML file. It is stored under the attribute *Latintranscription* within the same element that contains the Arabic text.

Furthermore, additional information, such as total number of textlines in the processed video clip and number of apparition intervals for each textline, is added in the global XML file. Figure 3.7 depicts an extract of a global XML file of AljazeeraHD TV channel. This file can be used as meta-data for text-tracking and end-to-end tasks.

Dynamic text is composed of scrolling series of tickers. To annotate this type of text, we note for each ticker its content, the first frame where the ticker appears, and the initial offset in the first frame, which is estimated using a virtual line. This information is stored in the *scrollingText* element of the global XML file. Figure 3.8 illustrates an example of these ground-truth data in addition to some channel specific information,

```xml
<?xml version="1.0" encoding="UTF-8"?>
- <video id="1" channel="AljazeeraHD" resolution="1080p" duration="00:09:17" fps="25"
  nbOfFrames="13925">
  - <staticText font="aljazeeraFont" nbOftextBox="17">
    - <textBox id="1" nbOfaInterval="1">
        <aInterval id="1" frame_S="1338" frame_E="1405"/>
        <position x="1161" y="909" width="268" height="63"/>
      - <content nbTextLines="1" textColor="51,65,90" bgColor="240,242,237" bgType="opaque">
          <textLine id="1" ArabicTranscription="حسن جمّول" LatinTranscription="Haaa_B Siin_M Nuun_E
            Space Jiim_B MiimChadda_M Waaw_I Laam_I"/>
        </content>
      </textBox>
    - <textBox id="2" nbOfaInterval="3">
        <aInterval id="1" frame_S="3371" frame_E="6691"/>
        <aInterval id="2" frame_S="6960" frame_E="7916"/>
        <aInterval id="3" frame_S="8329" frame_E="9833"/>
        <position x="1352" y="514" width="251" height="50"/>
      - <content nbTextLines="1" textColor="240,250,255" bgColor="11,93,93" bgType="opaque">
          <textLine id="1" ArabicTranscription="ريتشارد فولك" LatinTranscription="Raa_I Yaa_B Taaa_M
            Shiin_M Alif_E Raa_I Daal_I Space Faa_B Waaw_E Laam_B Kaaf_E"/>
        </content>
      </textBox>
    - <textBox id="3" nbOfaInterval="3">
        <aInterval id="1" frame_S="3372" frame_E="6691"/>
        <aInterval id="2" frame_S="6960" frame_E="7916"/>
        <aInterval id="3" frame_S="8329" frame_E="9833"/>
        <position x="1129" y="562" width="475" height="43"/>
      - <content nbTextLines="1" textColor="240,250,255" bgColor="0,24,66" bgType="opaque">
          <textLine id="1" ArabicTranscription="المقرر الخاص لمجلس الأمم المتحدة لحقوق الإنسان السابق"
            LatinTranscription="Alif_I Laam_B Miim_M Gaaf_M Raa_E Raa_I Space Alif_I
            Laam_B Xaa_M Alif_E Saad_I Space Laam_B Miim_M Jiim_M Laam_M Siin_E
            Space Alif_I Laam_EHamzaAboveAlif_E Miim_B Miim_E Space Alif_I Laam_B
            Miim_M Taaa_M Haaa_M Daal_E TaaaClosed_I Space Laam_B Haaa_M Gaaf_M
            Waaw_E Gaaf_I Space Alif_I Laam_EHamzaUnderAlif_E Nuun_B Siin_M Alif_E
            Nuun_I Space Alif_I Laam_B Siin_M Alif_E Baa_B Gaaf_E"/>
        </content>
      </textBox>
```


Figure 3.7: Example of global XML file: part of static text. This figure includes ground truth information about 3 textlines from a total of 17.


```xml
<scrollingText orientation="left-right" textColor="251,251,255" bgColor="" bgType="opaque"
   runningSpeed="6,770 pixel/frame">
  <bandPosition x="0" y="977" width="1432" height="65"/>
  <content nbOftickerInformation="56">
   <tickerInformation id="1" frame_S="252" offset="4" transcription="مصر: المجلس الأعلى للقوات المسلحة يفوض السيسي الترشح للرئاسة" transcriptionLabel="Miim_B Saad_M Raa_E Colon Space Alif_I Laam_B Miim_M Jiim_M Laam_M Siin_E Space Alif_I Laam_EHamzaAboveAlif_E Ayn_B Laam_M AlifBroken_E Space Laam_B Laam_M Gaaf_M Waaw_E Alif_I Taaa_I Space Alif_I Laam_B Miim_M Siin_M Laam_M Haaa_M TaaaClosed_E Space Yaa_B Faa_M Waaw_E Daad_I Space Alif_I Laam_B Siin_M Yaa_M Siin_M Yaa_E Space Alif_I Laam_B Taaa_M Raa_E Shiin_B Haaa_E Space Laam_B Laam_M Raa_E HamzaAboveAlifBroken_B Alif_E Siin_B TaaaClosed_E"/>
   <tickerInformation id="2" frame_S="443" offset="0" transcription="المجلس العسكري: لم يكن بوسعنا إلا الاستجابة لرغبة الجماهير في ترشيح السيسي" transcriptionLabel="Alif_I Laam_B Miim_M Jiim_M Laam_M Siin_E Space Alif_I Laam_B Ayn_M Siin_M Kaaf_M Raa_E Yaa_I Colon Space Laam_B Miim_E Space Yaa_B Kaaf_M Nuun_E Space Baa_B Waaw_E Siin_B Ayn_M Nuun_M Alif_E Space HamzaUnderAlif_I Laam_EAlif_E Space Alif_I Laam_EAlif_E Siin_B Taaa_M Jiim_M Alif_E Baa_B TaaaClosed_E Space Laam_B Raa_E Ghayn_B Baa_M TaaaClosed_E Space Alif_I Laam_B Jiim_M Miim_M Alif_E Haa_B Yaa_M Raa_E Space Faa_B Yaa_E Space Taaa_B Raa_E Shiin_B Yaa_M Haaa_E Space Alif_I Laam_B Siin_M Yaa_M Siin_M Yaa_E"/>
   ...
</scrollingText>
</video>
```


Figure 3.8: Example of global XML file: part of dynamic text. This figure illustrates ground-truth data about two out of 56 scrolling texts

.

e.g. *runningSpeed* and *bandPosition*.

- **The local annotation** is automatically performed at the frame level according to the information contained in the global XML file. Two appropriate types of XML files are generated, one for the detection task and the other for the recognition task. Figure 3.9


```
  <?xml version="1.0" encoding="UTF-8"?>
- <Protocol4 channel="France24">
  - <frame source="vd25" id="4">
      <rectangle id="1" x="75" y="452" width="51" height="25"/>
      <rectangle id="2" x="619" y="437" width="34" height="20"/>
      <rectangle id="3" x="271" y="457" width="381" height="35"/>
    </frame>
  - <frame source="vd25" id="30">
      <rectangle id="1" x="592" y="38" width="29" height="17"/>
      <rectangle id="2" x="631" y="37" width="27" height="18"/>
      <rectangle id="3" x="632" y="438" width="21" height="19"/>
      <rectangle id="4" x="325" y="462" width="327" height="30"/>
      <rectangle id="5" x="75" y="452" width="51" height="25"/>
    </frame>
  - <frame source="vd25" id="70">
      <rectangle id="1" x="556" y="34" width="103" height="21"/>
      <rectangle id="2" x="75" y="452" width="51" height="25"/>
      <rectangle id="3" x="622" y="438" width="31" height="19"/>
      <rectangle id="4" x="291" y="458" width="361" height="34"/>
    </frame>
  - <frame source="vd25" id="122">
      <rectangle id="1" x="75" y="452" width="51" height="25"/>
      <rectangle id="2" x="593" y="437" width="60" height="23"/>
      <rectangle id="3" x="219" y="461" width="433" height="31"/>
    </frame>
  - <frame source="vd25" id="154">
      <rectangle id="1" x="348" y="61" width="304" height="57"/>
      <rectangle id="2" x="110" y="213" width="159" height="50"/>
      <rectangle id="3" x="510" y="217" width="76" height="46"/>
      <rectangle id="4" x="198" y="278" width="72" height="17"/>
      <rectangle id="5" x="531" y="278" width="55" height="17"/>
      <rectangle id="6" x="110" y="315" width="122" height="51"/>
      <rectangle id="7" x="430" y="315" width="119" height="51"/>
      <rectangle id="8" x="75" y="452" width="51" height="25"/>
    </frame>
```


Figure 3.9: Extract of detection XML file of France24 TV channel.

depicts a part of a detection XML file of France24 TV channel. The ground-truth information are provided at the line level for each frame. One text bounding-box is described by the element *Rectangle* which contains the rectangle's attributes: (x, y) coordinates, width and height. As a side note, the result image and the ground-truth image should have the same name: *channel_source_frame_id*, e.g., TunisiaNat1_vd01_frame_7. The scrolling band should be hidden for AljazeeraHD and France24 channels before any processing since there are no ground-truth data for dynamic text in the detection dataset.

The recognition ground-truth file is provided at the line level for each textline image. The XML file is composed of two principal markup sections: *ArabicTranscription* and *LatinTranscription*. Figure 3.10 highlights an example of a ground-truth XML file and its corresponding textline image.

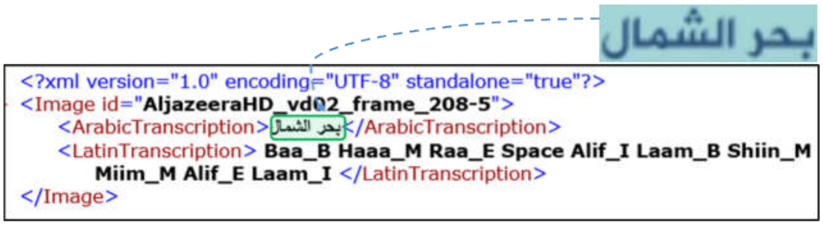


Figure 3.10: Recognition ground-truth file and its corresponding textline image.

3.3.4 Data organization

In addition to the video sequences and their annotation XML files, AcTiV includes two appropriate datasets, namely AcTiV-D and AcTiV-R, for detection and recognition tasks. Figure 3.11 illustrates the architecture and content of these datasets, where JHD, Fr24, RT, TN1 and TN1+ respectively denote AljazeeraHD, France24 Arabe, RussiaToday Arabic, TunisiaNat1 and TunisiaNat1 Youtube; and F, Ln, Wd and Ch respectively denote Frame, Line, Word and Character.

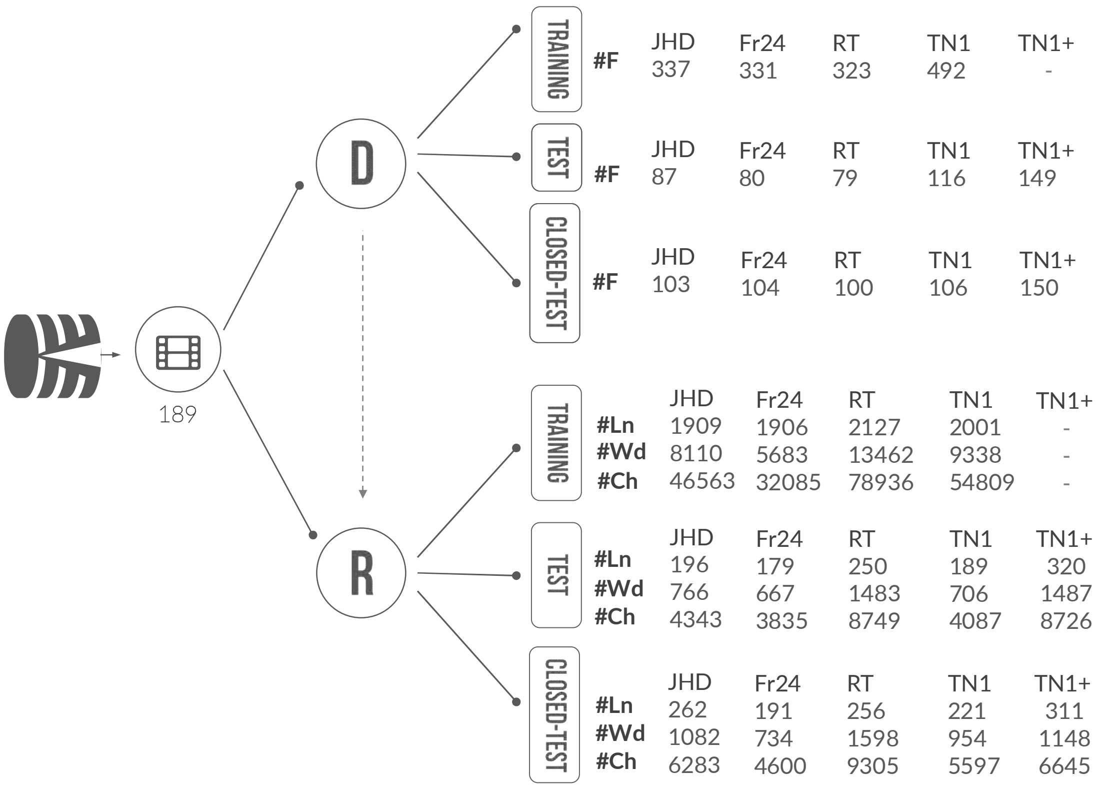


Figure 3.11: AcTiV architecture and statistics of detection (D) and recognition (R) datasets.

AcTiV-D

AcTiV-D represents a dataset of non-redundant frames used to measure the performance of existing methods for text detection in HD / SD video frames. These frames are hand-selected with a particular attention to achieve a high diversity in text regions. Figure 3.12 states examples from AcTiV-D for typical problems in video text detection. The latter dataset contains a total of 2,557 frames distributed over four sets (one set per channel). Every set includes three sub-sets for training, test and closed-test (used in competitions only). Table 3.3 gives an idea about the content of this dataset. In Figure 3.11, we present the statistics of AcTiV-D in the form of sub-sets in order to closely see the distribution of data. For instance, a set of 492, 116 and 106 frames are collected from TunisiaNat1 TV to serve as training, test and closed-test files, respectively.

For sake of testing the systems' ability to detect text under different situations, the proposed dataset includes some frames that do not contain any text and some others that contain the same text regions but with different backgrounds.

Figure 3.12: Typical video frames from AcTiV-D dataset. From left to right: examples of RussiaToday Arabic, France24 Arabe,TunisiaNat1 and AljazeeraHD frames.

Table 3.3: Statistics of AcTiV-D dataset

Resolution	TV channel	# of hand-selected frames
HD (1920 x 1080)	AlJazeeraHD	527
SD (720 x 576)	France24 Arabe	515
	Russia Today	502
	TunisiaNat1	714
SD (480 x 360)	TunisiaNat1 YouTube	299

AcTiV-R

AcTiV-R represents a dataset of cropped textline images used to assess the performance of Arabic text recognition systems. AcTiV-R texts are in various colors, unknown font sizes and font families, and with different degrees of background complexity. These text images cover a broad range of characteristics that distinguish video frames from scanned documents, as shown in Figure 3.13. AcTiV-R consists of 10,415 textline images, 44,583 words and 259,192 characters. Table 3.4 presents a general idea about the statistics of this dataset. More details per sub-set are depicted in Figure 3.11. For instance, RussiaToday TV respectively includes 2127, 250 and 256 lines as training, test and closed-test files for OCR systems.

Figure 3.13: Example of text images from AcTiV-R depicting typical characteristics of video text images.

Table 3.4: Statistics of AcTiV-R dataset

Resolution	TV channel	# of lines	# of words	# of characters
HD (1920 x 1080)	AlJazeeraHD	2367	9958	57189
SD (720 x 576)	France24	2276	7084	40520
	Russia Today	2633	16543	96990
	TunisiaNat1	2411	10998	64493
SD (480 x 360)	TunisiaNat1 YouTube	631	2635	15371

During the annotation process, we have taken into account 164 Arabic character forms:

- 125 letters, considering the "positioning" variability of each glyph.

- 15 additional characters combined with the diacritic sign "Chadda".

- 10 digits.

- 14 punctuation marks including the *white space*.

The different character labels can be seen in Table 3.5. The same table provides for each character its frequency in the dataset.

Table 3.5: Distribution of letters in AcTiV-R dataset

Character label	# of times	In Arabic	Character label	# of times	In Arabic
Alif	28433	ا	**HamzaAboveAlif**	1653	أ
Baa	7417	ب	**HamzaUnderAlif**	1049	إ
Taaa	8948	ت	**TildAboveAlif**	87	لآ
Thaa	851	ث	**HamzaAboveAlifBroken**	1022	ىء
Jiim	3270	ج	**HamzaAboveWaaw**	268	ؤ
Haaa	3976	ح	**LaamHamzaAboveAlif**	925	لأ
Xaa	1345	خ	**LaamHamzaUnderAlif**	563	لإ
Daal	6656	د	**LaamTildAboveAlif**	63	لآ
Thaal	459	ذ	**Space**	31458	
Raa	11460	ر	**Digit_0**	606	0
Zaay	1478	ز	**Digit_1**	655	1
Siin	6348	س	**Digit_2**	475	2
Shiin	2353	ش	**Digit_3**	276	3
Saad	2123	ص	**Digit_4**	306	4
Daad	1085	ض	**Digit_5**	248	5
Thaaa	2184	ط	**Digit_6**	203	6
Taa	481	ظ	**Digit_7**	138	7
Ayn	5989	ع	**Digit_8**	148	8
Ghayn	890	غ	**Digit_9**	116	9
Faa	4942	ف	**Point**	313	.
Gaaf	4443	ق	**Colon**	424	:
Kaaf	2999	ك	**Comma**	118	,
Laam	18868	ل	**Slash**	76	/
Miim	11907	م	**Percent**	101	%
Nuun	10027	ن	**QuestionMark**	8	?
Haa	2608	ه	**ExclamationMark**	12	!
Waaw	10614	و	**Quote**	445	"
Yaa	18153	ي	**Hyphen**	457	-
AlifBroken	1211	ى	**Add**	14	+
Hamza	696	ء	**ParenthesisO**	30	(
TaaaClosed	7239	ة	**ParenthesisC**	29	)
LaamAlif	1916	لاَ	**Bar**	13	|
			Overall	259,192	

3.4 Evaluation protocols and metrics

3.4.1 AcTiV protocols

As mentioned before, the proposed dataset is mainly dedicated to train and evaluate existing Arabic Video-OCR systems. Hence, to objectively compare and measure the performance of such systems under the same experimental conditions, we suggest a set of evaluation protocols. Table 3.6 presents the protocols.

Table 3.6: AcTiV evaluation protocols

Protocol	Resolution	Type of text instances	Task
1	1920 x 1080	Static	Detection
2	1920 x 1080	Static Scrolling	Tracking
3	1920 x 1080	Static	Recognition
4	720 x 576	Static	Detection
5	480 x 360	Static Scrolling	Tracking
6	480 x 360	Static	Recognition
7	All	Static	Detection
8	All	Static Scrolling	Tracking
9	All	Static	Recognition
10	All	Static Scrolling	End-to-end
11	All	-	TV logo detection

- **Protocol 1** aims to measure the performance of detection methods in HD frames.
- **Protocol 2** focuses on static and scrolling text detection and tracking in HD videos. This protocol requires that text lines are both detected correctly in every frame and tracked correctly over the video sequence.
- **Protocol 3** is dedicated to evaluate the performance of OCR systems to recognize text in HD frames.
- **Protocol 4** is similar to protocol 1, varying only in the channel resolution. All SD (720 x 576) channels in our database can be targeted by this protocol which is split in four sub-protocols: three channel-dependent protocols (4.1, 4.2 and 4.3) and one channel-free protocol (4.4).
- **Protocol 4bis** is dedicated to the last added resolution (480 x 360). The main idea of this protocol is to train a given system with SD (720 x 576) data, i.e. Protocol 4.3, and test it with different data resolution and quality.
- **Protocol 5** focuses on static and dynamic text detection and tracking in SD videos.

- **Protocol 6** is similar to protocol 3, differing only in the channel resolution. All SD (720 x 576) channels in our database can be targeted by this protocol which is split in four sub-protocols: three channel-dependent protocols (6.1, 6.2 and 6.3) and one channel-free protocol (6.4).

- **Protocol 6bis** is dedicated to the last added resolution (480 x 360). The aim of this protocol is to train a given system with SD (720 x 576) data, i.e. Protocol 6.3, and test it with varied data resolution and quality.

- **Protocols 7** is the generic version of protocols 1 and 4 where text detection is evaluated regardless of data quality.

- **Protocol 8** focuses on static and scrolling text detection and tracking in all video clips of AcTiV.

- **Protocol 9** is the generic version of protocols 3 and 6 where text recognition is assessed without considering data quality.

- **Protocol 10** is dedicated to measure the performance of end-to-end systems (simultaneous detection, tracking and recognition tasks) in a given video sequence.

- **Protocol 11** is meant for TV logo identification in video clips. Although it is unrelated to previous protocols, it can be very helpful as a preprocessing stage for other tasks to select the corresponding system depending on the TV channel.

3.4.2 Metrics

Text detection metrics

The evaluation of a text detection algorithm is generally based on two sets of information: a list G of ground-truth rectangles and a list D of detected ones. During the evaluation process, G and D are compared using a matching function. Final performance values are then calculated based on the well-known precision and recall metrics. Actually, such metrics are computed by measuring the overlap between the intersection area of two rectangles (G_i , D_i) and the area of G_i (recall score) or D_i (precision score), as expressed in Equations (3.1) and (3.2), respectively. If an algorithm detects too little text, its recall rate will decrease. Whereas, if it detects too many text regions, its precision rate will decline.

$$R_{ij} = \frac{Area(G_i \cap D_j)}{Area(G_i)} \tag{3.1}$$

$$P_{ij} = \frac{Area(G_i \cap D_j)}{Area(D_j)} \tag{3.2}$$

Indeed, in text detection the split and merge cases are very frequent; i.e., one G rectangle may correspond to more than one D rectangle, and vice-versa. In order to correctly match such sets of rectangles, several optimized algorithms have been proposed in the literature

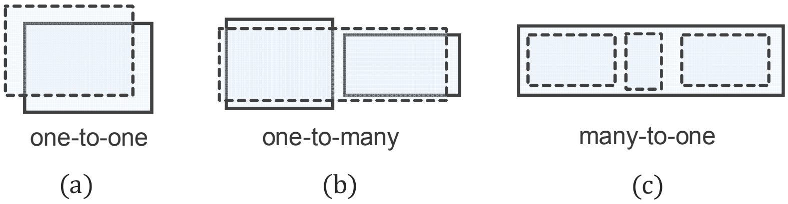


Figure 3.14: Different matching cases. G is represented by dashed rectangles and D by plain line rectangles.

[Luc05, KGS+09, AGP10]. Most of them take into account the case of one-to-one matching only. In our work, we use the matching strategy proposed in [LH97, WJ06]. Three different matching cases are considered:

- **One-to-one matching:** One rectangle $\mathrm{G_i}$ matches with one $\mathrm{D_j}$ if $\mathrm{R_{ij}} > t_r$ and $\mathrm{P_{ij}} > t_p$, where $t_r \in [0, 1]$ and $t_p \in [0, 1]$ are two quality constraints on area recall and area precision, respectively (see Figure 3.14(a)).

- **One-to-many matching (split case):** One rectangle $\mathrm{G_i}$ matches against a set of D rectangles if those latter cover a large portion (greater than t_r) of it and each of them overlaps enough with this $\mathrm{G_i}$, i.e. overlap greater than t_p (see Figure 3.14(b)).

- **Many-to-one matching (merge case):** One rectangle $\mathrm{D_j}$ matches against a set of G rectangles (see Figure 3.14(c)).

New measures are defined for this matching strategy, as expressed in Equations (3.3) and (3.4), respectively.

$$Recall = \frac{\sum_{i=1}^{|G|} matchG(G_i)}{|G|} \tag{3.3}$$

$$Precision = \frac{\sum_{i=1}^{|D|} matchD(D_i)}{|D|} \tag{3.4}$$

where $matchD$ and $matchG$ are functions that calculate the matching value depending on the quality of the match: 1 for one-to-one match, 0 for no match, and 0.8 for split and merge cases. The latter represents the amount of punishment in case of scattering.
A text detector algorithm can be evaluated using one single performance value, i.e. the harmonic mean of the precision and recall measures, also known as F-measure or F-score, given by Equation (3.5).

$$F\text{-}measure = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{3.5}$$

These metrics are calculated using our evaluation tool [ZTH+16]. Figure 3.15 shows the user interface of this tool. The same figure depicts a split case, where the ground-truth object is represented by a dashed rectangle and the detection results are in plain line rectangles. The

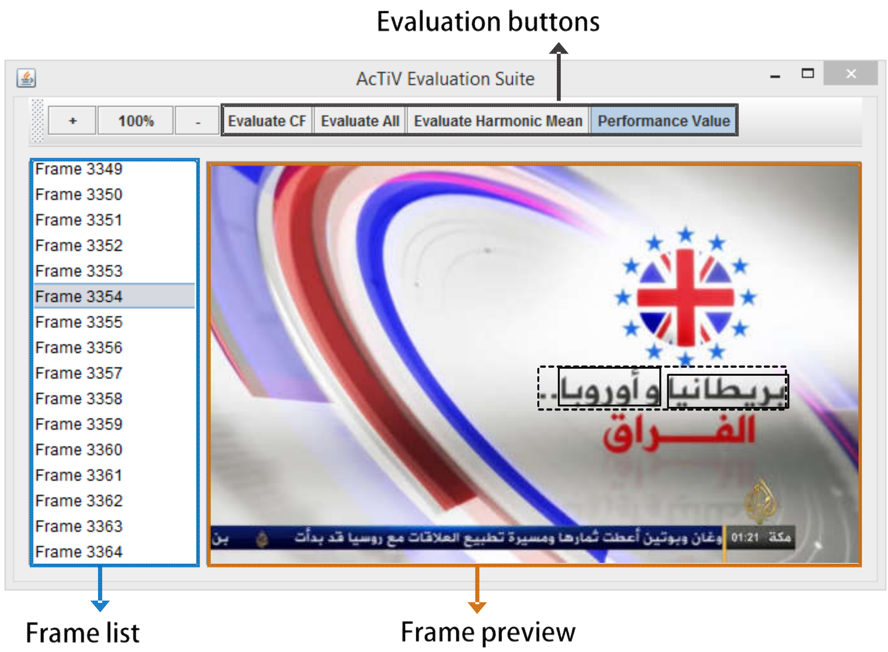


Figure 3.15: User interface of AcTiV-Eval tool.

user can apply the evaluation procedure to the current frame (by clicking on the "Evaluate CF" button) or all video frames (by clicking on the "Evaluate All" button). The "Performance Value" button displays precision, recall and F-measure values. The precision and recall curves

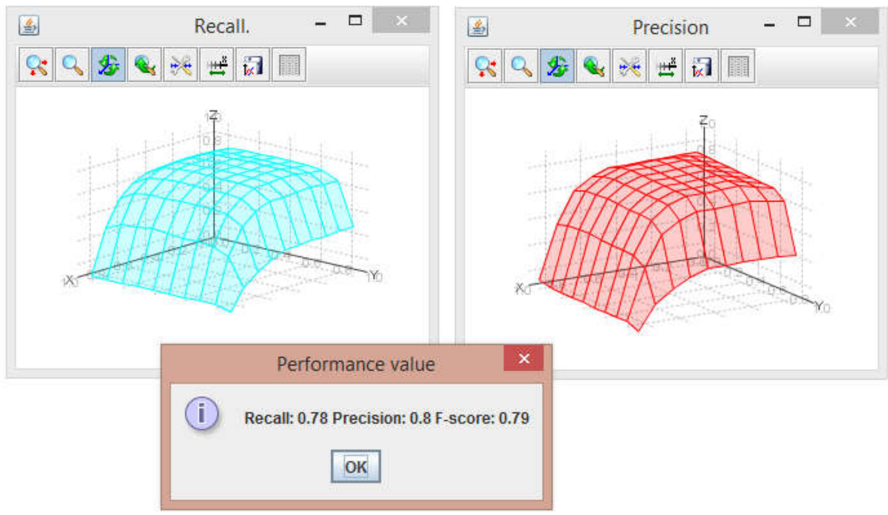


Figure 3.16: AcTiV-Eval output.

are illustrated in Figure 3.16, where x-axis denotes t_r values and y-axis denotes t_p values (precision and recall values by varying t_r and t_p from 0 to 1 by a step of 0.1). This helps choosing a good threshold value to decide whether a rectangle is correctly detected or not. In our evaluation process, the recall and precision thresholds (t_r and t_p) are set to 0.7 and 0.4, respectively.

Text tracking metrics

As in the ICDAR'13 and ICDAR'15 RRCs [KSU+13, KGBN+15], we suggest to use the metrics of VACE Framework [KGS+09] for measuring the performance of tracking systems. These metrics are Multiple Object Tracking Precision (MOTP), given by Equation 3.6, and Multiple Object Tracking Accuracy (MOTA), given by Equation 3.7. In our work, the main goal of a text tracking scheme is to determine the appearing / disappearing frame for each text (static or scrolling) in a given video clip, as explained before in Protocol 2.

$$MOTP = \frac{\sum_{i,t} o_t^i}{\sum_t c_t} \tag{3.6}$$

where ${o^i}_t$ refers to the overlapping ratio of the ith correspondence in the mapping π_t and c_t is the number of correspondences in π_t.

$$MOTA = 1 - \frac{\sum_t (FN_t + FP_t + IDSW_t)}{\sum_t g_t} \tag{3.7}$$

where FN_t, FP_t, $IDSW_t$, and g_t respectively refer to the number of false negatives, false positives, ID switches, and ground-truth texts at frame t.

Text recognition metrics

The performance measure for the recognition task is based on the Line Recognition Rate (LRR) and the Word Recognition Rate (WRR) at the line and word levels, respectively, and on the computation of Insertion (I), Deletion (Dl) and Substitution (S) errors at the level of Character Recognition Rate (CRR), which are given by Equations (3.8), (3.9) and (3.10).

$$CRR = \frac{\#characters - I - S - Dl}{\#characters} \tag{3.8}$$

$$WRR = \frac{\#words\_correctly\_recognized}{\#words} \tag{3.9}$$

$$LRR = \frac{\#lines\_correctly\_recognized}{\#lines} \tag{3.10}$$

Figure 3.17 presents an example explaining the impact on CRR and WRR metrics resulting from substitution and deletion errors. In some work, the recognition performances have been evaluated based on the computation of error rates, i.e. Character Error Rate (CER), Word Error Rate (WER) and Line Error Rate (LER), which is the same in practice.

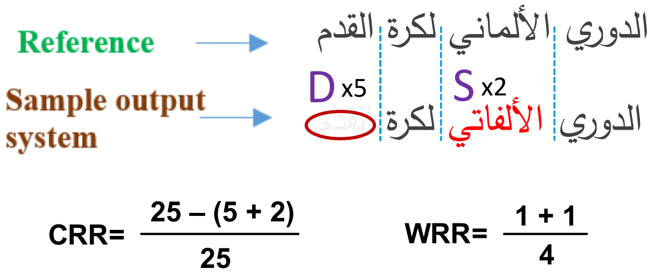


Figure 3.17: Example of CRR and WRR computation based on different system output errors

3.5 Conclusion

We have presented in this chapter the AcTiV dataset for the development and evaluation at a large-scale of Video-OCR systems. This dataset particularly addresses the problems of text detection and recognition, which are essential stages in the whole end-to-end recognition module, by providing two appropriate datasets: AcTiV-D and AcTiV-R.

AcTiV is freely available to research institutions. We have provided details about the characteristics and statistics of the database. We have also reported about our ground-truthing tool used to semi-automatically annotate the video clips and our text detection evaluation tool. Additionally, a set of evaluation protocols has been made to measure the systems' performance under different situations.

Chapter 4

Text Detection by SWT and Auto-encoders

4.1 Introduction

A preliminary step to Video-OCR processing is the detection of text areas in video frames. Nevertheless, text detection is a challenging problem due to the complexity of video content (text pattern variability, low resolution, cluttered background, etc). Particularly for Arabic text, several additional difficulties are present. Compared to Latin, Arabic text has more strokes in different directions, various character aspect ratios, and more diacritics above and below characters.

In this chapter, we present our hybrid approach for Arabic text detection in video frames. The originality of this approach revolves around the combination of two techniques, an adapted version of the SWT algorithm and a Convolutional Auto-Encoder (CAE).

We aim in this work to stand out from the dominant methodology, based on so-called hand-crafted features. This is done by automating the feature extraction process, using CAE

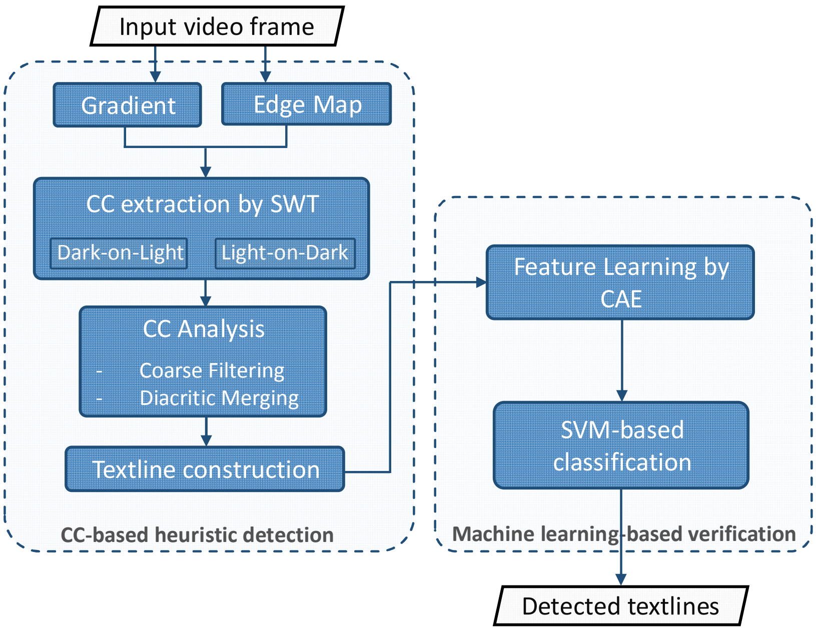


Figure 4.1: Flowchart of the proposed text detection approach.

as an unsupervised feature learning scheme.

The following section gives details about the proposed approach. Section 4.3 provides the experimental results in terms of parameter settings, discussion and comparison. Section 4.4 concludes the chapter and summarizes the principal findings.

4.2 Proposed text detection approach

Our method consists of two main stages, i.e. CC-based heuristic detection and machine learning verification, as shown in Figure 4.1. The first stage extracts, filters and groups CC text candidates based on the SWT operator, a set of geometrical constraints and a suggested textline construction technique. The second stage exploits CAE to automatically produce features, using previously obtained textline candidates as training data. An SVM classifier receives these features as an input for discriminating textlines from non-text ones.

4.2.1 Connected component-based heuristic detection

Preprocessing and edge detection

To perform SWT, an edge map and X & Y gradients are required. Before calculating these, we blur the input grayscale image using the Gaussian filter in order to increase robustness against noise. For the edge map, we exploit a 3 x 3 filter matrix and perform the Canny edge detection with empirical thresholds of 175 and 320. For the X & Y gradients, we use the Sobel operator.

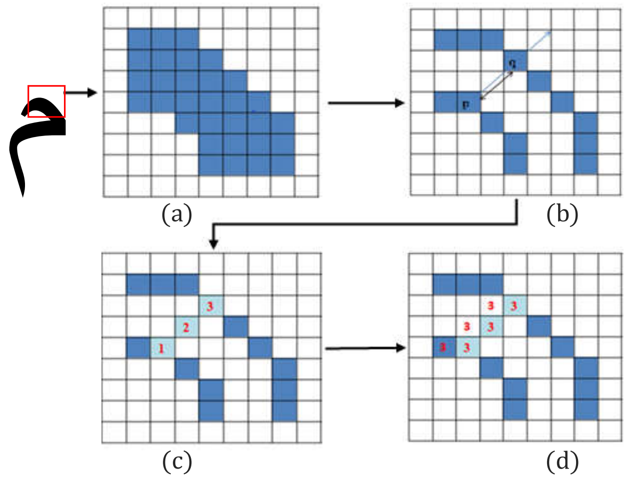


Figure 4.2: Stroke Width Transform.
(a) Zoom on upper right part of the Arabic character Miim م
(b) Shooting pixel ray between two opposing gradients $< p, q >$
(c) Counting number of pixels belonging to this ray
(d) Labeling these pixels by the value of distance between p and q

Component extraction by SWT

As mentioned before, the SWT [EOW10] operator is employed at this step for its efficiency in text component extraction from both scene images and video frames. It detects stroke pixels by shooting a search ray $r = p + n * d_{\mathrm{p}}$ $(n > 0)$ from an edge pixel p to its opposite one q along the gradient direction d_p. If these two edge pixels have nearly opposite gradient orientations, ray r will be considered as valid. All pixels inside this ray are labeled by the length $|p - q|$ (as shown in Figure 4.2(b)), and the input frame is consequently transformed into a SWT map. A constraint is defined in Equation (4.1) to verify whether the gradient direction is approximately opposite and the gradient magnitude is identical.

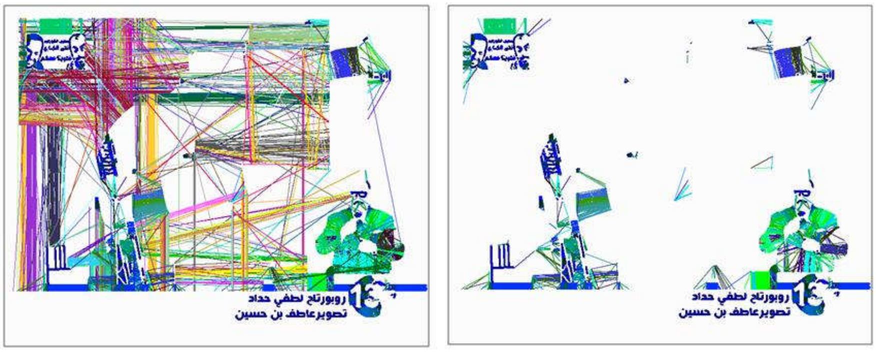
$$\|d_{\mathrm{p}} - d_{\mathrm{q}}\| \leq \frac{\pi}{6} \tag{4.1}$$

In this way, SWT filters out background pixels and assigns text pixels with stroke widths. However, such an operator is quite sensitive to edge defection, which leads to several false rays (Figure 4.3(a)). To reduce the amount of incorrect connections, we propose to discard the rays whose length is higher than a predefined threshold $\mathrm{T_r}$, which is selected according to a structural analysis of text strokes. See Figure 4.3(b) for an illustration.

It is worth noting that to deal with both dark-on-light text (DL) and light-on-dark text (LD) scenarios, we need to apply the SWT algorithm twice —in gradient and counter gradient directions —and merge the results of both directions.

Finding connected components

After calculating the stroke widths, we group the pixels in the resulting SWT map into CCs.



(a) (b)

Figure 4.3: Restriction on length of rays. (a) Results of original SWT (b) Our modified version.

Indeed, adjacent pixels are grouped if the ratio of their stroke widths is less than 3 and their *Lab* color distance is less than 30, to ensure uniformity in both stroke width and color. Note that if a pixel is labeled more than once, the minimal value will be assigned to it. In the Arabic alphabet, one character may consist of several strokes and consequently several labels. Thus, in order to give a unique label for each character, we propose a CC-labeling algorithm, which allows scanning the pixels of the SWT image in four directions (Figure 4.4). We denote by X the pixel to be treated. We assign to X the minimal label value of its neighboring pixels (colored in blue for each case). At each scan, the new values are displayed in red. Figure 4.5(b) shows the result of this algorithm for the character Yaa (ي).

Component analysis

Coarse filtering

At this step, we apply a set of heuristic constraints based on the spatial characteristics of text to filter out non-text components and background outliers. In these rules, a candidate component C is described by a set of geometrical measurements: H(C), W(C), coorX(C), coorY(C), SWM(C) and SWV(C), which respectively denote the height, width, X-coordinates, Y-coordinates, SW mean and SW variance of a component. The involved constraints are defined as follows:

$$\begin{aligned} H(C) &< 40\ px \\ W(C) &< 150\ px \\ 0.5 < \frac{W(C)}{H(C)} &< 5 \\ SWV(C) &< \frac{1}{2} \cdot SWM(C) \\ coorX(C) &< \frac{8}{10} \cdot ImageWidth \\ \frac{1}{10} \cdot ImageHeight < coorY(C) &< \frac{9}{10} \cdot ImageHeight \end{aligned} \tag{4.2}$$

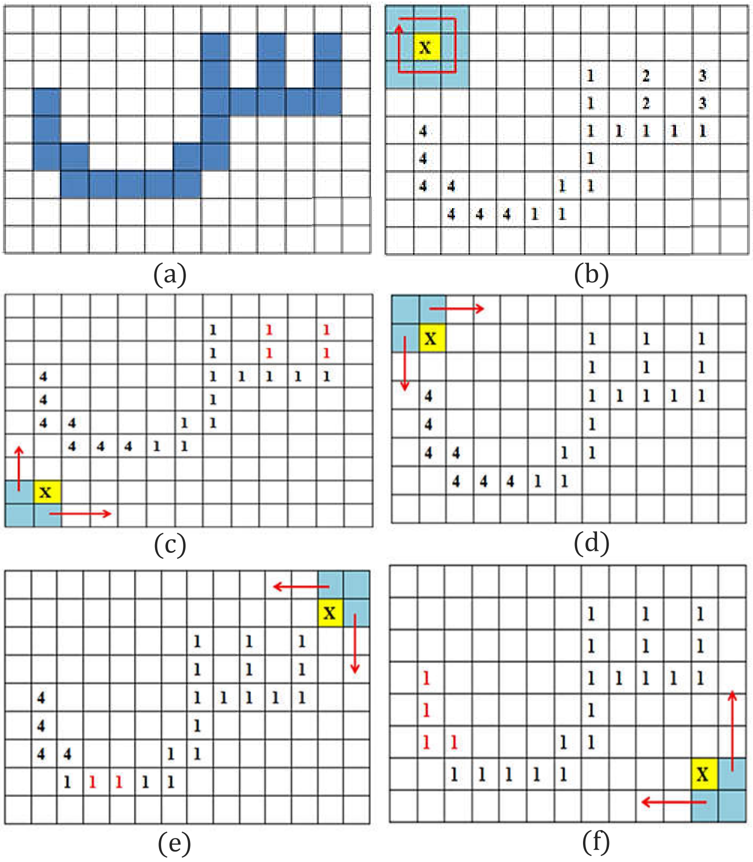


Figure 4.4: Example of CC labeling. (a) SWT map of letter Siin. (b) Output of labeling algorithm by [EOW10]. (c) - (f) Scans and output of proposed labeling algorithm.

First, we discard very big and tiny objects by limiting the width and height of the candidate component to 150 pixel and 40 pixel for SD resolution, and to 330 pixel and 90 pixel for HD resolution, respectively. Second, we remove the components with too large and too small aspect ratios under a conservative threshold $[0.5 - 5]$ so that characters like Alif (ا) will not be discarded. Third, a CC will be discarded if its SWV exceeds an empirically fixed threshold. A high value of SWV means that the CC consists of foliage, bricks or fences which are commonly mistaken as text in video frames. Finally, the objects located at the border of the image are also discarded from further processes. The rules defined in this step are weak conditions, so as to preserve the text components in a higher priority rather than filter background noise. Therefore, only the obvious non-text components are rejected. The remaining false alarms will be handled in the verification stage.

Diacritic merging

Several Arabic characters include diacritic marks like dots and Hamza. Thus, among the previously obtained CC candidates, some of them are parts of the same character, which need to be merged into one single bounding box. We design the following rules to group these CCs:

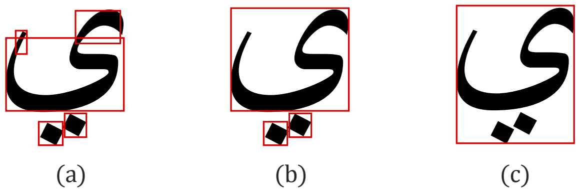


Figure 4.5: Labeling and merging of letter Yaa. (a) Result of the [EOW10] labeling algorithm. (b) Result of proposed labeling algorithm. (c) Vertical merging of diacritics (two dots).

- The vertical distance between two components, $\mathrm{C_i}$ and $\mathrm{C_j}$, should not exceed an empirically fixed threshold $\mathrm{T_{vd}}$.
- $\mathrm{C_i}$ and $\mathrm{C_j}$ should have a similar stroke width value; i.e., the ratio between their SW has to be less than 2.0.

Figure 4.5(c) shows the updated bounding box that results in applying this merging procedure for the character Yaa (ي).

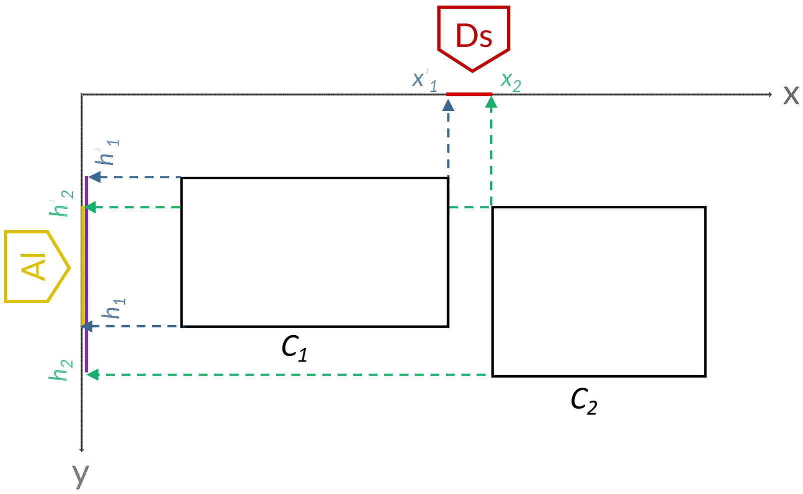


Figure 4.6: Alignment and distance between two components C1 and C2

Textline construction

We propose in this step a grouping method to correctly form textline candidates out of a huge set of components. Specifically, we define an upper triangular matrix M, where $\mathrm{m_{ij}}$ is the matching score corresponding to a pair of components ($\mathrm{C_i}$, $\mathrm{C_j}$).

$$M = \begin{bmatrix} m_{11} & m_{12} & m_{13} & \dots & m_{1n} \\ 0 & m_{22} & m_{23} & \dots & m_{2n} \\ & & \dots\dots\dots\dots & & \\ 0 & 0 & 0 & \dots & m_{nn} \end{bmatrix}$$

In order to compute m_{ij} for a given pair (C_i, C_j), we firstly calculate the following score functions:

- *$O_v(C_i,\ C_j)$*: score based on the spatial overlap between their corresponding rectangles R_i and R_j.

- *$D_s(C_i,\ C_j)$*: score based on the proximity of R_i and R_j. The closer R_i and R_j are, the more important $D_s(C_i,\ C_j)$ is. Let x_i and x'_i (resp. x_j and x'_j) be the X-coordinates of the left and right points of R_i (resp. R_j), and let w_i (resp. w_j) denotes its width. The horizontal distance Ds is then calculated using Equation (4.3).

$$Ds(C_i, C_j) = 2^{-\frac{dist}{max_h}} \tag{4.3}$$

  where $dist =\ (max_{x'} - min_x) - (w_i + w_j)$, which is depicted in Figure 4.6 by a red line in the X-axis, and max_h represents the height of the biggest component.

- *$A_l(C_i,\ C_j)$*: score based on component alignment. Let y_i and y'_i (resp. y_j and y'_j) be the Y-coordinates of the upper side and bottom side of R_i (resp. R_j), and let h_i (resp. h_j) denotes its height. The vertical alignment function Al is given by Equation (4.4).

$$A_{\mathrm{l}}(C_i, C_j) = \frac{min_h - max_{h'}}{max_h - min_{h'}} \tag{4.4}$$

  If we take the two components illustrated in Figure 4.6 as an example, $Al(C_1, C_2)$ will be equal to $(h_1 - h'_2)/(h_2 - h'_1)$. Generally speaking, the ideal case would be that both C_i and C_j have the same areas, while the worst case would be that they do not intersect, therefore Al would be equal to 0.

- *$Sw(C_i,\ C_j)$*: score based on the SW similarity. It is given by Equation (4.5).

$$Sw(C_{\mathrm{i}}, C_{\mathrm{j}}) = 2^{-|SWM_i - SWM_j|} \tag{4.5}$$

The probability matrix M is then calculated as follows:

$$\mathrm{m_{i,j}} = \begin{cases} \mathbf{1} & \textbf{if } Ov(C_{\mathrm{i}}, C_{\mathrm{j}}) > T_{\mathrm{ov}} \\ \mathrm{s} & \textbf{if } Ds(C_{\mathrm{i}}, C_{\mathrm{j}}) > T_{\mathrm{ds}} \textbf{ and} \\ & Al(C_{\mathrm{i}}, C_{\mathrm{j}}) > T_{\mathrm{Al}} \quad \textbf{and} \\ & Sw(C_{\mathrm{i}}, C_{\mathrm{j}}) > T_{\mathrm{Sw}} \\ \mathbf{0} & otherwise \end{cases}$$

where T_{ov}, T_{ds}, T_{Al} and T_{Sw} are predefined thresholds for the overlap ratio, distance, alignment and SW scores, respectively, of (C_i, C_j). Subsequently, s is determined by Equation (4.6).

$$s = \frac{Ds(C_{\mathrm{i}}, C_{\mathrm{j}}) + Al(C_{\mathrm{i}}, C_{\mathrm{j}}) + Sw(C_{\mathrm{i}}, C_{\mathrm{j}})}{3} \tag{4.6}$$

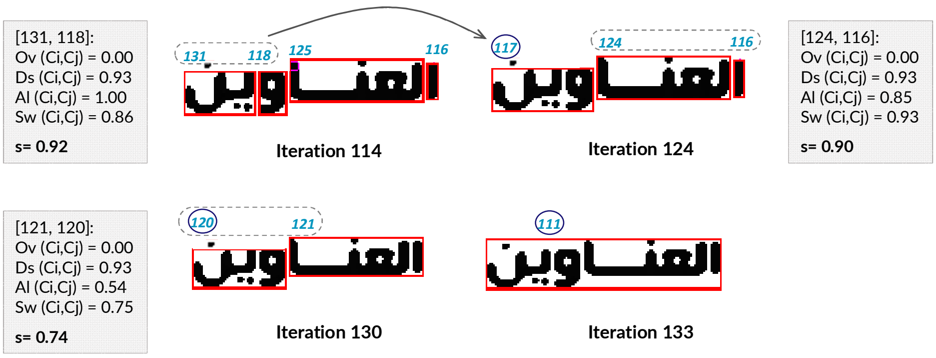


Figure 4.7: Updated CCs after the textline construction step

The line construction process consists finally in pairing $\mathrm{C_i}$ and $\mathrm{C_j}$ when $\mathrm{m_{ij}} = \max(\mathrm{M})$ with respect to a minimal matching score threshold $\mathrm{T_m}$. The gray boxes in Figure 4.7 represent examples of such a pairing, where score s of the most likely pair to be grouped ($\mathrm{C_i}$, $\mathrm{C_j}$) is presented for each iteration. The process ends when no component can be merged.

4.2.2 Machine learning-based verification

In this stage, we utilize CAE to generate features learned from previously obtained textline candidates. Afterwards, an SVM classifier receives these features as an input in order to distinguish text lines from non-text ones.

Feature learning by CAE

Machine learning methods take features as an input and return a class label as an output. There are various ways to extract features from data, for example by hard-coding mathematical or morphological operators. In this thesis, we aim to automatize this task by utilizing CAE for unsupervised feature learning.

Auto-Encoders (AE) represent a family of neural networks for which the input is the same as the output. They work by compressing the input data (e.g. image) into a lower-dimensional representation and then reconstructing the output from this representation. AE is composed of two parts, the first is called encoding and the second one is decoding, which can have multiple layers. However, for the sake of simplicity, we consider that each of them has only one layer (Figure 4.8).
Training an AE is unsupervised in the sense that no labeled data are needed. The training process is based on the back-propagation algorithm, which minimizes the average difference between the input x and its reconstruction at the output $\hat{x}$.

CAE are stacked auto-encoders where layers, except for the top one, are convolved. This allows covering a larger area while keeping the number of weights of the neural network small enough to have an acceptable training time. The output of CAE, which is used as features during classification, is the encoded values (i.e. compressed representation in Figure 4.8) not the result of the reconstruction, as the latest is used only during the training phase.

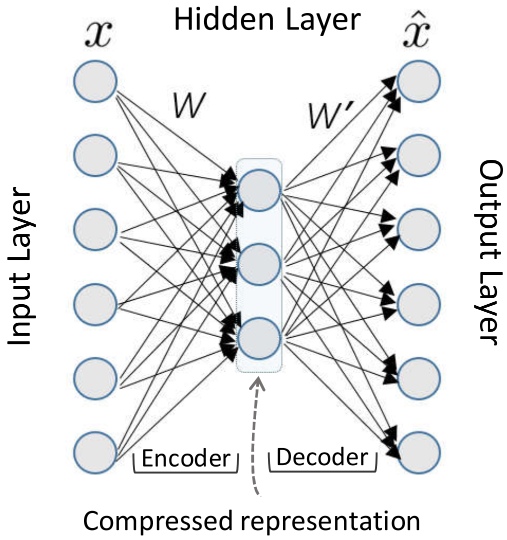


Figure 4.8: A basic auto-encoder with one hidden layer.

The unsupervised CAE feature learning method introduced in [SIL16] is used to learn features in this thesis. Each of its layers is composed of a convolved artificial neural network that has two neuron layers, one for encoding and one for decoding. A single-layer CAE is illustrated in Figure 4.9. The first neural layer encodes the inputs, and the second one, which is used only during the training phase, reconstructs the inputs from the encoded values. For stacked auto-encoders, the first layer of CAE takes raw pixel data as an input; the other layers take as an input the output of the previous layer. We use the soft-sign as an activation function for two reasons: (1) It is very fast, and (2) its derivative converges polynomially towards zero, which is more interesting during the training phase than the activation function whose derivative converges exponentially. The soft-sign activation is given by Equation (4.7).

$$f(x) = \frac{x}{|x| + 1} \tag{4.7}$$

Our CAE encodes an input x of dimension n to an output $\hat{x}$ of dimension m as described by Equation (4.8).

$$\hat{x}_i = f\left(\sum_{j=1}^{n}(w_i^e j \ . \ x_{\mathrm{j}}) + b_i^e\right) \tag{4.8}$$

where b_i is a bias, and w^e are the weights used for encoding.

Decoding an output y to reconstruct the input is done in a similar way:

$$\chi_j = f\left(\sum_{i=1}^{m}(w_j^d i \ . \ \hat{x}_{\mathrm{i}}) + b_j^d\right) \tag{4.9}$$

where χ is an approximation of the x vector encoded by $\hat{x}$, b_j is a bias, and w^d are the weights used for decoding. This means that CAE has to learn $m \times (n+1) + n \times (m+1)$ weights

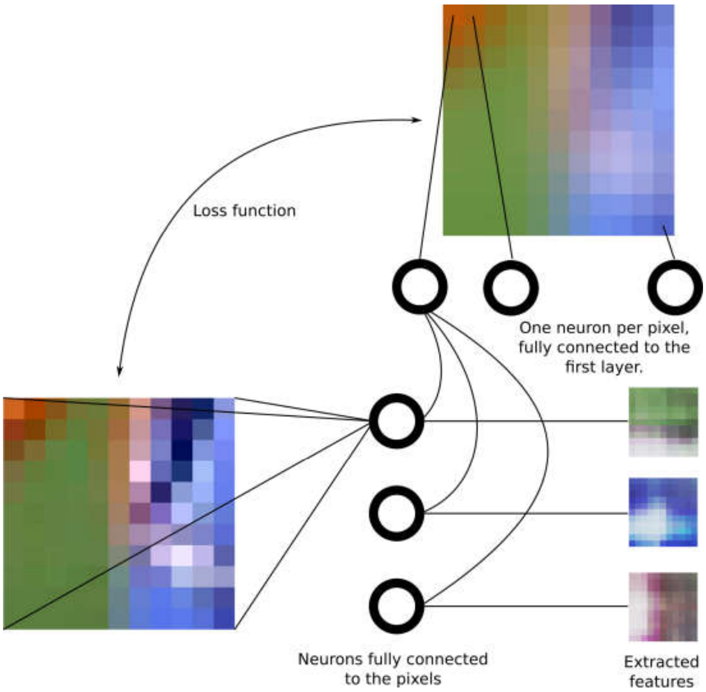


Figure 4.9: Illustration of single-layer CAE.

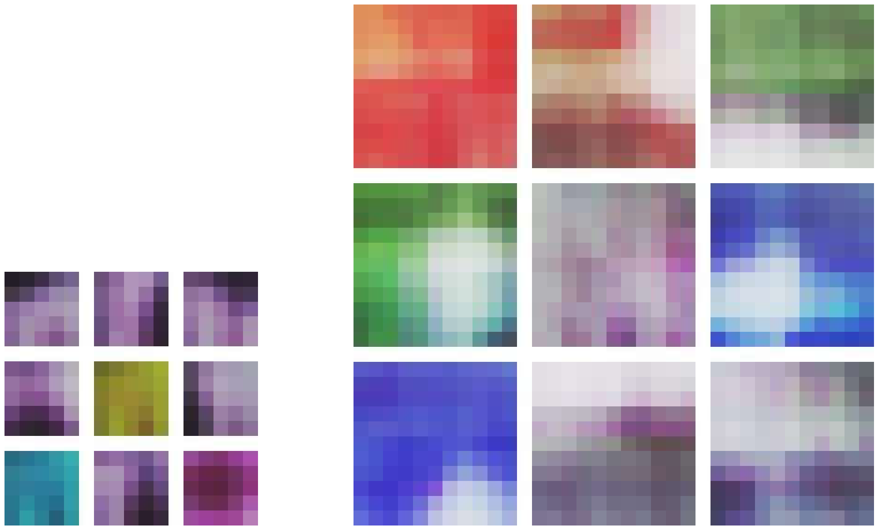
during its training. For this reason, it is more time-efficient to use a convolution of small AEs rather than train a single one covering a large patch.

The convolutions are created as follows. First, an AE covering $W_1 \times H_1$ pixels and having m_1 outputs is trained. After that, we create $W_2 \times H_2$ copies of it, and put them in a grid, with an offset of $O_1x \times O_1y$ pixels. This grid then covers $(O_1x \times \ (W_2 - 1) + W_1) \times (O_1y \times (H_2 - 1) + H_1)$ pixels. The output of the AEs in this grid can be seen as an array composed of $W_2 \times H_2 \times m_1$ values, which can be given afterwards to a second-level AE. When creating a convolution of second-level AE, i.e. to add a third level, the convolution of the first-level AE must be redimensioned accordingly.

The layers of CAE are trained one after another with standard back-propagation and gradient descent in their two-layer neural network, to minimize $(\chi - x)^2$. The layers of CAE must learn to encode and decode their own input. If we back-propagate the reconstruction error of the top-layer to the previous layers, then the top layer will "ask" through a back-propagation the previous layers to have easy-to-reconstruct values (e.g. constants). This will lead to a degeneration of weights, making the AE useless. For this reason, we add a new layer to the network only when its current top-layer is sufficiently trained. While CAE can be trained without supervision, its topology has to be manually defined, i.e. the number of layers, the size of convolutions, the number of features and the offset.

Feature visualization
We can display the i-th feature learned by CAE by manually setting its outputs to zero, except for $\hat{x}_i$ which is set to 1, and then decoding it layer after layer until reaching the pixel-level.

Figure 4.10 depicts some features, which are automatically learned by CAE. We can see that the learned patterns are more complex when there are more layers.



(a) (b)

Figure 4.10: Illustration of features learned by two CAE layers. (a) First layer, (b) Second layer

SVM-based classification

SVMs were proposed by Vapnik [CV95] and have yielded excellent results for several two-class classification problems and nonlinear regression. The main strength of such a classifier is that it is easy to train, it needs few training samples, and it has a good generalization ability that makes it effective for text identification. SVMs use the structural risk minimization to find the hyperplane that optimally separates two classes of objects. This hyperplane is computed as described by Equation (4.10).

$$f(x) = sgn(\sum_{i=1}^{n} y_i \alpha_i K(x, x_i^*) + \alpha_0) \tag{4.10}$$

where sgn is a sign function, K is a kernel function, and y ={-1,1} is the class label of the data point x. Moreover, x_i^* are support vectors and define the separating hyperplane. The parameters α_i $(0 < i < n)$ are optimized during training. The kernel function used in this thesis is the Radial Basis Function (RBF) expressed by Equation (4.11).

$$K(X, X_j) = \exp\{\frac{-|X - X_j|^2}{2\sigma^2}\} \tag{4.11}$$

where σ denotes the kernel bandwidth, which is determined through cross-validation experiments.

We train SVM with the extracted CAE features. We roughly select as many patches from text candidates as from non-text ones in order to have balanced training data. To make it clearer, let R1, R2 and R3 be three training positive samples with various sizes: T1, T2 and T3 (in terms of pixels). To extract an $\mathrm{N_i}$ number of patches per rectangle considering its size, we use the Equation (4.12).

$$N_{\mathrm{i}} = \frac{T_{\mathrm{i}}}{T_{\mathrm{g}} \times N_{\mathrm{tot}}} \tag{4.12}$$

where $T_{\mathrm{g}} = T_1+T_2+T_3$ is the total surface of positive samples, and $N_{\mathrm{tot}} = N_{\mathrm{T1}}+N_{\mathrm{T2}} + N_{\mathrm{T3}}$ is the total number of patches to extract from the positive (or negative) training set.

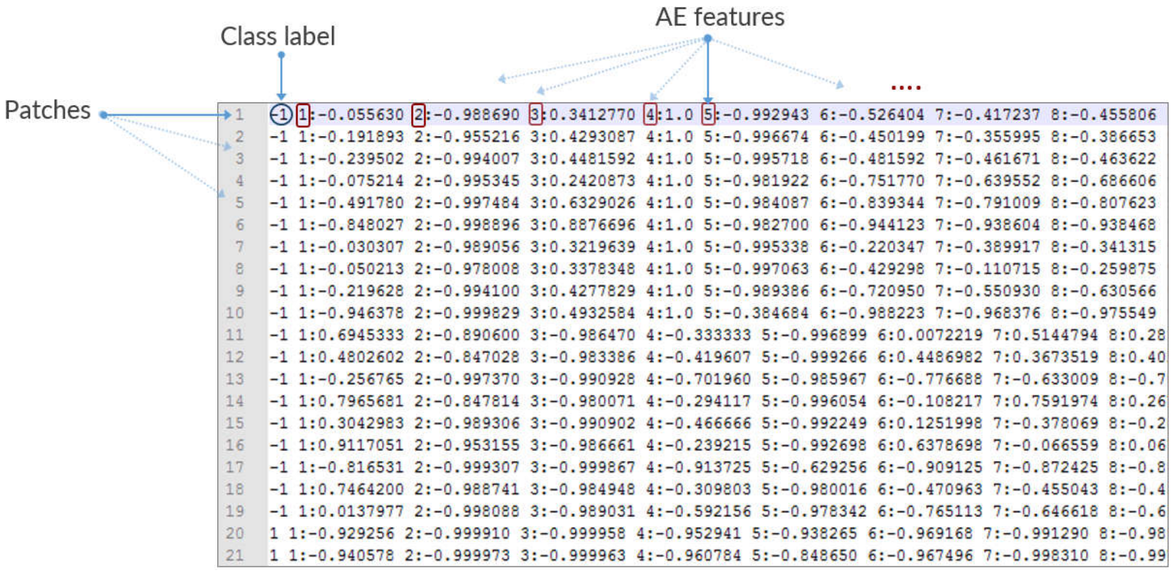


Figure 4.11: Example of SVM training file

Figure 4.11 highlights a part of an SVM training file. This file contains a total of 5,790 lines, i.e. about 2,895 line for each class. Each line represents a fixed-size vector of features extracted by an AE patch (27 features in our case).

In the prediction step, we classify patches located along the center of the candidate. Other locations such as the bottom or the top of the candidate area might contain no text despite belonging to a text area. After that, a majority voting procedure is applied to classify the candidate textline areas, as illustrated in Figure 4.12.

We use the LibSVM implementation for JAVA, introduced by [CL11], to perform our classification.

4.3 Results and discussion

4.3.1 Parameter settings

In all our experiments, the parameters of the first stage (Section 4.2.1) are empirically set as a function of data resolution and according to a statistical study on text characteristics.

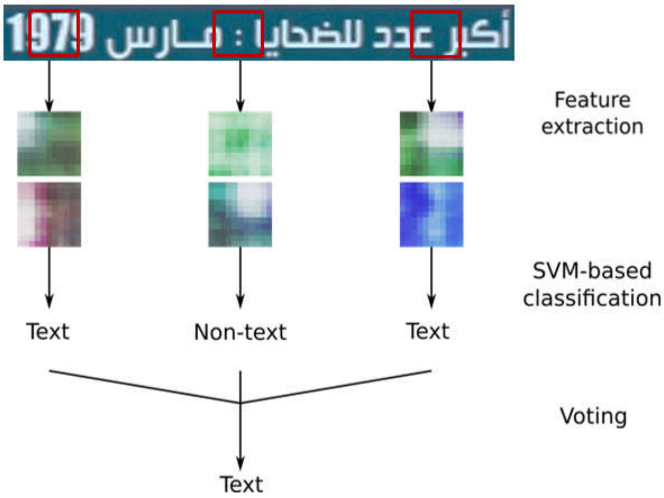


Figure 4.12: Text line classification based on majority voting

Table 4.1: Optimal CAE topology for HD/SD channels

	HD		SD	
	Layer 1	**Layer 2**	**Layer 1**	**Layer 2**
Input patch size	10 x 10	20 x 20	5 x 5	11 x 11
Convolution	3 x 3	-	3 x 3	-
Offsets	5 x 5	-	3 x 3	-
# of features	12	15	12	15

In the component extraction module, the maximal ray length $T_r = 60$ pixels. In the coarse filtering module, the maximal character/sub-word height $h_{max} = 40$ pixels, the maximal character/sub-word width $w_{max} = 120$ pixels and the maximal aspect ratio $r_{max} = 5$. In the vertical merging module, the maximal vertical distance $T_{vd} = 3$ pixels. Note that these values concern only the SD channels. For HD resolution, they should be doubled. The score thresholds of the textline construction procedure are set empirically to the following values: T_{ov}=0.75, T_{ds}=0.35, T_{al}=0.35, T_{sw}=0.24 and T_m=0.5.

A fundamental part in our experiments consists in optimizing the settings of CAE, particularly its topology. We begin with a single-layer CAE and a topology estimated as a good starting point, i.e. an input patch of a size slightly larger than the text strokes and enough neurons for having a relatively good looking reconstruction. Next, we try to optimize the topology by iteratively improving the number of features and the input patch size with regard to the classification accuracy. The optimal topologies that give us the best detection rate are presented in Table 4.1. It is interesting to note firstly that the dimensions of the first layer input patch for the HD channel are twice larger than for the SD channels, and secondly that the optimal number of features does not change. The first is due to the difference of resolution (roughly twice higher for the HD channel), and the second is explained by the fact that despite the variability in resolution the content of the inputs is similar, hence requiring a similar number of features.

Table 4.2: Number of training samples used by CAE

Protocol	TV channel	# of training data
1	AljazeeraHD	1978
4.1	France24	2153
4.2	RussiaToday	2768
4.3	TunisiaNat	3924
4.4	All SD	9623

We have noted that the second CAE layer is more efficient when it receives useful data as an input. Therefore, when we start to create a two-layer CAE, we use for the first layer the previously obtained settings, which were optimal for classification, and we optimize only the second layer.

We train CAE on the obtained textline candidates without supervision by utilizing patches randomly placed on them. Thus, their features are learned to describe the kind of content that the AEs will have to deal with during the classification phase. Table 4.2 presents the amount of the used textline candidates per protocol for training CAEs. Figure 4.13 shows some samples of them.

Figure 4.13: Example of CAE training samples

4.3.2 Experimental results

We start our experiments on the AcTiV-D dataset by firstly testing the Epshtein's SWT-based system [EOW10], as one of the most cited and used CC-based method for text detection in the recent years. Next, we propose our fully heuristic approach [ZHT+15, ZTH+16], called here "SysA", which is mainly based on the steps presented in Section 4.2.1, in addition to a refinement step. Actually, the latter utilizes projection profiles, aspect ratio and contrast information to filter out non-text lines, since text appears in horizontal arrangement and has a high contrast compared to its background. Figure 4.14 presents the input, intermediate and final images obtained with this approach. Given a video frame, color-to-grayscale transformation is first performed (Figure 4.14(b)). SWT is then applied on the edge map (Figure 4.14(c)). After that, CCs are extracted, filtered and merged to form textline candidates (Figures 4.14(d-h)). The true textlines are finally identified in the refinement step (Figure 4.14(i)).

We obtain results roughly 45% higher than the Epshtein's method. Nevertheless, the detection error rate could still be much decreased by the suggested hybrid approach (called

Figure 4.14: Detection process of fully heuristic-based system. (a) Input frame, (b) Gray-scale (c) Canny edge detection (d) SWT Map, (e) CC extraction (f) CCs after geometrical filtering (g) Diacritic merging (h) Textline construction (i) Refinement step and output result.
For clarity, only the results of one pass (DL) are presented here.

here "LADI"), which replaces the refinement step by a machine-learning solution, as presented in Section 4.2.2. The results are given in Table 4.3 in terms of Recall, Precision and F-measure metrics. For protocol P1, LADI increases the F-measure by roughly 10% in contrast to SysA. For protocols P4.1, P4.2, P4.3 and P4.4 (SD channels), the results are higher, with gains of 11%, 17%, 15% and 9%, respectively. The best accuracies of this approach are achieved on TunisiaNat1 subset (P4.3) with 0.84% as an F-measure for the SD resolution, and on Aljazeera subset (P4.1) with 0.85% as an F-measure for the HD resolution.

Comparison with other methods

In order to validate the performance of our proposed approach, we compare it with two recent methods. The first one was proposed by Gaddour *et al.* [GKV16] to detect Arabic text in natural scene images. The main process involved is:
- Pixel-color clustering using k-means to form pairs of thresholds for each RGB channel.

Table 4.3: Evaluation results and comparison with other methods.

Protocol	Method	Recall	Precision	F-measure
1	Epstein [EOW10]	0.53	0.36	0.45
	Gaddour [GKV16]	0.55	0.46	0.50
	Iwata [SWTF16]	-	-	-
	SysA [ZHT+15]	0.76	0.77	0.76
	LADI [ZST+16]	**0.84**	**0.86**	**0.85**
4.1	Epstein [EOW10]	0.5	0.3	0.4
	Gaddour [GKV16]	0.61	0.5	0.55
	Iwata [SWTF16]	0.52	0.59	0.56
	SysA [ZHT+15]	0.6	0.69	0.64
	LADI [ZST+16]	**0.75**	**0.75**	**0.75**
4.2	Epstein [EOW10]	0.42	0.36	0.39
	Gaddour [GKV16]	0.44	0.37	0.41
	Iwata [SWTF16]	**0.8**	0.77	**0.78**
	SysA [ZHT+15]	0.55	0.66	0.6
	LADI [ZST+16]	0.75	**0.79**	0.77
4.3	Epstein [EOW10]	0.47	0.35	0.41
	Gaddour [GKV16]	0.57	0.51	0.54
	Iwata [SWTF16]	0.8	**0.84**	0.82
	SysA [ZHT+15]	0.71	0.68	0.69
	LADI [ZST+16]	0.85	0.83	**0.84**
4.4	Epstein [EOW10]	0.5	0.39	0.44
	Gaddour [GKV16]	-	-	-
	Iwata [SWTF16]	0.67	**0.71**	0.69
	SysA [ZHT+15]	0.61	0.62	0.61
	LADI [ZST+16]	**0.72**	0.68	**0.7**

- Creation of a binary map for each pair of thresholds and extraction of CCs.
- Preliminary filtering according to the “area stability” criterion.
- Second filtering based on a set of statistical and geometrical rules.
- Horizontal merging of the remaining components to form textlines.

The second method was suggested by Iwata *et al.* [SWTF16] to detect Arabic text in news videos. It operates as follows:
- Binarization of input image by the Otsu method.
- Extraction of CC from the binary image using a region labeling algorithm.
- Elimination of components with a width and a height greater than predefined thresholds.
- Textline detection by vertical profile analysis and 1D difference of the Gaussian filter.
- False textline reduction by measuring the average of eccentricity ($e = perimeter^2/area$) for all CCs in the textline and removing the line if e is less than a predefined threshold.

Table 4.3 presents all systems' results using AcTiV-D as a benchmark. The LADI system scores best for protocols P1, P4.1, P4.3 and P4.4. Iwata's system performs well for all SD

(a)

(b)

Figure 4.15: Detection results from three different SD channels: Impact of the machine-learning module. (a) Results before classification. (b) Results after classification.

protocols and scores best for protocol P4.2. However, its current version is incompatible with the HD resolution. Gaddour's system has a fragmentation and miss detection tendency, as shown by their obtained results, specifically for the precision values.

Error analysis

A visual representation of some obtained results by our proposed approach are shown in Figure 4.15. Top sub-figures depict the outputs of the first stage (before classification), while the bottom ones present the final results. As it can be observed from these figures, most of the non-text regions are eliminated after the classification phase. In general, the obtained results are satisfactory and the proposed machine-learning solution seems to cope with the variability of text regions in scale, font and color. Yet, it may fail in certain conditions, to wit: (1) The edges of background objects may emit strokes to nearby text causing texts cluttered with background, as presented in Figure 4.16(a). (2) Our approach is found sensitive to structured zones; i.e., some non-text regions like balcony handrails (Figure 4.16(b)), fences (Figure4.16 (c)) or foliage (Figure 4.16(d)) are misclassified as text. Besides, in some cases, our method does not detect text affected by low contrast or low resolution (Figure 4.16 (e)). The causes of these errors can include the sensitivity of SWT to blurry images for its dependency on successful edge detection. (3) Another discovered weakness is that in some cases, true candidates are filtered by the classifier as false alarms. This can be explained by the fact that the texture (e.g., font and color) of such text is rare in the training data. Hence, some errors can be corrected by providing to the CAE module more training samples that encompass

Figure 4.16: Typical detection errors. Sub-figures (a)-(c): False alarms. Sub-figures (d)-(f): Miss detection and related problems. Sub-figures (g)-(h): Merging and fragmentation problems.

various patterns of text candidates.

4.4 Conclusion

In this chapter, we have described our approach for text detection in news video frames. The approach is based on three main steps. First, the SWT operator is applied to detect initial text components, which are then filtered and merged to form textline candidates following human-defined constraints. Finally the true textlines are pruned with a set of heuristic rules including aspect-ratio analysis and projection profiles. Lately, the third part of this approach is replaced with a machine-learning scheme based on CAE as an unsupervised feature extractor and on SVM as a classifier [ZST+16]. The experiments in this chapter demonstrate:

- The significant increase in the F-measure by 9% to 17% thanks to the use of machine-learning to filter the results given by the SWT.
- The effectiveness of AE-generated features in text/non-text classification.
- The ability of the hybrid approach to achieve a higher detection rate compared to the CC-based heuristic approach.
- The high dependence of the results of CAE-SVM classification on the performance of the SWT procedure.

As a future work, the usage of temporal information will be considered to better remove false alarms in individual frames since they are usually unstable throughout time.

Chapter 5

Text Recognition by MDLSTM Networks

5.1 Introduction

Recognition of Arabic text has become a subject of intensive research during the last decades. Particularly, several techniques have been proposed in the conventional field of Arabic OCR in scanned documents either for printed or handwriting text [AB96, LG06, TAA07, SIAH08, EAKMA11, MEA12]. However, little work has been made regarding the development of recognition systems for overlaid text in Arabic news videos [HAV+12, YBG15b].

As it was mentioned in the Introduction (Chapter 1), Arabic text has special characteristics. For instance, several Arabic letters share common primary shapes, differing only in the number of dots and whether the dots are above or below the main character, like *Baa* (ب) and *Taaa* (ت) characters. Therefore, any binarization or morphological operation needs to efficiently deal with these dots so as not to change the identity of the character. Arabic has several ligatures, which are formed by combining two or more letters, such as the two first letters *Miim* (م) and *Haaa* (ح) of the word *Mohamed* (محمد), making it difficult to segment the words or PAWs into individual characters for subsequent recognition. All these characteristics along with the complexity of video content may give rise to failures in the Arabic video text recognition task.

In this chapter, we propose a novel text recognition system based on a segmentation-free methodology, which relies on the use of Long short-term memory (LSTM) networks. These networks have been successfully applied in different sequence classification problems and have outperformed alternative HMMs [TAA07, SZK+12], RNNs [SP97, SR98] and their combination. Some benchmark work has been developed using the RNN-LSTM networks, such as handwriting recognition of Latin and Asian scripts [GLF+09, ML15]. The good performance of such networks has motivated us to investigate their application for the recognition of Arabic text in video frames. The multidimensional LTSM (MDLSTM) architecture [Gra12] is particularly adopted to model the text variations on both axes of the input image. Up to our knowledge, we have been the first to use this architecture for such a problem. Besides, we suggest an efficient preprocessing step and a compact representation of character models, which permits improving the behavior of our system and increasing the recognition rates.

The rest of the chapter is organized as follows. Section 5.2 presents a short overview of the RNN-LSTM networks. The proposed system is presented in Section 5.3. Section 5.4 describes the grouping strategy of character models. The experimental setup and obtained results are presented in Section 5.5. Section 5.6 draws conclusions.

5.2 Overview of RNN-based networks

RNNs were first introduced in the 80s and have become popular due to their ability to model contextual information. They represent powerful tools for processing patterns occurring in time series. In its simplest form, an RNN is an MLP with recurrent layers which receive inputs not only from the previous layers, but also from themselves, as shown in Figure 5.1(a).

Consider an input sequence x presented to an RNN with I input units, H hidden units and O output units. Then the hidden units a_h and the activations b_h of a recurrent layer are

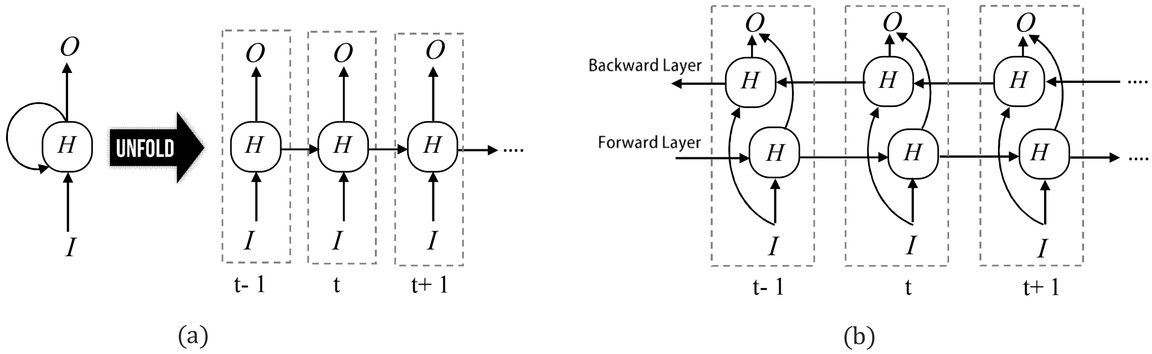


Figure 5.1: Standard and Bidirectional Recurrent Neural Networks. (a) RNN, (b) BRNN.

calculated as follows:

$$
\begin{aligned}
a_h(t) &= \sum_{i=1}^{I} w_{ih} x_i(t) + \sum_{h'=1}^{H} w_{h'h} b_{h'}(t-1) \\
b_h(t) &= \Theta_h(a_h(t))
\end{aligned}
\tag{5.1}
$$

where $x_i(t)$ is the value of input i at time t, $a_j(t)$ and $b_j(t)$ respectively denote the network input to unit j and the activation of unit j at time t, w_{ij} denotes the connection from unit i to unit j, and Θ_h is the activation function of hidden unit h.

Robinson [Rob94] was among the first who suggested the use of standard RNNs for speech recognition. Lee and Kim [LK95] and Senior and Robinson [SR98] applied such networks to handwriting recognition.

In 1997, Schuter and Paliwal [SP97] introduced Bidirectional RNNs (BRNNs) by implementing two recurrent layers, one processing the sequence in a forward direction (left to right) and the other backwards. Both layers are connected to the same input and output layers (see Figure 5.1(b)).

The MDRNN architecture [GFS07] represents a generalization of RNNs, which can deal with multidimensional data, such as images (2D) and videos (3D). In order to extend the RNN to a multidimensional one, let $p \in \mathbb{Z}^D$ be a point in an n-dimensional input sequence x of dimensions $D_1, ..., D_n$. Instead of $a(t)$ in a 1-dimensional case, we write a^p as an input in the multidimensional case. The upper index p_i, $i \in \{1, 2, 3, ..., n\}$, is used to define the position; i.e., $P_d^- = (p1, ..., p_d - 1, ..., p_n)$ denotes the position on a step back in dimension d. Let w_{ij}^d be the recurrent connection from i to j along dimension d. The forward equation for an n-dimensional MDRNN is calculated according to Equation (5.2).

$$
\begin{aligned}
a_h^p &= \sum_{i=1}^{I} w_{ih} x_i^P + \sum_{d=1}^{n} \sum_{h'=1}^{H} b_h^{P_d^-} w_{h'h}^d \\
b_h^p &= \Theta_h(a_h^p)
\end{aligned}
\tag{5.2}
$$

The backward pass is given by Equation (5.3), where $\epsilon_j^p = \frac{\partial E}{\partial b_j^p}$ and $\delta_j^p = \frac{\partial E}{\partial a_j^p}$ respectively denote the output error of unit j at time p and the error after accumulation.

$$\begin{aligned} \epsilon_h^p &= \sum_{o=1}^{O} \delta_o^p w_{ho} + \sum_{d=1}^{n} \sum_{h'=1}^{H} \delta_h^{p_d^+} w_{h'h}^d \\ \delta_h^p &= \theta_h'(a_h^p)\epsilon_h^p \end{aligned} \tag{5.3}$$

Figure 5.2 illustrates the two-dimensional case of an MDRNN. During the forward pass, at

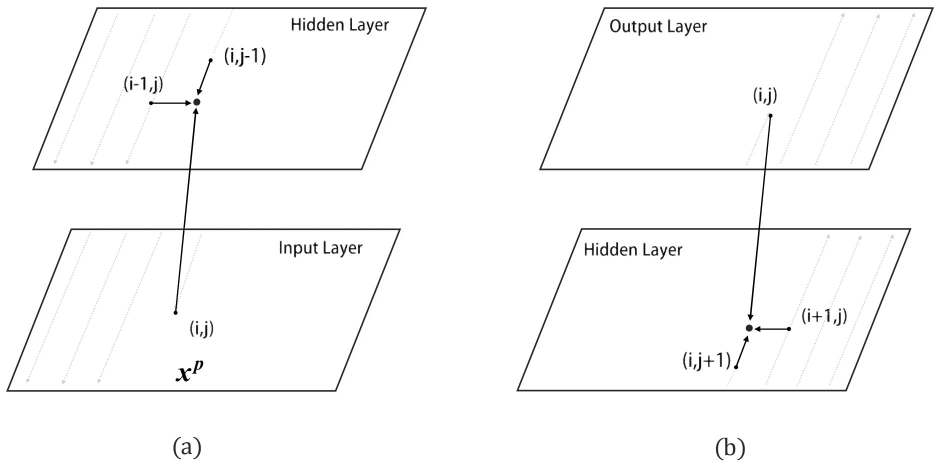


Figure 5.2: Illustration of two-dimensional MDRNN. (a) Forward pass (b) Backward pass, inspired from [G+12].

each point in the 2D input sequence X^p, the hidden layer of the network receives both an external input and its own activations from one step back along all dimensions (Figure 5.2(a)). It is to note that point (i, j) is never reached before both $(i-1, j)$ and $(i, j-1)$.
In the backward pass, the backpropagation through time (BPTT) [Wer90] is generally used to compute the *error gradient* of the network. Indeed, the sequence is processed in the reverse order of the forward pass; i.e., at each timestep the hidden layer receives both the output error derivatives and its own n 'future' derivatives, (Figure 5.2(b)) [G+12].

While standard RNNs use a recurrence only over one dimension, like the x-axis of an image, MDRNNs scan the input image along both axes, allowing the exploitation of more context and the modeling of the text variations in four directions (left, right, top, bottom). Figure 5.3(a) shows the axes used in the MDRNN scan. The arrows inside the rectangle indicate the direction of propagation during the forward pass. The hidden layers are connected to a single output layer which has access to all the surrounding context. One such layer is sufficient to give the network access to all context against the direction of scanning from the current point (e.g. to the top and left of (i, j) in Figure 5.2(a)). However we usually want surrounding context in all directions, as depicted in Figure 5.3(b).

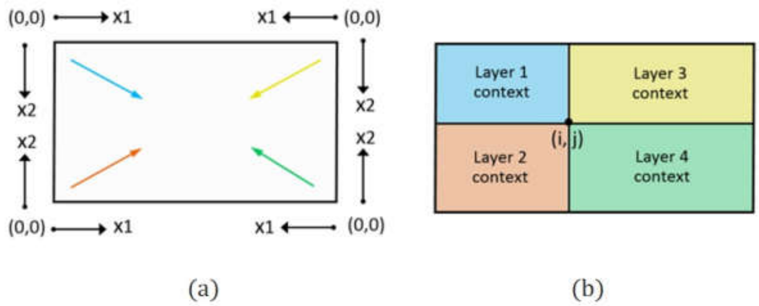


Figure 5.3: Scanning directions of MDRNN, inspired from [G+12]. (a) Axes used by four hidden layers in 2D MDRNN. (b) Context available at current point (i,j).

5.2.1 LSTM networks

The problems of long-term dependencies and *vanishing gradient* —the gradient of the loss function decays exponentially over time [BSF94] —have been the reason for the lack of practical applications of RNNs. In 1997 [HS97], an advance in designing such networks was introduced as the LSTM models. Indeed, they are a special kind of RNNs that use memory cells as hidden layer units. These cells can maintain information for long periods of time.

LSTM consists of a set of three multiplicative *gates*, so-called the input gate i, the output

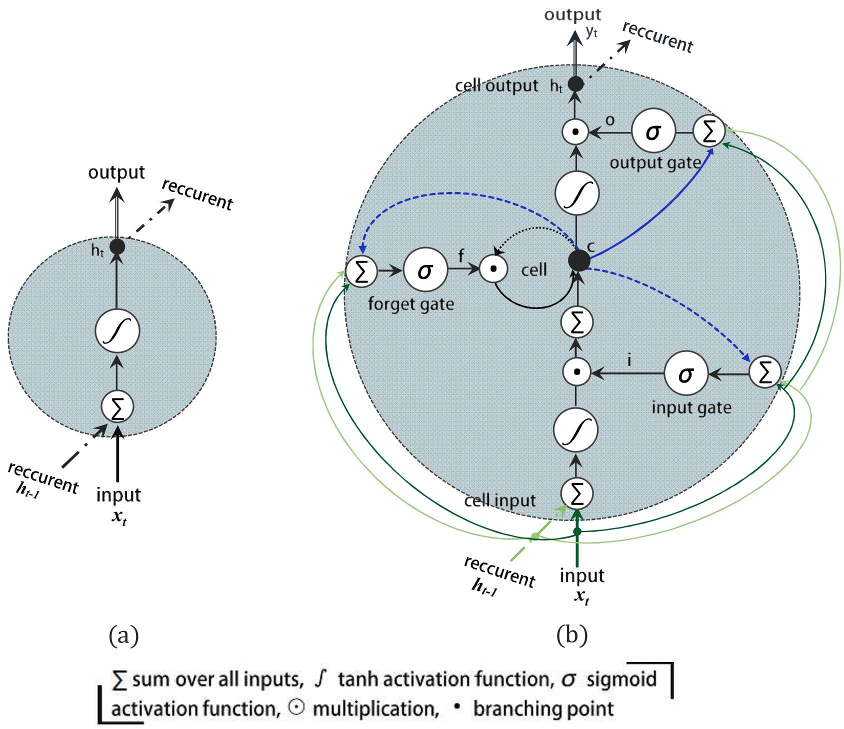


Figure 5.4: Detailed schematic of neurons for RNNs: (a) Simple neuron (b) LSTM unit.

gate o and the forget gate f, to control when information should be stored or removed from the memory cell c. This architecture lets them learn longer-term dependencies. See Figure 5.4(b) for an illustration. LSTM first computes its gates' activation i_t (Equation 5.4), f_t (Equation 5.5) and updates its cell state from c_{t-1} to c_t (Equation 5.6). It then computes the output gate activation o_t (Equation 5.7), and finally outputs a hidden representation h_t (Equation 5.8). The inputs of an LSTM unit are the observations $\mathrm{x_t}$ and the hidden representation from the previous time step $\mathrm{h_{t-1}}$. LSTM runs the following set of update operations:

$$i_t = \sigma(W_{ix}x_t + W_{ih}h_{t-1} + W_{ic}c_{t-1} + b_i) \tag{5.4}$$

$$f_t = \sigma(W_{fx}x_t + W_{fh}h_{t-1} + W_{fc}c_{t-1} + b_c) \tag{5.5}$$

$$c_t = f_t c_{t-1} + i_t \tanh(W_{xc}x_t + W_{hc}h_{t-1} + b_c) \tag{5.6}$$

$$o_t = \sigma(W_{ox}x_t + W_{oh}h_{t-1} + W_{oc}c_t + b_o) \tag{5.7}$$

$$h_t = o_t \tanh(c_t) \tag{5.8}$$

$$y_t = W_{yh}h_t + b_y \tag{5.9}$$

where W denotes weight matrices, b denotes bias vectors and σ (Equation 5.10) is the logistic sigmoid function.

$$\sigma(x) = \frac{1}{1 + \exp(-x)} \tag{5.10}$$

Standard LSTM is explicitly 1D, since each cell includes one single recurrent connection, whose activation is controlled by a single forget gate. Nevertheless, it is possible to extend this to n dimensions, i.e. an MDLSTM memory cell, by using n recurrent connections with n forget gates (one for each of the cell's previous states along every dimension).

5.2.2 Connectionist Temporal Classification layer

One basic problem with RNNs is that they require a target output at each timestep. Thus, training an RNN requires segmenting the training output, i.e. 'telling' the network which label should be output at each timestep. To overcome such a requirement, Graves *et al.* [GFGS06] proposed the Connectionist Temporal Classification (CTC) for sequence labeling task. This technique is inspired from the HMM forward-backward search algorithm [ASP91] and is used to align the target labels with the LSTM output sequences. During training, this alignment enables the network to learn the relative location of labels in the whole transcription. The CTC layer contains as many units as there are elements in the alphabet L of labels, plus one extra 'blank' unit $\oslash$; i.e., the output alphabet is $L' = L \cup \{\oslash\}$. 'Blank' is not a real character class, but a virtual symbol used to separate the consecutive real characters. Let x be an input sequence of length T and $\beta : L' \to L^{\leq T}$ a mapping function, which removes duplicates then blanks in the network prediction. For example, $\beta(a \oslash \oslash ab) = \beta(aa \oslash \oslash abb) = aab$.

Since the network outputs for different timesteps are conditionally independent given x, the probability of a label sequence $\pi \in L'^T$ in terms of LSTM outputs is as follows:

$$p(\pi|x) = \prod_{t=1}^{T} y^t_{\pi t}(x) \tag{5.11}$$

where y^t_k is the activation of output unit k at time t. Mapping β allows calculating the posterior probability of a character sequence $l \in L^T$, which is the sum of the probabilities of all *paths* (L'^T) corresponding to it:

$$p(l|x) = \sum_{\pi \in \beta^{-1}(l)} p(\pi|x) \tag{5.12}$$

This 'collapsing together' of various paths to the same labeling is what enables CTC to use unsegmented data. After that, the CTC objective function maximizes the probability to find the most probable label sequence for the corresponding unsegmented training data $S = \{(x, z), z \in L^{|x|}\}$ by minimizing the following cost:

$$\vartheta = - \sum_{(x,z) \in S} log\ p(z|x) \tag{5.13}$$

5.3 Proposed system

The proposed video text recognition system is based on RNNs. It relies specifically on an MDLSTM network coupled with a CTC output layer. It is mainly developed using an adapted version of the open-source RNNLib toolkit. The use of RNNLib goes typically through two phases: training and test. During the training step, the network learns the sequence-to-sequence matching in a supervised manner, i.e. the alignment between the input and the output sequences. In the test step, the normalized textline image is fed to the trained MDL-STM model, which generates the predicted sequence. For both steps we apply the same preprocessing, as illustrated in Figure 5.5.

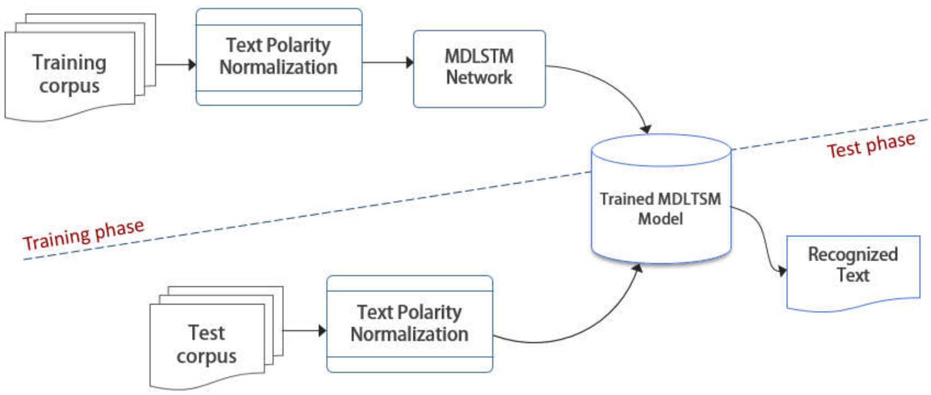


Figure 5.5: Complete pipeline of our LSTM-based recognition system.

In what follows, we describe the preprocessing stage.

5.3.1 Preprocessing

Blatantly, Video OCR domain has many problems to deal with concerning the variability of text patterns, the complexity of backgrounds, etc. Therefore, we propose to apply some preprocessing prior to the recognition step in order to reduce these undesirable effects. Given

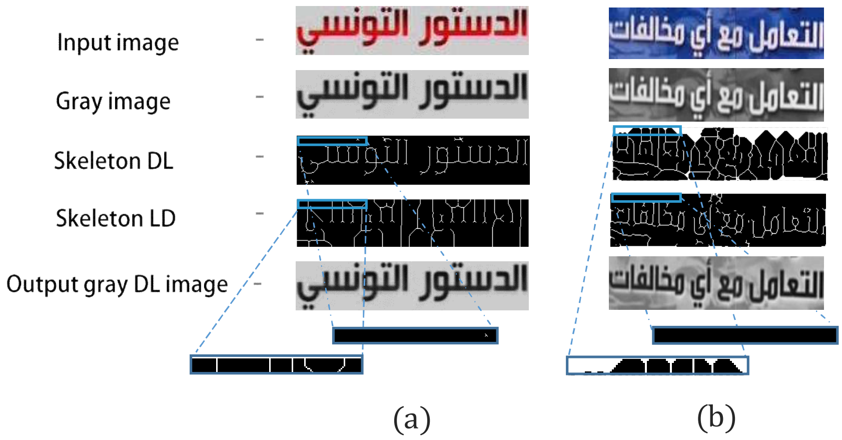


Figure 5.6: Pre-processing step of text gradient normalization

a textline image, the preprocessing operations of text polarity normalization and image size scaling are performed. First, the text polarity is determined, i.e. judging whether it is dark text on light background or vice-versa, using a skeleton-based technique. Skeletons are important shape descriptors in object representation and recognition. The generalized skeleton representation of a binary image is the union of sets $\{S_n\}$ given by Equation 5.14.

$$S_n(X) = (X \ominus nB) - (X \ominus nB) \circ B \tag{5.14}$$

where $S_n(X)$ represents the *skeleton subsets* of a binary image containing a set of topologically open shapes X, n is the number of shapes, and B is a structuring element. The symbols $\ominus$ and $\circ$ refer to the binary erosion and opening, respectively. Note that in our case the binary images Bin and $\overline{Bin}$ are obtained by adaptive thresholding the input grayscale image Gs and its negative version $\overline{Gs}$ (step (2) of Algorithm 1). It can be observed from the content distribution of the skeleton maps (step (3) and (4) of Algorithm 1) created with the correct gradient direction, that the skeleton pixels are retained in the center line of the character shape (e.g. skeleton dark-on-light (DL) in Figure 5.6(a) and skeleton light-on-dark (LD) in Figure 5.6(b)).

This is due to the characteristics of the skeleton function that generates a thin version of the original shape, which is equidistant to its boundaries. Otherwise, the skeleton pixels all surround the characters and are placed on the image boundaries (cf. skeleton LD in Figure 5.6(a) and skeleton DL in Figure 5.6(b)). Thus, the text gradient direction is simply obtained by comparing the quantity of white pixels (WPs) located on the boundaries of the two skeleton images (step (11) of Algorithm 1); i.e., we invert the input grayscale image if its skeleton LD has fewer WPs on the boundaries (step (12) of Algorithm 1). Subsequently, the

Algorithm 1: Text polarity normalization to dark-on-light

Input : original text image In
Output: normalized image
1 Gs $\leftarrow$ rgb-To-grayscale(In)
2 Bin$\leftarrow$ Binarization (Gs)
3 $\mathrm{S_i}$ $\leftarrow$ SkeletonExtract (Bin)
4 $\overline{S_i}$ $\leftarrow$ SkeletonExtract ($\overline{Bin}$)
5 **for** ***all*** *pixel* $\boldsymbol{I}(x, y)$ *in image* S_i **do**
6   **if** *(*$\boldsymbol{I}(x, y) \in$ *border of* S_i *& is* > 0*)* **then**
7   increase WP by 1;
8   **end**
9 **end**
10 Repeat steps (5 - 9) for image $\bar{S}_i$ to compute $\mathrm{WP}_{\bar{S}}$
11 **if** $WP_{\bar{S}} > WP_S$ **then**
12   Text polarity inversion to DL;
13 **else**
14   No inversion of text polarity;
15 **end**

text polarity is normalized to DL for all input grayscale images, as shown at the bottom of Figure 5.6. This method has been able to achieve an accuracy of 95% on our dataset.
All the normalized images are then scaled to a common height (determined empirically) using the bi-linear interpolation method.

5.3.2 Network architecture

As depicted in Figure 5.7, our network consists of five layers in which three are LSTM-based hidden layers (for each direction) and two are feedforward subsampling layers with *tanh* as an activation function. We adopt the hierarchical network topology as used in [G+12] by repeatedly composing MDLSTM layers with feedforward *tanh* layers. The principle of such a topology is detailed in Figure 5.8. The purpose of the subsampling step is to compress the sequence into windows, thus speeding up the training time with the MDLSTM architecture. Subsampling is also required for reducing the number of weight connections between hidden layers.

In this network, there are mainly four important parameters that require tuning during the training phase.

- The *Input Block size* refers to the "width x height" of the pixel block used to initially divide the input text image into small patches. For our proposed models we empirically set the size of this parameter as 2 x 4 or 1 x 4 depending upon the evaluation protocol (see Section 5.5.1).

- The *LSTM Size* refers to the number of LSTM cells in each hidden layer. In our work, 2, 10 and 50 represent the appropriate values for this parameter. These values are found empirically and they match as well those reported by other researchers [G+12, PBKL14,

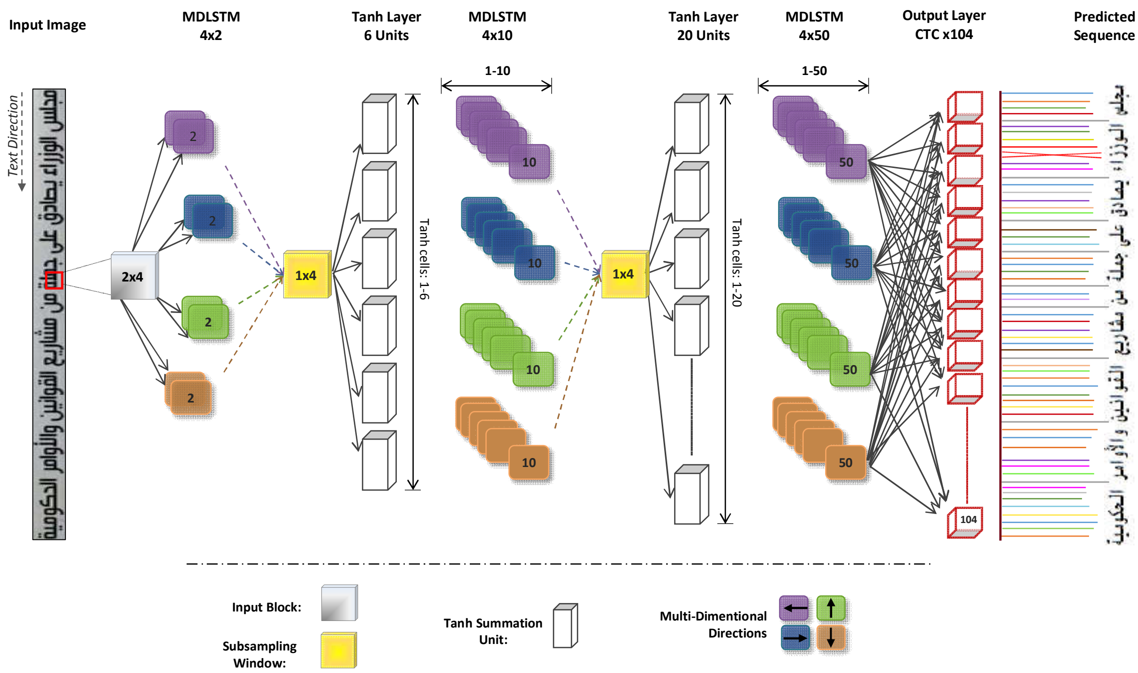


Figure 5.7: Architecture of the used hierarchical subsampling MDSLTM network

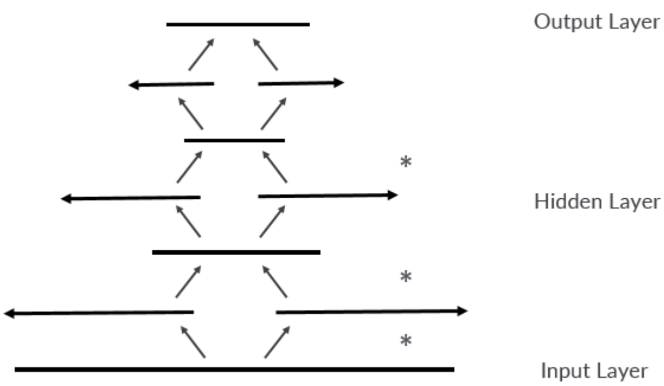


Figure 5.8: Data flow through a multidimensional HSRNN [G+12]. The input sequence is subsampled and then scanned by recurrent hidden layers. The sequence of hidden layer activations is subsampled again and scanned by the next hidden layers. The activations of the last hidden layer are fed to the output layer without subsampling. Subsampling is performed at the places indicated with a '*'.

NUA+17]. Note that the number of LSTM cells, for each hidden layer, should be equal to the size of that layer multiplied by the number of directions in which the input image is scanned. In the proposed architecture, the image is scanned in four different directions. Hence, the number of LSTM cells become 2 x 4, 10 x 4 and 50 x 4. This is shown in Figure 5.7 by four different colors of LSTM cells.

- The *Tanh Size* describes the number of *tanh* units in each subsampling layer. The suitable values of this parameter are set to 6 and 20, respectively, for the first and second feedforward *tanh* layers that are placed between each pair of LSTM layers.

- The last parameter is the *Subsampling Window size*. It refers to the window required for subsampling the input from each layer before feeding it to the next hidden layer. This parameter decreases the sequence length, in the applied layer, by a factor corresponding to the window width. The optimal sizes are set to 1 x 4 for both 1st and 2nd hidden layers. At the hidden-to-output layer transition, no subsampling is applied.

The output of the last LSTM hidden layer is passed to a CTC output layer, which transcribes the input sequence by choosing the sequence of labels with the highest conditional probability, as explained above in Section 5.2.2. This layer has 105 units: 104 basic class labels plus one for the 'blank'.

The training is carried out with the BPTT algorithm, and the steepset optimizer is used with a learning rate of 10^{-4} and a momentum value of 0.9. Training stops when the validation error shows no improvement in successive 20 epochs.

5.4 Choice of model sets

By a model set, we mean the number of classes used to represent the different variations in character shapes. Benefiting from the morphological characteristics of the Arabic alphabet, we propose a glyph-based grouping method, resulting in three sets with respectively 165, 104 and 72 classes, as described in the following. This proposal has a direct impact on the size of the CTC output layer, and consequently on the behavior of the network.

- **Set165:** As stated in the Introduction, the Arabic alphabet contains 28 characters and most of them change shape according to their position in the word. Taking into account this variability, the number of shapes increases from 28 up to 100. In addition, the Arabic script includes two groups of extra characters. The first one represents a variation in some basic characters like the *TaaaClosed* (ة), which is a special form of the character *Taaa* (ت), and the *HamzaAboveWaaw* (ؤ), a combination of *Hamza* (ء) + *Waaw* (و). The second group includes four ligatures created when the character *Alif* (or one of its variants) follows the character *Laam* (ل) in the word. Considering these extra characters, there are overall 125 shapes. Added to that, 10 digits, 13 punctuation marks and 12 additional characters that are combined with the diacritic mark *Chadda* (ّ), so the total number of models in our database goes up to 165.

- **Set104:** Using *set165*, we group similar glyphs into 104 models according to the following rules: (1) "Beginning" and "middle" shapes share the same model. (2) "End" and "isolated" shapes share the same model. These rules are applied for all alphabet characters except for the characters *Ayn* (ع) and *Ghayn* (غ) where the initial, middle, final and isolated shapes are too different. This strategy of grouping is natural as "beginning-middle" and "end-isolated" character shapes are visually similar. For instance, the two first character models (left-to-right) of the word in Figure 5.9 are grouped to one model as they belong to the same basic character *Taaa* (ت), so we obtain two samples of the model *Taaa_B* instead of having one for the model *Taaa_B* and another for the model *Taaa_M*.

- **Set72:** We use here one single model for each character of *set165*, regardless its position in the word.

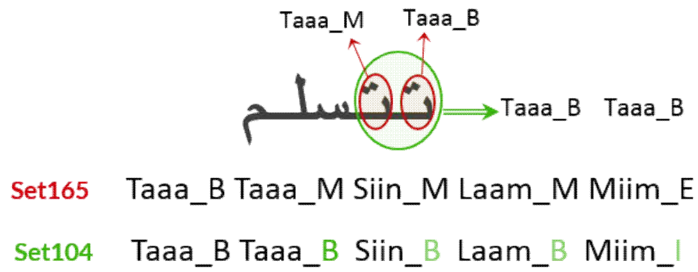


Figure 5.9: Sequence of models with proposed sets 165 and 104. 'B', 'M', 'E' and 'I' respectively denote the letter positions Begin, Middle, End and Isolate.

The question to address regarding these sets is: "Does a trade-off exist between having more models per character (to capture the intrinsic details of each glyph i.e., *set165*) and having more training samples per character model (without considering the details of character shapes i.e., *set104* and *set72*)?"

Table 5.1: Impact of MDLSTM size against a fixed size of feedforward layer

MDLSTM size	CRR (%)	LRR (%)	Total epochs	Time per epoch (minutes)
2, 10, 50	**96.26**	**68.26**	**128**	**7**
4, 20, 100	95.65	66.14	350	19
20, 60, 100	97.04	74.61	162	46
8, 30, 150	97.12	75.67	298	63

5.5 Experimental results

This section describes the set of experiments that we conducted separately to (i) fix the optimal network parameters, (ii) analyze the effect of both preprocessing and model sets on the recognition performance, and (iii) compare the proposed system with other recently published methods.

5.5.1 Selection of optimal network parameters

The optimal parameters for our proposed MDLSTM model are found by empirical analysis. Note that for these experiments we just pick out a small set of 2,000 text images from AcTiV-R, in which 190 are used as a validation set. We first need to find the best size of hidden MDLSTM layers, which gives us an optimal performance. To do that, we fix the size of feedforward *tanh* layers to 6 and 20. As shown in Table 5.1, the suitable values of the MDLSTM size, which give us optimal results, are 2, 10 and 50 for the 1^{st}, 2^{nd} and 3^{rd} hidden layers, respectively. Afterwards, we evaluate the impact of the feedforward size against the fixed optimal size of MDLSTM layers (2, 10 and 50). As a consequence, the best obtained size of feedforward layers is 6 and 20 for the 1^{st} and 2^{nd} feedforward *tanh* layers, respectively, as represented in Table 5.2. The indicators observed during the fine-tuning of these parameters are CRR and the average time per epoch. It is interesting to note that such results are not comparable with the system results obtained in the next sections.

Once the architecture is fixed, we perform several experiments to find the best sizes of the input block and the hidden block (subsampling window). Therefore, the size of the input block is fixed to 1 x 4 for protocols P6.1 and P3, and to 2 x 4 for the remaining protocols. The hidden block sizes are fixed to 1 x 4 and 1 x 4 for all protocols.

Table 5.2: Impact of feedforward layers size against a fixed size of MDLSTM layers

Feed-forward size	CRR (%)	LRR (%)	Total epochs	Time per epoch (minutes)
6, 20	**96.26**	**68.26**	**128**	**7**
8, 30	96.46	71.43	347	13
12, 40	96.43	70.38	309	8
16, 80	97.09	75.7	204	17

Table 5.3: Results of proposed recognition system on AcTiV-R dataset: Impact of polarity normalization

	Without normalization of text polarity			With normalization of text polarity		
	CRR (%)	WRR (%)	LRR (%)	CRR (%)	WRR (%)	LRR (%)
P3	90.03	71.18	51.54	**92.2**	**74.13**	**53**
P6.1	89.1	70.49	51.4	**91.5**	**79.66**	**57**
P6.2	93.8	68.22	40.8	**93.33**	**68.9**	**43.6**
P6.3	94.3	80.77	62.44	**96.16**	**85.14**	**67.73**
P6.4	93.17	73	52.4	**94.1**	**81.23**	**57.3**
P9	73.4	58.34	31.9	**80.41**	**60.6**	**38.14**

5.5.2 Impact of preprocessing step

To examine the impact of text polarity normalization on the input grayscale images of each protocol, we carry out several experiments by training two different types of input images, with and without text polarity normalization. Note that for these experiments, we use the same network architecture and we fix the height of all images to 70 pixels. By carefully examining the obtained results given in Table 5.3, it is concluded that the preprocessing step has a clear beneficial effect on the recognition accuracy. The results indicate that by using both height and polarity normalization, the LRR increases from 51.54% to 53% for AljazeeraHD's protocol (P3), from 51.40% to 57% for France24's protocol (P6.1), from 40.82% to 43.6% for RussiaToday's protocol (P6.2), and from 62.44% to 67.73% for TunisiaNat1's protocol (P6.3). An increase of 5.13% is achieved on the AllSD protocol (P6.4) and of 7% on the channel-free protocol (P9). The best results are marked in bold in Table 5.3.

Table 5.4: Final obtained results on AcTiV-R dataset: Impact of model sets choice

	Set165			Set104			Set72		
	CRR (%)	WRR (%)	LRR (%)	CRR (%)	WRR (%)	LRR (%)	CRR (%)	WRR (%)	LRR (%)
P3	92.2	74.93	53,8	**94.62**	**83.11**	**64.29**	92.71	75.29	54
P6.1	91.5	79.66	57	**92,27**	**81.19**	**59.55**	91	75.18	52
P6.2	93.33	68.9	43.6	**94.1**	**73.67**	**49.27**	90.45	67.24	43.2
P6.3	96.16	85.14	67.73	**96.48**	**86.05**	**72.49**	93.87	82.39	63.6
P6.4	94.1	81.23	57.3	**95.82**	**83.4**	**63.27**	92.11	78.53	55.8
P9	80.41	60.64	38.14	**88**	**70.28**	**47.32**	80.77	59.11	36.4

5.5.3 Effect of model set choice

Table 5.4 provides the recognition results of *set165*, *set104* and *set72*-based systems. We can see that the performances grow significantly (e.g. 11.29 % for P3) from *set165* to *set104*. It seems beneficial to finely model the difference between begin-middle shapes and end-isolate ones. For instance, the character *TildAboveAlif* (آ) in the end position is represented with only 32 occurrences in the dataset. Intuitively, we should lose more precision of the modeling utilizing less models. Nevertheless, we observe here the effect of having too few training data for less frequent representations of some character shapes. On the other hand, the performances decline considerably (at least 6%) from *set104* to *set72*, where a single sub-model per character is used.

Overall, our best system for all evaluation protocols is the one based on *set104*. The best accuracies are achieved on the TunisiaNat1 channel subset (P6.3) with 96.48% as a CRR and 72.49% as an LRR. An important rise of 9.4% for the channel-free protocol (P9) is achieved in terms of LRR.

5.5.4 Error analysis

Figure 5.10 depicts some typical misrecognized lines. It contains four blocks. Each block presents two (or three) input images and their corresponding output sequences. Block (a) shows two images from protocol P3. For each image we present its results with *set165* and *set104*, respectively. As it can be seen, most erroneous characters in the first set are correctly recognized (green color) using *set104*. Block (b) illustrates two output lines (per image) of

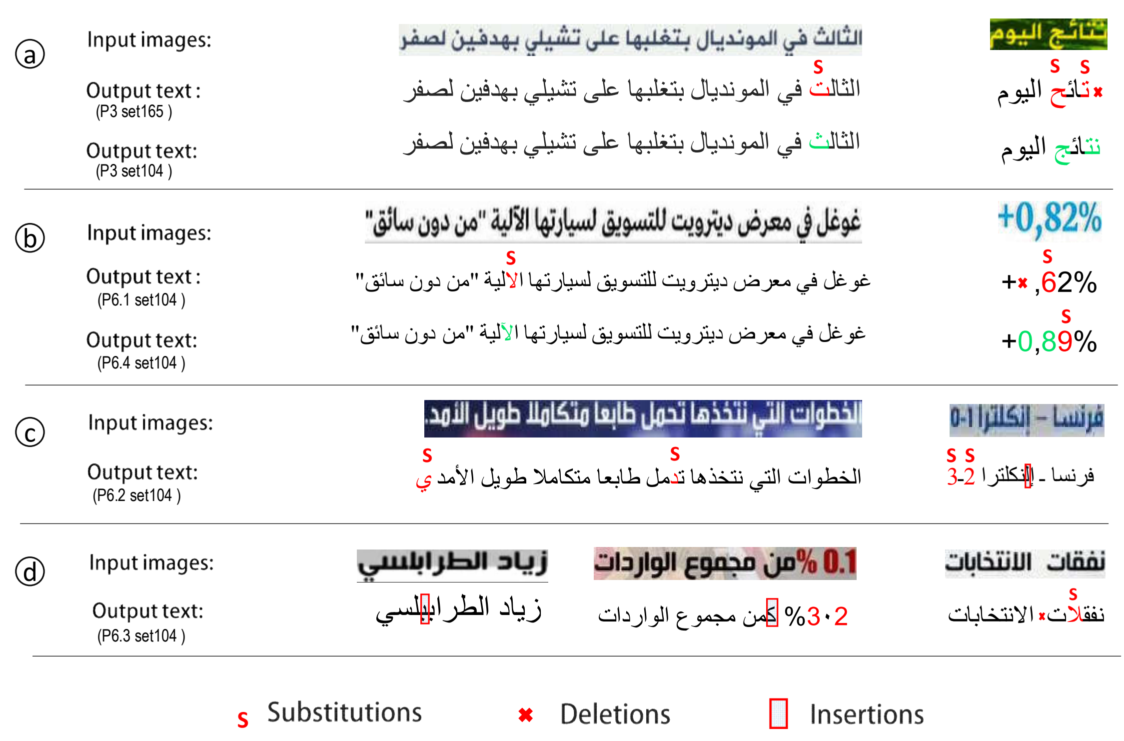


Figure 5.10: Examples of some output errors picked out from experimental results. Errors are marked by red symbols.

two different evaluation protocols, P6.1 and P6.4 (AllSD protocol). It is clear that for both images the results of P6.4 are better than those of P6.1. This can be explained by the presence of more training shapes in the AllSD protocol. Block (c) and (d) present examples of output lines from P6.2 and P6.3, respectively. A visual inspection of the errors is actually supporting the aforementioned statement, where frequent errors are related to less frequent shapes in the training database.

Based on our knowledge about the shapes of Arabic characters, we divide the cause of errors into two categories: character similarity (substitution errors of block (a)) and insufficient samples of punctuation, digits and symbols (substitution and deletion errors of blocks (b), (c) and (d)). Several measures can be taken to minimize the character error rate. For instance, some errors can be corrected by integrating language models and dropout regularization [PBKL14] to improve the LSTM-based recognition system and raise the generalization performance [FZMEB+12].

5.5.5 Comparison with other methods

We validate here the performance of our proposed system by comparing it to the method presented by Iwata *et al.* [IOWK16] (see Section 2.3 of Chapter 2). As depicted in Figure 5.11, we outperform Iwata's system by a large margin in all protocols. The obtained results, in terms of LRR, are higher with a gain ranging from 10% to 16% for protocols P6.2 and P6.3, respectively. It is to note that the current version of Iwata's system is not compatible with the HD resolution.

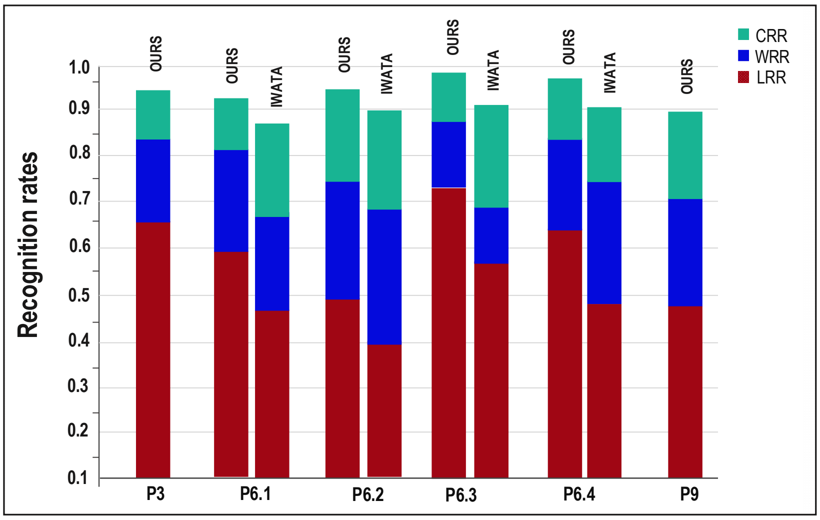


Figure 5.11: Comparison of our recognition system to Iwata's on the test-set of AcTiV-R.

We also evaluate our system on the ALIF dataset [YBG15a], which represents, to the best of our knowledge, the only benchmark for Arabic video text recognition, as stated in Section 3.2 of Chapter 3. The dataset is composed of 6,532 cropped text images extracted from diverse Arabic TV channels. Indeed, 4,152 images from the database are used for training and the remaining images constitute the test set. ALIF contains only one resolution (SD) and presents 140 character glyphs including digits and punctuation symbols. Figure 5.12 illustrates some samples from this dataset.

Figure 5.12: Examples of text images from ALIF dataset [YBG15a]

Table 5.5 shows the comparative results for the proposed text recognition method against five recently proposed systems. Note that these systems were developed by the same author who put forward the ALIF dataset [YBG15a], and four of them were BLSTM-based. For these experiments, we use the same preprocessing steps and optimal network parameters, which give us the best recognition accuracies on the AcTiV-R dataset. We also adopt the same rules of model grouping as those used for *set104* in Section 5.4. Interestingly, our proposed MDLSTM network with the normalization step outperforms the BLSTM systems whether they are based on manually crafted features (HC-BLSTM) or automatic learned features (DBN-AE-BLSTM, MLP-AE-BLSTM and CNN-BLSTM). We are able to achieve results that are roughly 16%

higher than the best rate obtained by the CNN-BLSTM system, in terms of LRR. These results are obtained on the ALIF_Test1 subset [YBG15a], which includes 900 textline images.

Table 5.5: Obtained results on ALIF dataset and comparison with others systems

	CRR (%)	**WRR (%)**	**LRR (%)**
DBN-AE-BLSTM [YBG15b]	90.73	59.45	39.39
MLP-AE-BLSTM [YBG15b]	88.50	59.95	33.19
ConvNet-BLSTM [YBG15b]	94.36	71.26	55.03
HC-BLSTM [YBG15a]	85.44	52.13	-
ABBYY [YBG15a]	83.26	49.80	26.91
Proposed system	**96.85**	**83.2**	**70.67**

5.6 Conclusion

We have presented in this chapter an Arabic video text recognition system based on MDLSTM coupled with a CTC output layer. The suggested system permits avoiding two hard OCR steps, which are textline segmentation and handcrafted feature extraction. The proposed method has been trained and evaluated using the AcTiV-R database. We have reported 96.5% as CRR and above 72.4% as LRR for the SD resolution. The preprocessing step and model sets choice have brought significant recognition improvement in terms of reduction in the line error rate. Our method has also outperformed the results of previous works on the ALIF dataset, more particularly those based on the combination of CNN and BLSTM [YBG15b, YBG15a]. The interesting findings in this study have been the application of the MDLSTM network to low-resolution Arabic text with unknown font sizes and font families, the use of an efficient normalization step and the analysis of the impact of model sets.

Chapter 6

Conclusions and future Work

In this chapter, we conclude the present thesis by summarizing our contributions, discussing the related limitations and providing some future directions.

Conclusions

In this thesis, we have tackled the problems of video text detection and recognition. The main purpose of this research work is to help end-users like archivists in the indexing and retrieval of broadcast news videos, especially when they need to deal with huge multimedia databases.

In contrast to the conventional field of printed and handwritten OCR, which has been widely addressed in the literature, text detection and recognition in videos are still an open problem. This is due to several challenges including background complexity (e.g. presence of noise, low resolution and text-like objects) and text variability in terms of size, color, position and font. The present study has focused on artificially superimposed Arabic text in news videos. Subsequently, there are other additional difficulties linked to the Arabic script, such as the cursive nature of the script and the presence of ligatures and diacritic marks. Another major challenge faced in this work is related to the absence of public text datasets dedicated to Arabic Video OCR systems. Actually, most of the existing Arabic text datasets are limited to printed or handwriting recognition tasks.

Starting from a clear understanding of the literature, we have suggested a new dataset and accurate methods to fill the aforementioned gaps. We highlight in the following our main contributions to the field of text detection and recognition in videos.

- A first contribution lies in the development of a method for Arabic text detection in video frames. The method represents a combination of a fully heuristic CC-based detection module and a machine learning-based verification stage. The first one makes use of an adapted version of the SWT operator to extract CCs, which are then filtered and merged by human-defined rules to form textlines. Whereas in the second stage, we train a CAE in unsupervised manner to produce features from the previously detected textline candidates. After that, an SVM classifier takes the AE-generated features as an input to distinguish text lines from non-text ones. We have achieved an F-Score of 85% (resp. 84%) for the HD (resp. SD) resolution on the detection dataset (AcTiV-D). Moreover, we have compared our method with two recently published ones using the same dataset, and the experimental results show the superiority of the proposed method. A particular strength of such a method is that it avoids the need for handcrafted features by using an unsupervised feature-learning scheme, namely CAE.

- As a second contribution, we have proposed an RNN-based method for Arabic video text recognition. The method relies specifically on a multidimensional LSTM network coupled with a CTC decoding layer. This network consists of five hierarchically structured layers, where three are LSTM hidden layers and two are feedforward subsampling layers. The used MDLSTM-CTC model operates directly on the raw image pixels and allows the modelization of text variations in four directions. Furthermore, we have introduced a novel preprocessing step to normalize the text polarity, in the input textline images, to dark text on light background. We have also suggested a compact representation of character models by grouping "beginning" - "middle" shapes and "end" - "isolated" ones. Our method has achieved 96.48% as a character recognition rate, 86.05% as a word recognition rate and 72.49% as a line recognition rate on the recognition dataset (AcTiV-R). The normalization step and the model set choice have brought significant recognition improvement in terms of reduction in the line error rate. More particularly, the compact representation of character models has allowed us to improve the behavior of our system, precisely in the training of the CTC layer, by increasing the quantity of training samples per character model. Moreover, we have outperformed the state-of-the-art results on the public dataset ALIF, specifically those based on the combination of CNN and BLSTM networks. Up to our knowledge, we have been the first to use the MDLSTM network for Arabic video text recognition. A major strength of our method is that it permits avoiding two hard OCR steps, namely textline segmentation and handcrafted feature extraction.
- One other important contribution of this thesis is the development of a standard dataset for Arabic Video OCR systems, called AcTiV for Arabic Text in Video. It consists of 189 news video clips, 4,063 text frames and 10,415 cropped text images. Actually, the video clips were recorded from four Arabic TV news channels during three years with a particular attention to ensure maximum diversity in text patterns and an important complexity in video environment. The proposed dataset has been used to train and evaluate our proposed detection and recognition methods. We have made AcTiV[1] public and freely available for the scientific community. It is also distributed with its annotation and evaluation tools that have been made open-source for standardization and validation purposes. Basically, AcTiV represents the first dataset designed to support the development and evaluation of Arabic Video OCR systems. More than twenty labs in the world are currently using this dataset. Besides, it served as a benchmark to compare the performances of participating systems in the first and second edition of "AcTiVComp" contests that we organized in conjunction with the ICPR 2016 and ICDAR 2017 conferences.

Future Work

We believe that this thesis has advanced the field of Arabic Video OCR by achieving noticeable improvements on text detection and recognition accuracies on two benchmark datasets.

[1]Available at http://tc11.cvc.uab.es/datasets/AcTiV_1
http://www.latis-eniso.org/content/fr/20/activ-data-base.html

Yet, our methods may fail to detect and recognize text objects in several cases due to some limitations. In the detection module, most of the errors are due to the sensitivity of SWT to edge defections and to the use of heuristic rules, especially in the first stage. Whereas, the main observed limitations, in the recognition module, are the failure in handling similar glyphs and less frequent shapes in the training database.

Accordingly, some possible future direction for the detection task are: (1) trying other CC extraction techniques, such as superpixel and MSER or one of its variants (CE-MSER, edge-enhanced MSERs...), which have recently won several competitions, and (2) replacing the combination of CAE and SVM by stacking a neural network on top of the auto-encoder, thus having the possibility to fine-tune the features for the classification task.

In order to further improve the recognition results, we intend to use linguistic information, namely language models, in our recurrent network. This can be achieved by introducing the probabilities of the characters estimated with an n-gram model in the decoding phase of the MDLSTM outputs. Hence, several errors could be removed and missed characters could be recovered.

We can also propose some long-term prospects that might help to improve performance:

- The classification of individual pixels as belonging to text or non-text, instead of working on CC level as an input to the classifier. This can be achieved by using a CNN (or one of its variants, e.g. FCRN), which can integrate feature extraction and classification together. Such networks have demonstrated state-of-the-art performance for text detection in recent years.
- The development of a text tracking system, which takes as an input the entire video sequence instead of individual text frames. The temporal redundancy is a key feature of video text; i.e., it remains on the screen for many consecutive frames (at least 2 seconds) in order to be readable. This redundant temporal information can be exploited by text tracking to (1) increase the chance of localizing text since the same text may appear under varying conditions from frame to frame, and (2) remove false text alarms in individual frames since they are usually not stable throughout time.

Appendices

Appendix A

Recognition System using RNNLib

1.1 Introduction

Based on [G+12], this chapter provides more details and comments about the use of the RNNLIB [1] toolkit in the field of text recognition. RNNLib was firstly introduced and used by Graves for sequence labelling problems, such as speech and handwriting recognition. The toolkit mainly implements the Long Short-Term Memory (LSTM) architecture. Its most important components are: (1) Bidirectional Long Short-Term Memory, (2) Connectionist Temporal Classification and (3) Multidimensional Recurrent Neural Networks. RNNLib also implements the hierarchical subsampling structure, which permits to efficiently label raw images and speech waveforms.

1.2 Data preparation

The first step is the preparation of data that we use during the two phases of learning and recognition. All RNNLib data files are in NetCDF (Network Common Data Form) format, a binary file format that support the creation and access of array-oriented scientific data. We Run the *./build_netcdf.sh* script to adapt our dataset format to the toolkit basic data files. The same script does all necessary preprocessing including normalization of the input and creates corresponding *.nc* files by processing every sequence in a file list. This file contains information about the input data, their dimensions (e.g., total number of data sequences, sum of lengths of all sequences, length of longest sequence tag string, number of distinct class labels) and some useful variables.

1.3 Training

This step consists in adjusting the weights so that the output data of the network will correspond to that desired. The following command line is used to start a training:

[1] https://sourceforge.net/projects/rnnl/

```
transcription.config
task transcription
hiddenType lstm
trainFile 158modelsPol.nc
dataFraction 1
maxTestsNoBest 20
hiddenSize 2,10,50
subsampleSize 6,20
hiddenBlock 1,4;1,4
inputBlock 2,4
learnRate 1e-4
momentum 0.9
optimiser rmsprop
verbose false
```

Figure 1.1: Sample of configuration file.

$rnnlib - -autosave = truetranscription.config$

where transcription.config is the configuration file that defines the network topology. Figure 1.1 depicts an example of a configuration file content: Three LSTM hidden layers of 2, 10 and 50 cells, two subsampling layers of 6 and 20 cells, 2 x 4 as size of the input block and 1 x 4 as size of the first and second hidden blocks. If the option "autosave" is set *true*, it allows to store all dynamic information about network weight changes and improved error measurements. After each training epoch, timestamped configuration files with dynamic information appended will be saved. In addition, a timestamped log file will be saved, containing all the console output. For instance, the following files were created at the end of a training task:

- transcription@2016.06.29-00.55.05.887968.best_ctcError.save
- transcription@2016.06.29-00.55.05.887968.best_labelError.save
- transcription@2016.06.29-00.55.05.887968.last.save
- transcription@2016.06.29-00.55.05.887968.log

```
data sequence:
file = Activ_R.nc
index = 31
tag = Data/testFiles/France24_vd02_frame_119-1.png
input shape = (120 70 1)
timesteps = 8400
target label sequence (length 3):
Miim_B Saad_B Raa_I
output label sequence (length 3):
Miim_B Saad_B Raa_I
output shape = (30 106)
errors:
ctcError 0.00166114
deletions 0
insertions 0
labelError 0
seqError 0
substitutions 0
```

Figure 1.2: Example of recognition result for one test image.

1.4 Test

For the recognition step, we use the *error_test.sh* script that takes two parameters as input, the trained model and the netCDF “testFile.nc”

./error_test.shtranscription@123.*best_labelError.save testFile.nc*

Finally, a *.log* file containing the system outputs of each text image was stored. Figure 1.2 shows an example of result for one test image. We can see the amount of deletion, insertion and substitution errors and some information about the target and output sequences. Figure 1.3 depicts an example of quantitative result on a set of 618 test images in terms of deletion, insertion and substitution errors, and overall error rates at character and line levels (i.e. "labelError" and "seqError" in the figure).

```
test errors:
ctcError 10.1763
deletions 1.03161%
insertions 0.539795%
labelError 5.90776%
seqError 42.7184%
substitutions 4.33635%
------------------------------
618 sequences tested in 1 minute 24.467 seconds
average 0.136678 seconds per sequence
```

Figure 1.3: Example of recognition result on a set of 618 images.

Appendix B

Organized competitions

2.1 Introduction

Amid the writing of this chapter, more than 20 research groups over the world have started to use the AcTiV dataset. In order to compare the performance of the different systems developed by those groups, we organized the first edition of the AcTiV Competition (AcTiVComp) in the framework of the 23$^{\text{rd}}$ International Conference on Pattern Recognition (ICPR'2016), during December 4-8, 2016, in Cancun, Mexico, and the second edition at the 14$^{\text{th}}$ International Conference on Document Analysis and Recognition (ICDAR'2017), during November 9-15, 2017, in Kyoto, Japan.
The main goal of both competitions was to objectively assess the performance of participants' algorithms to detect and recognize Arabic text in video frames.

2.2 AcTiVComp contests

Four groups with five systems have participated to the first edition of AcTiVComp. While in the second edition, three competitors have submitted five systems. Table 2.1 summarizes the characteristics of these systems in terms of related category, used heuristics, features and classifiers. In the detection challenge, the systems were compared based on the recall, precision and F-score metrics using our evaluation tool [ZTH+16]. The evaluation in the recognition task was at the character, word and line levels using the previously suggested metrics (see Chapter 3, section 3.4.2). The systems were tested in a blind manner on the closed-test set of the AcTiV dataset which was unknown to all participants. The competition protocols (see Tables 2.2 and 2.3) were defined to evaluate the ability of detection and recognition systems to handle different text sizes, colors and fonts using low resolution frames with complex background.

The best results in the detection challenge, of the first edition, were achieved by the FM-ATD system, which exploited geometric grouping over MSER regions and classified the regions using an AdaBoost classifier trained with gray-level features.
In the recognition challenge, the ATR-SID system scored best for most of the protocols. The system was based on an RNN+CTC architecture.

Table 2.1: Overview of the participating systems to the 1^{st} and 2^{sd} editions of AcTiVComp.

Submitted methods	**Description**	**Task**
ICPR 2016		
Arabic Text Detection based on Color Homogeneity (ATD-CH), by Houda Gaddour, MIRACL Lab., University of Sfax, TUNISIA	CC-based & Fully heuristic MSER-like technique, Geometric filtering, text candidates grouping	D
A Fast MSER-based Method for Arabic VideoText Detection (FM-ATD) by Xuehang Yang, NLPR, Chinese Academy of Sciences, CHINA	Hybrid approach: MSER, Gray-level features, AdaBoost classifier, False positive reduction	D
Detection of Arabic Text in Video Frames (D-ATVF) by Seiya Iwata, Mie University, JAPAN	Fully heuristic: Adaptive thresholding, Geometric filtering, False textline reduction	D
Recognition of Arabic Text in Video Frames (R-ATVF) by Seiya Iwata, Mie University, JAPAN	Segmentation-based Chain code histogram features MQDF classifier	R
Arabic Text Recognition in News Video Frames (ATR-SID) By Soumaya Essefi,National Engineering School of Sousse (ENISo), TUNISIA	Grayscale conversion, image resizing, MDRNN+CTC	R
ICDAR 2017		
THDL-Det system by Ruijie Yan, Tsinghua University, China	FCN-based feature extraction, Fast R-CNN classifier, Non-maximum suppression	D
CLS-Det system by Wenhao He, NLPR, Chinese Academy of Sciences, CHINA	Bi-task FCN	D
Deep System for Arabic Text Detection (DS-ATD) by Zied Selmi, REGIM Lab., Sfax, Tunisia	Hybrid approach: A set of preprocessing (HSV, Top hat transform, Gaussian blur), CNN classifier	D
THDL-Rec system by Ruijie Yan, Tsinghua University, China	Segmentation-free BRNNs (GRU-based) +CTC Dropout, Sparse training	R
DCR-Rec System by Yanfei Lv, NLPR, Chinese Academy of Sciences, CHINA	Segmentation-free BLSTM + CTC, Dropout, Batch normalization	R

In AcTiVComp 2.0, the THDL-Det system outperformed the other competitors (including those of the first edition) in all detection protocols with an F-score rates ranging from 0.8 to 0.9. This system was mainly based on the Fast R-CNN classifier and the NMS procedure. It provided an effective score of 0.85 for the channel-free protocol p7. This implies its generalization ability and robustness in detecting text regions regardless data resolution. We notice that the participating systems were affected by the image quality in protocol p4.3bis (SD 480x360,

used in the second edition only), with a significant decrease in the F-score metric, except for the DS-ATD system. Another interesting observation that can be drawn from the realized results is that all participating systems have used a quite similar CNN-based architecture, but differ in how they dealt with the original image in the first stage, i.e. proposal-based technique (TH-DL system), pixel-based classification (CLS-Det system) or a set of heuristic pre-processing steps (DS-ATD system). The latter could have an impact on the use of CNNs in the second stage.
For the recongition challenge, the DCR-Rec system have showed a superiority in the p3, p6.1, p6.2 and p6.3bis *channel-depending* protocols realizing a best LRR of 0.89 for HD resolution. The THDL-Rec system performed quite better in the p6.4 and p9 *channel-free* protocols as well as in the p6.3 protocol realizing a best LRR of 0.78 for the SD resolution. It is interesting to note that the obtained results in the global protocol p9, which were around 0.75 in terms of LRR, represents a significant improvement in the Arabic Video OCR field. An other important observation is that both systems use Bi-RNNs but in a different way. The first system used a hybrid RNN-CNN representation and an N-gram language model, while the second applied dropout and sparse training techniques.

Table 2.2: Detection Dataset and Evaluation Protocols

Protocol	**TV Channel**	**Training set**	**Test set**	**Closed-test set**
		#Frames	#Frames	#Frames
1	AljazeeraHD	337	87	103
4	France24 arabe	331	80	104
	RussiaToday arabic	323	79	100
	TunisiaNat1	492	116	106
	All SD channels	1,146	275	310
4bis	TunisiaNat Youtube	-	150	149
7	All channels	1,483	362	413

Table 2.3: Recognition Dataset and Evaluation Protocols."Lns" and "Wds" respectively denote "Lines" and "Words"

Protocol	**TV Channel**	**training set**			**test set**			**closed-test set**		
		#Lns	#Wds	#Chars	#Lns	#Wds	#Chars	#Lns	#Wds	#Chars
3	AlJazeeraHD	1,909	8,110	46,563	196	766	4,343	262	1,082	6,283
6	France24 arabe	1,906	5,683	32,085	179	667	3,835	191	734	4,600
	Russia Today arabic	2,127	13,462	78,936	250	1,483	8,749	256	1,598	9,305
	TunisiaNat1	2,001	9,338	54,809	189	706	4,087	221	954	5,597
	All SD channels	6,034	28,483	165,830	618	2,856	16,671	668	3,286	19,502
6bis	TunisiaNat1 Youtube	-	-	-	320	1,487	8,726	311	1,148	6,645
9	All channels	7,943	36,593	212,393	814	3,622	21,014	930	4,368	25,785

2.3 Conclusion

The AcTiVComp contests have attracted seven groups for participating and have received ten systems in total. The best results have been yielded by the system of Ruijie Yan (THDL-D) for all detection protocols. For the recognition task, the DCR-Rec system has scored best for the channel-depending protocols and the THDL-Rec has scored quite better for the channel-free protocols. The main difficulty for both text detectors and OCR systems was in the channel-free protocol where text was multi-font and multi-size. For more details about these competitions we refer to [ZHT+16, ZHIA17].

The obtained results can be further improved. Hence, we look forward to have more participants in the future editions of AcTiVComp and more researchers joining the Arabic video text detection and recognition research topic.

List of Figures

List of Tables

Glossary

Glossary - Acronyms

AR Aspect Ratio.

BB Bounding Box.

BLSTM Bidirectional Long Short Term Memory.

BRNN Bidirectional Recurrent Neural Networks.

CAE Convolutional Auto-Encoders.

CC Connected Component.

CNN Convolutional Neural Network.

CPD Contrast Preserving Decolorization.

CRF Conditional Random Field.

CTC Connectionist Temporal Classification.

DCT Discret Cosinus Transform.

D-SWT Discrete Stationary Wavelet Transform.

ER Extremal region.

FCN Fully Convolutional Networks.

FCRN Fully-Convolutional Regression Network.

FFT Fast Fourier Transform.

GAM Gradient Amplitude Map.

GMM Gaussian mixture modelling.

GLCM Gray-Level Co-occurrence Matrix.

HMM Hidden Markov Model.

HOG Histograms of Oriented Gradient.

KNN K-Nearest Neighbors.

LBP Local Binary Patterns.

LHBP Local Haar Binary Pattern.

LSTM Long Short Term Memory.

mb-LBP multi-block LBP.

MDF Mean Difference Feature.

MDLSTM Multidimensional Long Short Term Memory.

MDRNN Multidimensional Recurrent Neural Networks.

MGD Maximum Gradient Difference.

MLP Multi-layer perceptrons.

MSER Maximally Stable Extremal Regions.

MRF Markov Random Field.

OCR Optical Character Recognition.

PAWs Part of Arabic Words.

RF Random Forest.

RNN Recurrent Neural Network.

RPN Region Proposal Networks.

RLSA Run Length Smoothing Algorithm.

SGW Stroke Gabor words.

SVM Support Vector Machines.

SWT Stroke Width Transform.

TMMS Toggle Mapping Morphological Segmentation.

WMF Weighted Median Filter.

Bibliography

[AAAB15] Ibrahim Abdelaziz, Sherif Abdou, and Hassanin Al-Barhamtoshy. A large vocabulary system for arabic online handwriting recognition. *Pattern Analysis and Applications*, pages 1–13, 2015. (cited in 46).

[AB96] Najoua Ben Amara and Abdel Belaïd. Printed paw recognition based on planar hidden markov models. In *Pattern Recognition, 1996., Proceedings of the 13th International Conference on*, volume 2, pages 220–224. IEEE, 1996. (cited in 4 and 83).

[AGFV14] Jon Almazán, Albert Gordo, Alicia Fornés, and Ernest Valveny. Word spotting and recognition with embedded attributes. *IEEE transactions on pattern analysis and machine intelligence*, 36(12):2552–2566, 2014. (cited in 39).

[AGP10] Marios Anthimopoulos, Basilis Gatos, and Ioannis Pratikakis. A two-stage scheme for text detection in video images. *Image and Vision Computing*, 28(9):1413–1426, 2010. (cited in 30 and 60).

[AGP13] Marios Anthimopoulos, Basilis Gatos, and Ioannis Pratikakis. Detection of artificial and scene text in images and video frames. *Pattern Analysis and Applications*, 16(3):431–446, 2013. (cited in 30).

[AJQ14] Gheith A Abandah, Fuad T Jamour, and Esam A Qaralleh. Recognizing handwritten arabic words using grapheme segmentation and recurrent neural networks. *International Journal on Document Analysis and Recognition (IJDAR)*, 17(3):275–291, 2014. (cited in 36).

[AK09] SA Angadi and MM Kodabagi. A texture based methodology for text region extraction from low resolution natural scene images. *International Journal of Image Processing (IJIP)*, 3(5):229, 2009. (cited in 21 and 22).

[AP13] Ouais Alsharif and Joelle Pineau. End-to-end text recognition with hybrid hmm maxout models. *arXiv preprint arXiv:1310.1811*, 2013. (cited in 35 and 39).

[ASP91] Steve Austin, Richard Schwartz, and Paul Placeway. The forward-backward search algorithm. In *Acoustics, Speech, and Signal Processing, 1991. ICASSP-91., 1991 International Conference on*, pages 697–700. IEEE, 1991. (cited in 88).

[BKR+17] Ayan Kumar Bhunia, Gautam Kumar, Partha Pratim Roy, R Balasubramanian, and Umapada Pal. Text recognition in scene image and video frame using color channel selection. *Multimedia Tools and Applications*, pages 1–28, 2017. (cited in 36, 36, and 39).

[BSF94] Yoshua Bengio, Patrice Simard, and Paolo Frasconi. Learning long-term dependencies with gradient descent is difficult. *IEEE transactions on neural networks*, 5(2):157–166, 1994. (cited in 87).

[BYL13] Bo Bai, Fei Yin, and Cheng Lin Liu. Scene text localization using gradient local correlation. In *Document Analysis and Recognition (ICDAR), 2013 12th International Conference on*, pages 1380–1384. IEEE, 2013. (cited in 18, 18, and 19).

[CCC+11] Adam Coates, Blake Carpenter, Carl Case, Sanjeev Satheesh, Bipin Suresh, Tao Wang, David J Wu, and Andrew Y Ng. Text detection and character recognition in scene images with unsupervised feature learning. In *Document Analysis and Recognition (ICDAR), 2011 International Conference on*, pages 440–445. IEEE, 2011. (cited in 27 and 28).

[CKCS17] Ch'ng Chee Kheng and Chan Chee Seng. Total-text: A comprehensive dataset for scene text detection and recognition. In *2017 International Conference on Document Analysis and Recognition*, pages 935–942. IEEE, 2017. (cited in 46).

[CKL10] 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. (cited in 33).

[CL11] Chih-Chung Chang and Chih-Jen Lin. Libsvm: a library for support vector machines. *ACM transactions on intelligent systems and technology (TIST)*, 2(3):27, 2011. (cited in 75).

[CO05] Datong Chen and Jean-Marc Odobez. Video text recognition using sequential monte carlo and error voting methods. *Pattern Recognition Letters*, 26(9):1386–1403, 2005. (cited in 2).

[CRC16] Youssouf Chherawala, Partha Pratim Roy, and Mohamed Cheriet. Feature set evaluation for offline handwriting recognition systems: application to the recurrent neural network model. *IEEE transactions on cybernetics*, 46(12):2825–2836, 2016. (cited in 40, 40, and 40).

[CSJ16] Hojin Cho, Myungchul Sung, and Bongjin Jun. Canny text detector: Fast and robust scene text localization algorithm. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3566–3573, 2016. (cited in 14).

[CSLK11] Min Su Cho, Jae-Hyun Seok, Seonghun Lee, and Jin Hyung Kim. Scene text extraction by superpixel crfs combining multiple character features. In *Document Analysis and Recognition (ICDAR), 2011 International Conference on*, pages 1034–1038. IEEE, 2011. (cited in 32 and 34).

[CTS+11] Huizhong Chen, Sam S Tsai, Georg Schroth, David M Chen, Radek Grzeszczuk, and Bernd Girod. Robust text detection in natural images with edge-enhanced maximally stable extremal regions. In *Image Processing (ICIP), 2011 18th IEEE International Conference on*, pages 2609–2612. IEEE, 2011. (cited in 9, 10, 10, 13, 16, 18, 18, 19, and 115).

[CV95] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995. (cited in 74).

[CYHL15] Kai Chen, Fei Yin, Amir Hussain, and Cheng-Lin Liu. Efficient text localization in born-digital images by local contrast-based segmentation. In *Document Analysis and Recognition (ICDAR), 2015 13th International Conference on*, pages 291–295. IEEE, 2015. (cited in 15, 17, 21, and 30).

[DCC+07] Viet Cuong Dinh, Seong Soo Chun, Seungwook Cha, Hanjin Ryu, and Sanghoon Sull. An efficient method for text detection in video based on stroke width similarity. In *Asian Conference on Computer Vision*, pages 200–209. Springer, 2007. (cited in 14).

[DM00] David Doermann and David Mihalcik. Tools and techniques for video performance evaluation. In *Pattern Recognition, 2000. Proceedings. 15th International Conference on*, volume 4, pages 167–170. IEEE, 2000. (cited in 50).

[EAKMA11] Haikal El Abed, Monji Kherallah, Volker Märgner, and Adel M Alimi. On-line arabic handwriting recognition competition. *International Journal on Document Analysis and Recognition (IJDAR)*, 14(1):15–23, 2011. (cited in 83).

[EAMKA09] Haikal El Abed, Volker Märgner, Monji Kherallah, and Adel M Alimi. Icdar 2009 online arabic handwriting recognition competition. In *Document Analysis and Recognition, 2009. ICDAR'09. 10th International Conference on*, pages 1388–1392. IEEE, 2009. (cited in 4).

[EGMS12] Khaoula Elagouni, Christophe Garcia, Franck Mamalet, and Pascale Sébillot. Text recognition in videos using a recurrent connectionist approach. In *International Conference on Artificial Neural Networks*, pages 172–179. Springer, 2012. (cited in 30, 38, and 38).

[EGMS14] Khaoula Elagouni, Christophe Garcia, Franck Mamalet, and Pascale Sébillot. Text recognition in multimedia documents: a study of two neural-based ocrs using and avoiding character segmentation. *International Journal on Document Analysis and Recognition (IJDAR)*, 17(1):19–31, 2014. (cited in 35 and 39).

[EOW10] Boris Epshtein, Eyal Ofek, and Yonatan Wexler. Detecting text in natural scenes with stroke width transform. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 2963–2970. IEEE, 2010. (cited in 4, 11, 11, 14, 14, 21, 66, 68, 69, 77, 79, 79, 79, 79, 79, 115, 117, and 117).

[FGG05] Silvio Ferreira, Vincent Garin, and Bernard Gosselin. A text detection technique applied in the framework of a mobile camera-based application. In *International Workshop on Camera-based Document Analysis and Recognition (CBDAR*, 2005. (cited in 21).

[FMC09] Jonathan Fabrizio, Beatriz Marcotegui, and Matthieu Cord. Text segmentation in natural scenes using toggle-mapping. In *Image Processing (ICIP), 2009 16th IEEE International Conference on*, pages 2373–2376. IEEE, 2009. (cited in 29).

[FRSD+16] Jonathan Fabrizio, Myriam Robert-Seidowsky, Séverine Dubuisson, Stefania Calarasanu, and Raphaël Boissel. Textcatcher: a method to detect curved and challenging text in natural scenes. *International Journal on Document Analysis and Recognition (IJDAR)*, 19(2):99–117, 2016. (cited in 29, 29, and 116).

[FSZ16] Yuanyuan Feng, Yonghong Song, and Yuanlin Zhang. Scene text detection based on multi-scale swt and edge filtering. In *Pattern Recognition (ICPR), 2016 23rd International Conference on*, pages 645–650. IEEE, 2016. (cited in 14, 15, 15, 15, and 16).

[FTB12] Mehdi Felhi, Salvatore Tabbone, and Nicolas Bonnier. Un nouveau descripteur de texte pour la détection des lignes de texte multi-orientées dans les scènes réelles. In *7ème Colloque International Francophone sur l'Écrit et le Document-CIFED 2012*, 2012. (cited in 11).

[FZMEB+12] Volkmar Frinken, Francisco Zamora-Martínez, Salvador Espana-Boquera, Maria José Castro-Bleda, Andreas Fischer, and Horst Bunke. Long-short term memory neural networks language modeling for handwriting recognition. In *Pattern Recognition (ICPR), 2012 21st International Conference on*, pages 701–704. IEEE, 2012. (cited in 97).

[G+12] Alex Graves et al. *Supervised sequence labelling with recurrent neural networks*, volume 385. Springer, 2012. (cited in 86, 86, 87, 91, 92, 92, 107, 118, 118, and 118).

[GBYB12] Alvaro Gonzalez, Luis M Bergasa, J Javier Yebes, and Sebastián Bronte. Text location in complex images. In *Pattern Recognition (ICPR), 2012 21st International Conference on*, pages 617–620. IEEE, 2012. (cited in 29).

[GCWL17] Yunze Gao, Yingying Chen, Jinqiao Wang, and Hanqing Lu. Reading scene text with attention convolutional sequence modeling. *arXiv preprint arXiv:1709.04303*, 2017. (cited in 36, 37, 38, 39, 45, and 116).

[GF15] Shaho Ghanei and Karim Faez. Robust localization of texts in real-world images. *International Journal of Pattern Recognition and Artificial Intelligence*, 29(07):1555012, 2015. (cited in 10, 11, 21, and 115).

[GFGS06] Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In *Proceedings of the 23rd international conference on Machine learning*, pages 369–376. ACM, 2006. (cited in 88).

[GFS07] Alex Graves, Santiago Fernandez, and Juergen Schmidhuber. Multi-dimensional recurrent neural networks. *arXiv preprint arXiv:0705.2011*, 2007. (cited in 85).

[GGJ16] Deepika Ghai, Divya Gera, and Neelu Jain. A new approach to extract text from images based on dwt and k-means clustering. *International Journal of Computational Intelligence Systems*, 9(5):900–916, 2016. (cited in 21, 23, and 28).

[GKV16] Houda Gaddour, Slim Kanoun, and Nicole Vincent. A new method for arabic text detection in natural scene image based on the color homogeneity. In *International Conference on Image and Signal Processing*, pages 127–136. Springer, 2016. (cited in 14, 16, 16, 21, 78, 79, 79, 79, 79, and 79).

[GLF+09] Alex Graves, Marcus Liwicki, Santiago Fernández, Roman Bertolami, Horst Bunke, and Jürgen Schmidhuber. A novel connectionist system for unconstrained handwriting recognition. *IEEE transactions on pattern analysis and machine intelligence*, 31(5):855–868, 2009. (cited in 84).

[GMAJ13] Vibhor Goel, Anand Mishra, Karteek Alahari, and CV Jawahar. Whole is greater than sum of parts: Recognizing scene text words. In *Document Analysis and Recognition (ICDAR), 2013 12th International Conference on*, pages 398–402. IEEE, 2013. (cited in 40).

[Gra12] Alex Graves. Offline arabic handwriting recognition with multidimensional recurrent neural networks. In *Guide to OCR for Arabic scripts*, pages 297–313. Springer, 2012. (cited in 40 and 84).

[GSG+17] Raul Gomez, Baoguang Shi, Lluis Gomez, Lukas Numann, Andreas Veit, and Jiri Matas. Icdar2017 robust reading challenge on coco-text. In *2017 International Conference on Document Analysis and Recognition*, pages 1435–1443. IEEE, 2017. (cited in 46).

[GVZ16] Ankush Gupta, Andrea Vedaldi, and Andrew Zisserman. Synthetic data for text localisation in natural images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2315–2324, 2016. (cited in 27 and 28).

[GWX+13] Song Gao, Chunheng Wang, Baihua Xiao, Cunzhao Shi, Yang Zhang, Zhijian Lv, and Yanqin Shi. Adaptive scene text detection based on transferring adaboost. In *Document Analysis and Recognition (ICDAR), 2013 12th International Conference on*, pages 388–392. IEEE, 2013. (cited in 21, 21, 25, and 28).

[HAV+12] Mohamed Ben Halima, Adel Alimi, Ana Fernández Vila, et al. Nf-savo: Neuro-fuzzy system for arabic video ocr. *arXiv preprint arXiv:1211.2150*, 2012. (cited in 4, 34, 34, 34, 39, 83, and 116).

[HHL14] Shih Chang Hsia, Cheng Nan Ho, and Chien Hung Liu. Real-time text detection using pac/due embedded system. In *Intelligent Information Hiding and Multimedia Signal Processing (IIH-MSP), 2014 Tenth International Conference on*, pages 321–324. IEEE, 2014. (cited in 22).

[HHQ+16] Pan He, Weilin Huang, Yu Qiao, Chen Change Loy, and Xiaoou Tang. Reading scene text in deep convolutional sequences. In *Thirtieth AAAI Conference on Artificial Intelligence*, 2016. (cited in 36 and 38).

[HHQY16] Tong He, Weilin Huang, Yu Qiao, and Jian Yao. Text-attentional convolutional neural network for scene text detection. *IEEE transactions on image processing*, 25(6):2529–2541, 2016. (cited in 14, 17, and 18).

[HLYW13] Weilin Huang, Zhe Lin, Jianchao Yang, and Jue Wang. Text localization in natural images using stroke feature transform and text covariance descriptors. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1241–1248, 2013. (cited in 9, 14, 15, 17, 21, 44, 44, 44, and 116).

[HMZ09] Xiaodong Huang, Huadong Ma, and He Zhang. A new video text extraction approach. In *Multimedia and Expo, 2009. ICME 2009. IEEE International Conference on*, pages 650–653. IEEE, 2009. (cited in 34).

[How11] Nicholas R Howe. A laplacian energy for document binarization. In *Document Analysis and Recognition (ICDAR), 2011 International Conference on*, pages 6–10. IEEE, 2011. (cited in 33 and 116).

[HP09] Shehzad Muhammad Hanif and Lionel Prevost. Text detection and localization in complex scene images using constrained adaboost algorithm. In *Document Analysis and Recognition, 2009. ICDAR'09. 10th International Conference on*, pages 1–5. IEEE, 2009. (cited in 21, 25, and 28).

[HQT14] Weilin Huang, Yu Qiao, and Xiaoou Tang. Robust scene text detection with convolution neural network induced mser trees. In *European Conference on Computer Vision*, pages 497–511. Springer, 2014. (cited in 13, 21, and 30).

[HS97] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997. (cited in 87).

[HST13] Kaiming He, Jian Sun, and Xiaoou Tang. Guided image filtering. *IEEE transactions on pattern analysis and machine intelligence*, 35(6):1397–1409, 2013. (cited in 10).

[HWL15] Ping Hu, Weiqiang Wang, and Ke Lu. A novel binarization approach for text in images. In *Image Processing (ICIP), 2015 IEEE International Conference on*, pages 956–960. IEEE, 2015. (cited in 30 and 32).

[HYH+16] Dafang He, Xiao Yang, Wenyi Huang, Zihan Zhou, Daniel Kifer, and C Lee Giles. Aggregating local context for accurate scene text detection. In *Asian Conference on Computer Vision*, pages 280–296. Springer, 2016. (cited in 18 and 20).

[IOWK16] Seiya Iwata, Wataru Ohyama, Tetsushi Wakabayashi, and Fumitaka Kimura. Recognition and transition frame detection of arabic news captions for video retrieval. In *Pattern Recognition (ICPR), 2016 23rd International Conference on*, pages 4005–4010. IEEE, 2016. (cited in 35 and 97).

[IP13] Andrej Ikica and Peter Peer. Swt voting-based color reduction for text detection in natural scene images. *EURASIP journal on advances in signal processing*, 2013(1):95, 2013. (cited in 14 and 15).

[JJ+15] Munho Jeong, Kang-Hyun Jo, et al. Multi language text detection using fast stroke width transform. In *Frontiers of Computer Vision (FCV), 2015 21st Korea-Japan Joint Workshop on*, pages 1–4. IEEE, 2015. (cited in 14).

[JPZ+14] Arpit Jain, Xujun Peng, Xiaodan Zhuang, Pradeep Natarajan, and Huaigu Cao. Text detection and recognition in natural scenes and consumer videos. In *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*, pages 1245–1249. IEEE, 2014. (cited in 13).

[JSVZ14] Max Jaderberg, Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Synthetic data and artificial neural networks for natural scene text recognition. *arXiv preprint arXiv:1406.2227*, 2014. (cited in 30, 36, 37, and 39).

[JSVZ16] Max Jaderberg, Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Reading text in the wild with convolutional neural networks. *International Journal of Computer Vision*, 116(1):1–20, 2016. (cited in 18, 36, 37, and 44).

[JVZ14] Max Jaderberg, Andrea Vedaldi, and Andrew Zisserman. Deep features for text spotting. In *European conference on computer vision*, pages 512–528. Springer, 2014. (cited in 27 and 28).

[JWS09] Zhong Ji, Jian Wang, and Yu-Ting Su. Text detection in video frames using hybrid features. In *Machine Learning and Cybernetics, 2009 International Conference on*, volume 1, pages 318–322. IEEE, 2009. (cited in 23 and 28).

[JXY+08] Rongrong Ji, Pengfei Xu, Hongxun Yao, Zhen Zhang, Xiaoshuai Sun, and Tianqiang Liu. Directional correlation analysis of local haar binary pattern for text detection. In *Multimedia and Expo, 2008 IEEE International Conference on*, pages 885–888. IEEE, 2008. (cited in 25, 25, 28, and 116).

[KAJK16] Sul-Ho Kim, Kwon-Jae An, Seok-Woo Jang, and Gye-Young Kim. Texture feature-based text region segmentation in social multimedia data. *Multimedia Tools and Applications*, 75(20):12815–12829, 2016. (cited in 21 and 22).

[KEBE15] Akram Khémiri, Afef Kacem Echi, Abdel Belaïd, and Mourad Elloumi. Arabic handwritten words off-line recognition based on hmms and dbns. In *Document Analysis and Recognition (ICDAR), 2015 13th International Conference on*, pages 51–55. IEEE, 2015. (cited in 4).

[KEBE16] Akram Khémiri, Afef Kacem Echi, Abdel Belaïd, and Mourad Elloumi. A system for off-line arabic handwritten word recognition based on bayesian approach. In *Frontiers in Handwriting Recognition (ICFHR), 2016 15th International Conference on*, pages 560–565. IEEE, 2016. (cited in 4).

[KGBN+15] 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*, pages 1156–1160. IEEE, 2015. (cited in 45 and 62).

[KGS+09] 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 Transactions on Pattern Analysis and Machine Intelligence*, 31(2):319–336, 2009. (cited in 60 and 62).

[KJ11] Sukhwinder Kaur and Gurpreet Singh Josan. Gurmukhi text extraction from image using support vector machine (svm). *International Journal of Engineering Science and Technology (IJEST)*, 2011. (cited in 17).

[KJM13] S Karthikeyan, Vignesh Jagadeesh, and BS Manjunath. Learning bottom-up text attention maps for text detection using stroke width transform. In *Image Processing (ICIP), 2013 20th IEEE International Conference on*, pages 3312–3316. IEEE, 2013. (cited in 14 and 15).

[KK09] Wonjun Kim and Changick Kim. A new approach for overlay text detection and extraction from complex video scene. *IEEE transactions on image processing*, 18(2):401–411, 2009. (cited in 11, 12, and 115).

[KK13] Hyung Il Koo and Duck Hoon Kim. Scene text detection via connected component clustering and nontext filtering. *IEEE transactions on image processing*, 22(6):2296–2305, 2013. (cited in 17).

[KKR07] Thotreingam Kasar, Jayant Kumar, and AG Ramakrishnan. Font and background color independent text binarization. In *Second international workshop on camera-based document analysis and recognition*, pages 3–9, 2007. (cited in 33 and 116).

[KMM+11] Dimosthenis Karatzas, S Robles Mestre, Joan Mas, Farshad Nourbakhsh, and P Pratim Roy. Icdar 2011 robust reading competition-challenge 1: reading text in born-digital images (web and email). In *Document Analysis and Recognition (ICDAR), 2011 International Conference on*, pages 1485–1490. IEEE, 2011. (cited in 33).

[KP10] Saurav Kumar and Andrew Perrault. Text detection on nokia n900 using stroke width transform. *no. Cornell University*, 2010. (cited in 11).

[KSR15] Vijeta Khare, Palaiahnakote Shivakumara, and Paramesran Raveendran. A new histogram oriented moments descriptor for multi-oriented moving text detection in video. *Expert Systems with Applications*, 42(21):7627–7640, 2015. (cited in 21).

[KSU+13] Dimosthenis Karatzas, Faisal Shafait, Seiichi Uchida, Masakazu Iwamura, Lluis Gomez i Bigorda, Sergi Robles Mestre, Joan Mas, David Fernandez Mota, Jon Almazan Almazan, and Lluis Pere de las Heras. Icdar 2013 robust reading competition. In *2013 12th International Conference on Document Analysis and Recognition*, pages 1484–1493. IEEE, 2013. (cited in 44, 45, and 62).

[KTA+11] Monji Kherallah, Najiba Tagougui, Adel M Alimi, Haikal El Abed, and Volker Margner. Online arabic handwriting recognition competition. In *2011 International Conference on Document Analysis and Recognition*, pages 1454–1458. IEEE, 2011. (cited in 4 and 46).

[LCJK10] SeongHun Lee, Min Su Cho, Kyomin Jung, and Jin Hyung Kim. Scene text extraction with edge constraint and text collinearity. In *Pattern Recognition (ICPR), 2010 20th International Conference on*, pages 3983–3986. IEEE, 2010. (cited in 45).

[LG06] Liana M Lorigo and Venugopal Govindaraju. Offline arabic handwriting recognition: a survey. *IEEE transactions on pattern analysis and machine intelligence*, 28(5):712–724, 2006. (cited in 4, 46, and 83).

[LH97] Jisheng Liang and Robert M Haralick. Performance evaluation of document layout analysis algorithms on the uw data set. volume 3027, pages 149–161, 1997. (cited in 60).

[LJSvdH14] Yao Li, Wenjing Jia, Chunhua Shen, and Anton van den Hengel. Characterness: An indicator of text in the wild. *IEEE transactions on image processing*, 23(4):1666–1677, 2014. (cited in 10).

[LK95] Seong-Whan Lee and Young-Joon Kim. A new type of recurrent neural network for handwritten character recognition. In *Document Analysis and Recognition, 1995., Proceedings of the Third International Conference on*, volume 1, pages 38–41. IEEE, 1995. (cited in 85).

[LL12] Yao Li and Huchuan Lu. Scene text detection via stroke width. In *Pattern Recognition (ICPR), 2012 21st International Conference on*, pages 681–684. IEEE, 2012. (cited in 18, 18, and 19).

[LLL+11] Jung-Jin Lee, Pyoung-Hean Lee, Seong-Whan Lee, Alan Yuille, and Christof Koch. Adaboost for text detection in natural scene. In *Document Analysis and Recognition (ICDAR), 2011 International Conference on*, pages 429–434. IEEE, 2011. (cited in 24, 24, and 28).

[LLP96] Seong-Whan Lee, Dong-June Lee, and Hee-Seon Park. A new methodology for gray-scale character segmentation and recognition. *IEEE transactions on pattern analysis and machine intelligence*, 18(10):1045–1050, 1996. (cited in 35).

[LLQ+17] Zhandong Liu, Yong Li, Xiangwei Qi, Yong Yang, Mei Nian, Haijun Zhang, and Reziwanguli Xiamixiding. Method for unconstrained text detection in natural scene image. *IET Computer Vision*, 11(7):596–604, 2017. (cited in 45).

[LO16] Chen-Yu Lee and Simon Osindero. Recursive recurrent nets with attention modeling for ocr in the wild. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2231–2239, 2016. (cited in 38, 39, and 39).

[LPS+05] Simon M Lucas, Alex Panaretos, Luis Sosa, Anthony Tang, Shirley Wong, Robert Young, Kazuki Ashida, Hiroki Nagai, Masayuki Okamoto, Hiroaki Yamamoto, et al. Icdar 2003 robust reading competitions: entries, results, and future directions. *International Journal of Document Analysis and Recognition (IJDAR)*, 7(2-3):105–122, 2005. (cited in 32 and 44).

[LPTL14] Tong Lu, Shivakumara Palaiahnakote, Chew Lim Tan, and Wenyin Liu. *Video Text Detection*. Springer Publishing Company, Incorporated, 2014. (cited in 8, 9, 30, and 44).

[LSB+17] Minghui Liao, Baoguang Shi, Xiang Bai, Xinggang Wang, and Wenyu Liu. Textboxes: A fast text detector with a single deep neural network. In *AAAI*, pages 4161–4167, 2017. (cited in 45).

[Luc05] Simon M Lucas. Icdar 2005 text locating competition results. In *Document Analysis and Recognition, 2005. Proceedings. Eighth International Conference on*, pages 80–84. IEEE, 2005. (cited in 44, 44, 44, 60, and 116).

[LXJ12] Cewu Lu, Li Xu, and Jiaya Jia. Contrast preserving decolorization. In *Computational Photography (ICCP), 2012 IEEE International Conference on*, pages 1–7. IEEE, 2012. (cited in 11).

[MAAK+14] Sabri A Mahmoud, Irfan Ahmad, Wasfi G Al-Khatib, Mohammad Alshayeb, Mohammad Tanvir Parvez, Volker Märgner, and Gernot A Fink. Khatt: An open arabic offline handwritten text database. *Pattern Recognition*, 47(3):1096–1112, 2014. (cited in 4 and 46).

[MAJ17] Anand Mishra, Karteek Alahari, and CV Jawahar. Unsupervised refinement of color and stroke features for text binarization. *International Journal on Document Analysis and Recognition (IJDAR)*, 20(2):105–121, 2017. (cited in 32, 33, 33, 33, 33, 33, 34, 45, 116, 116, and 116).

[MB01] U-V Marti and Horst Bunke. Using a statistical language model to improve the performance of an hmm-based cursive handwriting recognition system. *International journal of Pattern Recognition and Artificial intelligence*, 15(01):65–90, 2001. (cited in 36).

[MBH12] Ali Mosleh, Nizar Bouguila, and A Ben Hamza. Image text detection using a bandlet-based edge detector and stroke width transform. In *BMVC*, pages 1–12, 2012. (cited in 14 and 17).

[MBH13] Ali Mosleh, Nizar Bouguila, and Abdessamad Ben Hamza. Automatic inpainting scheme for video text detection and removal. *IEEE Transactions on image processing*, 22(11):4460–4472, 2013. (cited in 11 and 21).

[MBN+13] 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. (cited in 32, 33, 34, and 116).

[MCUP04] Jiri Matas, Ondrej Chum, Martin Urban, and Tomás Pajdla. Robust wide-baseline stereo from maximally stable extremal regions. *Image and vision computing*, 22(10):761–767, 2004. (cited in 12).

[MCZ18] Sadek Mansouri, Mbarek Charhad, and Mounir Zrigui. A heuristic approach to detect and localize text on arabic news video. *Computación y Sistemas*, 22(1), 2018. (cited in 13).

[MEA08] Volker Märgner and Haikal El Abed. Databases and competitions: strategies to improve arabic recognition systems. *Arabic and Chinese Handwriting Recognition*, pages 82–103, 2008. (cited in 4).

[MEA12] Volker Märgner and Haikal El Abed. *Guide to OCR for arabic scripts.* Springer, 2012. (cited in 4, 46, and 83).

[MFSS18] Ali Mirza, Marium Fayyaz, Zunera Seher, and Imran Siddiqi. Urdu caption text detection using textural features. In *Proceedings of the 2nd Mediterranean Conference on Pattern Recognition and Artificial Intelligence*, pages 70–75. ACM, 2018. (cited in 21).

[MKKEA12] Anis Mezghani, Slim Kanoun, Maher Khemakhem, and Haikal El Abed. A database for arabic handwritten text image recognition and writer identification. In *Frontiers in Handwriting Recognition (ICFHR), 2012 International Conference on*, pages 399–402. IEEE, 2012. (cited in 4).

[ML15] Ronaldo Messina and Jérôme Louradour. Segmentation-free handwritten chinese text recognition with lstm-rnn. In *Document Analysis and Recognition (ICDAR), 2015 13th International Conference on*, pages 171–175. IEEE, 2015. (cited in 40 and 84).

[MM13] Mohieddin Moradi and Saeed Mozaffari. Hybrid approach for farsi/arabic text detection and localisation in video frames. *IET Image Processing*, 7(2):154–164, 2013. (cited in 26).

[MP07] Stéphane Mallat and Gabriel Peyré. A review of bandlet methods for geometrical image representation. *Numerical Algorithms*, 44(3):205–234, 2007. (cited in 11).

[MTC+14] 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*, 122:92–104, 2014. (cited in 9).

[Naz75] A Nazif. A system for the recognition of the printed arabic characters, 1975. (cited in 3).

[NDMW11] Robert Nagy, Anders Dicker, and Klaus Meyer-Wegener. Neocr: A configurable dataset for natural image text recognition. In *International Workshop on Camera-Based Document Analysis and Recognition*, pages 150–163. Springer, 2011. (cited in 45).

[NGP11] Konstantinos Ntirogiannis, Basilis Gatos, and Ioannis Pratikakis. Binarization of textual content in video frames. In *Document Analysis and Recognition (ICDAR), 2011 International Conference on*, pages 673–677. IEEE, 2011. (cited in 32 and 34).

[Nib85] Wayne Niblack. *An introduction to digital image processing.* Strandberg Publishing Company, 1985. (cited in 31 and 32).

[NM10] Lukas Neumann and Jiri Matas. A method for text localization and recognition in real-world images. In *Asian Conference on Computer Vision*, pages 770–783. Springer, 2010. (cited in 13 and 40).

[NM13] Luka Neumann and Jiri Matas. Scene text localization and recognition with oriented stroke detection. In *Computer Vision (ICCV), 2013 IEEE International Conference on*, pages 97–104. IEEE, 2013. (cited in 40 and 40).

[NUA+17] Saeeda Naz, Arif I Umar, Riaz Ahmad, Saad B Ahmed, Syed H Shirazi, and Muhammad I Razzak. Urdu nasta'liq text recognition system based on multi-dimensional recurrent neural network and statistical features. *Neural Computing and Applications*, 28(2):219–231, 2017. (cited in 37, 39, and 92).

[Ots79] Nobuyuki Otsu. A threshold selection method from gray-level histograms. *IEEE transactions on systems, man, and cybernetics*, 9(1):62–66, 1979. (cited in 31, 33, and 116).

[PBKL14] Vu Pham, Théodore Bluche, Christopher Kermorvant, and Jérôme Louradour. Dropout improves recurrent neural networks for handwriting recognition. In *Frontiers in Handwriting Recognition (ICFHR), 2014 14th International Conference on*, pages 285–290. IEEE, 2014. (cited in 40, 92, and 97).

[PHL11] Yi-Feng Pan, Xinwen Hou, and Cheng-Lin Liu. A hybrid approach to detect and localize texts in natural scene images. *IEEE Transactions on Image Processing*, 20(3):800–813, 2011. (cited in 15, 20, and 29).

[PMM+02] Mario Pechwitz, S Snoussi Maddouri, Volker Märgner, Noureddine Ellouze, Hamid Amiri, et al. Ifn/enit-database of handwritten arabic words. In *Proc. of CIFED*, volume 2, pages 127–136. Citeseer, 2002. (cited in 4 and 46).

[PSST11] Trung Quy Phan, Palaiahnakote Shivakumara, Bolan Su, and Chew Lim Tan. A gradient vector flow-based method for video character segmentation. In *Document Analysis and Recognition (ICDAR), 2011 International Conference on*, pages 1024–1028. IEEE, 2011. (cited in 35, 35, and 116).

[PYKY16] Wei-Yi Pei, Chun Yang, Lih-Jen Kau, and Xu-Cheng Yin. Multi-orientation scene text detection with multi-information fusion. In *Pattern Recognition (ICPR), 2016 23rd International Conference on*, pages 657–662. IEEE, 2016. (cited in 17, 18, and 21).

[QDGJ16] Guo Qiang, Tu Dan, Li Guohui, and Lei Jun. Memory matters: Convolutional recurrent neural network for scene text recognition. *arXiv preprint arXiv:1601.01100*, 2016. (cited in 38 and 39).

[QLWS07] Xueming Qian, Guizhong Liu, Huan Wang, and Rui Su. Text detection, localization, and tracking in compressed video. *Signal Processing: Image Communication*, 22(9):752–768, 2007. (cited in 21 and 21).

[RAS13] Ahsen Raza, Ali Abidi, and Imran Siddiqi. Multilingual artificial text detection and extraction from still images. In *Document Recognition and Retrieval XX*, volume 8658, page 86580V. International Society for Optics and Photonics, 2013. (cited in 26, 26, 28, and 116).

[RHGS15] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015. (cited in 27).

[RM03] Xiaofeng Ren and Jitendra Malik. Learning a classification model for segmentation. In *null*, page 10. IEEE, 2003. (cited in 15).

[Rob94] Anthony J Robinson. An application of recurrent nets to phone probability estimation. *IEEE transactions on Neural Networks*, 5(2):298–305, 1994. (cited in 85).

[RRS+13] Sangheeta Roy, Partha Pratim Roy, Palaiahnakote Shivakumara, Georgios Louloudis, Chew Lim Tan, and Umapada Pal. Hmm-based multi oriented text recognition in natural scene image. In *Pattern Recognition (ACPR), 2013 2nd IAPR Asian Conference on*, pages 288–292. IEEE, 2013. (cited in 36, 36, and 39).

[RSDE13] Ahsen Raza, Imran Siddiqi, Chawki Djeddi, and Abdellatif Ennaji. Multilingual artificial text detection using a cascade of transforms. In *Document Analysis and Recognition (ICDAR), 2013 12th International Conference on*, pages 309–313. IEEE, 2013. (cited in 21, 21, 24, and 26).

[RSPM13] Jose A Rodriguez-Serrano, Florent Perronnin, and France Meylan. Label embedding for text recognition. 2013. (cited in 39).

[RSR+15] Sangheeta Roy, Palaiahnakote Shivakumara, Partha Pratim Roy, Umapada Pal, Chew Lim Tan, and Tong Lu. Bayesian classifier for multi-oriented video text recognition system. *Expert Systems with Applications*, 42(13):5554–5566, 2015. (cited in 8, 30, 32, and 34).

[SAM+14] Fouad Slimane, Sameh Awaida, Anis Mezghani, Mohammad Tanvir Parvez, Slim Kanoun, Sabri A Mahmoud, and Volker Märgner. Icfhr2014 competition on arabic writer identification using ahtid/mw and khatt databases. In *Frontiers in Handwriting Recognition (ICFHR), 2014 14th International Conference on*, pages 797–802. IEEE, 2014. (cited in 4 and 46).

[SBY17] Baoguang Shi, Xiang Bai, and Cong Yao. An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition. *IEEE transactions on pattern analysis and machine intelligence*, 39(11):2298–2304, 2017. (cited in 38 and 38).

[SCS09] Temucin Som, Dogan Can, and Murat Saraclar. Hmm-based sliding video text recognition for turkish broadcast news. In *Computer and Information Sciences, 2009. ISCIS 2009. 24th International Symposium on*, pages 475–479. IEEE, 2009. (cited in 36).

[SDTP10] Palaiahnakote Shivakumara, Anjan Dutta, Chew Lim Tan, and Umapada Pal. A new wavelet-median-moment based method for multi-oriented video text detection. In *Proceedings of the 9th IAPR International Workshop on Document Analysis Systems*, DAS '10, pages 279–286. ACM, 2010. (cited in 23).

[SDTP14] Palaiahnakote Shivakumara, Anjan Dutta, Chew Lim Tan, and Umapada Pal. Multi-oriented scene text detection in video based on wavelet and angle projection boundary growing. *Multimedia tools and applications*, 72(1):515–539, 2014. (cited in 21 and 24).

[SG07] Zohra Saidane and Christophe Garcia. Robust binarization for video text recognition. In *Document Analysis and Recognition, 2007. ICDAR 2007. Ninth International Conference on*, volume 2, pages 874–879. IEEE, 2007. (cited in 32 and 34).

[SG08] Zohra Saidane and Christophe Garcia. An automatic method for video character segmentation. In *International Conference Image Analysis and Recognition*, pages 557–566. Springer, 2008. (cited in 35).

[SGD09] Zohra Saidane, Christophe Garcia, and Jean Luc Dugelay. The image text recognition graph (itrg). In *Multimedia and Expo, 2009. ICME 2009. IEEE International Conference on*, pages 266–269. IEEE, 2009. (cited in 35 and 35).

[SHJC15] Lei Sun, Qiang Huo, Wei Jia, and Kai Chen. A robust approach for text detection from natural scene images. *Pattern Recognition*, 48(9):2906–2920, 2015. (cited in 14 and 18).

[SIAH08] Fouad Slimane, Rolf Ingold, Adel M Alimi, and Jean Hennebert. Duration models for arabic text recognition using hidden markov models. In *Computational Intelligence for Modelling Control & Automation, 2008 International Conference on*, pages 838–843. IEEE, 2008. (cited in 83).

[SIK+09] Fouad Slimane, Rolf Ingold, Slim Kanoun, Adel M Alimi, and Jean Hennebert. A new arabic printed text image database and evaluation protocols. In *2009 10th International Conference on Document Analysis and Recognition*, pages 946–950. IEEE, 2009. (cited in 4, 46, and 51).

[SIL16] Mathias Seuret, Rolf Ingold, and Marcus Liwicki. N-light-n: A highly-adaptable java library for document analysis with convolutional auto-encoders and related architectures. In *Frontiers in Handwriting Recognition (ICFHR), 2016 15th International Conference on*, pages 459–464. IEEE, 2016. (cited in 72).

[SK17] Usman Shahzad and Khurram Khurshid. Oriental-script text detection and extraction in videos. In *Arabic Script Analysis and Recognition (ASAR), 2017 1st International Workshop on*, pages 15–20. IEEE, 2017. (cited in 11).

[SKEA$^+$11] Fouad Slimane, Slim Kanoun, Haikal El Abed, Adel M Alimi, Rolf Ingold, and Jean Hennebert. Icdar 2011-arabic recognition competition: Multi-font multi-size digitally represented text. In *2011 International Conference on Document Analysis and Recognition*, pages 1449–1453. IEEE, 2011. (cited in 4).

[SKEA$^+$13] Fouad Slimane, Slim Kanoun, Haikal El Abed, Adel M Alimi, Rolf Ingold, and Jean Hennebert. Icdar2013 competition on multi-font and multi-size digitally represented arabic text. In *2013 12th International Conference on Document Analysis and Recognition*, pages 1433–1437. IEEE, 2013. (cited in 46).

[SKH$^+$99] Toshio Sato, Takeo Kanade, Ellen K Hughes, Michael A Smith, and Shin'ichi Satoh. Video ocr: indexing digital news libraries by recognition of superimposed captions. *Multimedia Systems*, 7(5):385–395, 1999. (cited in 8).

[SL14] Bolan Su and Shijian Lu. Accurate scene text recognition based on recurrent neural network. In *Asian Conference on Computer Vision*, pages 35–48. Springer, 2014. (cited in 30, 37, and 39).

[SLT12] Bolan Su, Shijian Lu, and Chew Lim Tan. A learning framework for degraded document image binarization using markov random field. In *Pattern Recognition (ICPR), 2012 21st International Conference on*, pages 3200–3203. IEEE, 2012. (cited in 32).

[SNDC07] Krishna Subramanian, Premkumar Natarajan, Michael Decerbo, and David Castanon. Character-stroke detection for text-localization and extraction. In *Document Analysis and Recognition, 2007. ICDAR 2007. Ninth International Conference on*, volume 1, pages 33–37. IEEE, 2007. (cited in 14).

[SP97] Mike Schuster and Kuldip K Paliwal. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing*, 45(11):2673–2681, 1997. (cited in 84 and 85).

[SP00] Jaakko Sauvola and Matti Pietikäinen. Adaptive document image binarization. *Pattern recognition*, 33(2):225–236, 2000. (cited in 31).

[SPLT11] Palaiahnakote Shivakumara, Trung Quy Phan, Shijian Lu, and Chew Lim Tan. Video character recognition through hierarchical classification. In *Document Analysis and Recognition (ICDAR), 2011 International Conference on*, pages 131–135. IEEE, 2011. (cited in 35 and 39).

[SPT09] Palaiahnakote Shivakumara, Trung Quy Phan, and Chew Lim Tan. A robust wavelet transform based technique for video text detection. In *Document Analysis and Recognition, 2009. ICDAR'09. 10th International Conference on*, pages 1285–1289. IEEE, 2009. (cited in 21, 22, 23, 23, 23, 28, and 116).

[SPT10] Palaiahnakote Shivakumara, Trung Quy Phan, and Chew Lim Tan. New fourier-statistical features in rgb space for video text detection. *IEEE Transactions on Circuits and Systems for Video Technology*, 20(11):1520–1532, 2010. (cited in 21, 21, 23, and 23).

[SPT11] Palaiahnakote Shivakumara, Trung Quy Phan, and Chew Lim Tan. A laplacian approach to multi-oriented text detection in video. *IEEE transactions on pattern analysis and machine intelligence*, 33(2):412–419, 2011. (cited in 21, 24, and 28).

[SR98] Andrew W Senior and Anthony J Robinson. An off-line cursive handwriting recognition system. *IEEE transactions on pattern analysis and machine intelligence*, 20(3):309–321, 1998. (cited in 84 and 85).

[SR12] Imran Siddiqi and Ahsen Raza. A database of artificial urdu text in video images with semi-automatic text line labeling scheme. In *The Fourth International Conferences on Advances in Multimedia (MMEDIA'12)*, 2012. (cited in 50).

[SSD11] Asif Shahab, Faisal Shafait, and Andreas Dengel. Icdar 2011 robust reading competition challenge 2: Reading text in scene images. In *2011 international conference on document analysis and recognition*, pages 1491–1496. IEEE, 2011. (cited in 44 and 44).

[SWTF16] Iwata Seiya, Ohyama Wataru, Wakabayashi Tetsushi, and Kimura Fumitaka. Recognition and transition frame detection of arabic news captions for video retrieval. In *23th International Conference on Pattern Recognition (ICPR)*, pages 4005–4010. IEEE, 2016. (cited in 4, 9, 15, 15, 16, 21, 36, 39, 79, 79, 79, 79, 79, 79, and 116).

[SWX+13] Cunzhao Shi, Chunheng Wang, Baihua Xiao, Yang Zhang, and Song Gao. Scene text detection using graph model built upon maximally stable extremal regions. *Pattern recognition letters*, 34(2):107–116, 2013. (cited in 17).

[SX15] Feng Su and Hailiang Xu. Robust seed-based stroke width transform for text detection in natural images. In *Document Analysis and Recognition (ICDAR), 2015 13th International Conference on*, pages 916–920. IEEE, 2015. (cited in 14 and 15).

[SXWZ12] Cunzhao Shi, Baihua Xiao, Chunheng Wang, and Yang Zhang. Adaptive graph cut based binarization of video text images. In *Document Analysis Systems (DAS), 2012 10th IAPR International Workshop on*, pages 58–62. IEEE, 2012. (cited in 32).

[SZ14] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. (cited in 29).

[SZK+12] Fouad Slimane, Oussama Zayene, Slim Kanoun, Adel M Alimi, Jean Hennebert, and Rolf Ingold. New features for complex arabic fonts in cascading recognition system. In *Pattern Recognition (ICPR), 2012 21st International Conference on*, pages 738–741. IEEE, 2012. (cited in 40 and 84).

[TAA07] Sameh Masmoudi Touj, Najoua Essoukri Ben Amara, and Hamid Amiri. A hybrid approach for off-line arabic handwriting recognition based on a planar hidden markov modeling. In *Document Analysis and Recognition, 2007. ICDAR 2007. Ninth International Conference on*, volume 2, pages 964–968. IEEE, 2007. (cited in 83 and 84).

[TGLZ02] Xiaoou Tang, Xinbo Gao, Jianzhuang Liu, and Hongjiang Zhang. A spatial-temporal approach for video caption detection and recognition. *IEEE Transactions on Neural Networks*, 13(4):961–971, 2002. (cited in 8).

[THA15] Houssem Turki, Mohamed Ben Halima, and Adel M Alimi. Scene text detection images with pyramid image and mser enhanced. In *Intelligent Systems Design and Applications (ISDA), 2015 15th International Conference on*, pages 301–306. IEEE, 2015. (cited in 9 and 17).

[THH+16] Zhi Tian, Weilin Huang, Tong He, Pan He, and Yu Qiao. Detecting text in natural image with connectionist text proposal network. In *European Conference on Computer Vision*, pages 56–72. Springer, 2016. (cited in 27).

[TPB+] S Tsai, Vasu Parameswaran, Jerome Berclaz, Ramakrishna Vedantham, Radek Grzeszczuk, and Bernd Girod. Design of a text detection system via hypothesis generation and verification. (cited in 10).

[VMN+16] Andreas Veit, Tomas Matera, Lukas Neumann, Jiri Matas, and Serge Belongie. Coco-text: Dataset and benchmark for text detection and recognition in natural images. *arXiv preprint arXiv:1601.07140*, 2016. (cited in 46).

[VTPEK16] Matias Valdenegro-Toro, Paul Plöger, Stefan Eickeler, and Iuliu Konya. Histograms of stroke widths for multi-script text detection and verification in road scenes. *IFAC-PapersOnLine*, 49(15):100–107, 2016. (cited in 19, 19, 19, 21, and 116).

[VV05] Alexander Vezhnevets and Vladimir Vezhnevets. Modest adaboost-teaching adaboost to generalize better. volume 12, 2005. (cited in 24).

[WB10] Kai Wang and Serge Belongie. Word spotting in the wild. In *European Conference on Computer Vision*, pages 591–604. Springer, 2010. (cited in 33, 40, and 45).

[Wer90] Paul J Werbos. Backpropagation through time: what it does and how to do it. *Proceedings of the IEEE*, 78(10):1550–1560, 1990. (cited in 86).

[WFCL17] Cong Wang, Yin Fei, and Liu Cheng-Lin. Scene text detection with novel superpixel based character candidate extraction. In *2017 International Conference on Document Analysis and Recognition*, pages 929–934. IEEE, 2017. (cited in 9, 15, 17, 18, and 21).

[WGLZ17] Fenglei Wang, Qiang Guo, Jun Lei, and Jun Zhang. Convolutional recurrent neural networks with hidden markov model bootstrap for scene text recognition. *IET Computer Vision*, 11(6):497–504, 2017. (cited in 38 and 38).

[WJ04] Christian Wolf and J-M Jolion. Extraction and recognition of artificial text in multimedia documents. *Formal Pattern Analysis & Applications*, 6(4):309–326, 2004. (cited in 31, 33, and 116).

[WJ06] Christian Wolf and Jean-Michel Jolion. Object count/area graphs for the evaluation of object detection and segmentation algorithms. *International Journal of Document Analysis and Recognition (IJDAR)*, 8(4):280–296, 2006. (cited in 60).

[WJQ+17] Fei Wang, Mengqing Jiang, Chen Qian, Shuo Yang, Cheng Li, Honggang Zhang, Xiaogang Wang, and Xiaoou Tang. Residual attention network for image classification. *arXiv preprint arXiv:1704.06904*, 2017. (cited in 39).

[WJW04] Rongrong Wang, Wanjun Jin, and Lide Wu. A novel video caption detection approach using multi-frame integration. In *Pattern Recognition, 2004. ICPR 2004. Proceedings of the 17th International Conference on*, volume 1, pages 449–452. IEEE, 2004. (cited in 8).

[WPW16] Yaqi Wang, Liangrui Peng, and Shengjin Wang. A multi-stage method for chinese text detection in news videos. *Procedia Computer Science*, 96:1409–1417, 2016. (cited in 15).

[WWCN12] Tao Wang, David J Wu, Adam Coates, and Andrew Y Ng. End-to-end text recognition with convolutional neural networks. In *Pattern Recognition (ICPR), 2012 21st International Conference on*, pages 3304–3308. IEEE, 2012. (cited in 27, 27, 37, and 116).

[XXS14] Hailiang Xu, Like Xue, and Feng Su. Scene text detection based on robust stroke width transform and deep belief network. In *Asian Conference on Computer Vision*, pages 195–209. Springer, 2014. (cited in 9, 11, 14, and 16).

[YBG14] Sonia Yousfi, Sid-Ahmed Berrani, and Christophe Garcia. Arabic text detection in videos using neural and boosting-based approaches: Application to video indexing. In *2014 IEEE International Conference on Image Processing (ICIP)*, pages 3028–3032. IEEE, 2014. (cited in 4, 21, 27, 27, 28, 28, 30, and 116).

[YBG15a] 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*, pages 1221–1225. IEEE, 2015. (cited in 46, 98, 98, 98, 99, 99, 99, 99, and 118).

[YBG15b] Sonia Yousfi, Sid-Ahmed Berrani, and Christophe Garcia. Deep learning and recurrent connectionist-based approaches for arabic text recognition in videos. In *Document Analysis and Recognition (ICDAR), 2015 13th International Conference on*, pages 1026–1030. IEEE, 2015. (cited in 38, 38, 38, 39, 83, 99, 99, 99, 99, and 116).

[YBL+12] Cong Yao, Xiang Bai, Wenyu Liu, Yi Ma, and Zhuowen Tu. Detecting texts of arbitrary orientations in natural images. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 1083–1090. IEEE, 2012. (cited in 10, 14, 17, 20, 21, and 45).

[YBSL14] Cong Yao, Xiang Bai, Baoguang Shi, and Wenyu Liu. Strokelets: A learned multi-scale representation for scene text recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4042–4049, 2014. (cited in 39).

[YD15] Qixiang Ye and David Doermann. Text detection and recognition in imagery: A survey. *IEEE transactions on pattern analysis and machine intelligence*, 37(7):1480–1500, 2015. (cited in 8, 9, and 44).

[YQS12] Haojin Yang, Bernhard Quehl, and Harald Sack. Text detection in video images using adaptive edge detection and stroke width verification. In *Systems, Signals and Image Processing (IWSSIP), 2012 19th International Conference on*, pages 9–12. IEEE, 2012. (cited in 11 and 15).

[YSBS15] Mohammad Reza Yousefi, Mohammad Reza Soheili, Thomas M Breuel, and Didier Stricker. A comparison of 1d and 2d lstm architectures for the recognition of handwritten arabic. In *DRR*, page 94020H, 2015. (cited in 40 and 40).

[YSM+15] Chong Yu, Yonghong Song, Quan Meng, Yuanlin Zhang, and Yang Liu. Text detection and recognition in natural scene with edge analysis. *IET Computer Vision*, 9(4):603–613, 2015. (cited in 15).

[YT11] Chucai Yi and Yingli Tian. Text detection in natural scene images by stroke gabor words. In *Document Analysis and Recognition (ICDAR), 2011 International Conference on*, pages 177–181. IEEE, 2011. (cited in 12, 18, 18, 19, 20, 21, 24, 24, and 115).

[YT12] Chucai Yi and Yingli Tian. Localizing text in scene images by boundary clustering, stroke segmentation, and string fragment classification. *IEEE Transactions on Image Processing*, 21(9):4256–4268, 2012. (cited in 14, 15, 15, and 115).

[YYHH14] Xu-Cheng Yin, Xuwang Yin, Kaizhu Huang, and Hong-Wei Hao. Robust text detection in natural scene images. *IEEE transactions on pattern analysis and machine intelligence*, 36(5):970–983, 2014. (cited in 13, 18, and 20).

[YYHI12] Xuwang Yin, Xu-Cheng Yin, Hong-Wei Hao, and Khalid Iqbal. Effective text localization in natural scene images with mser, geometry-based grouping and adaboost. In *Pattern Recognition (ICPR), 2012 21st International Conference on*, pages 725–728. IEEE, 2012. (cited in 17).

[YYZ+15] Junjie Yan, Yinan Yu, Xiangyu Zhu, Zhen Lei, and Stan Z Li. Object detection by labeling superpixels. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5107–5116, 2015. (cited in 15).

[YZTL16] Xu-Cheng Yin, Ze-Yu Zuo, Shu Tian, and Cheng-Lin Liu. Text detection, tracking and recognition in video: A comprehensive survey. *IEEE Transactions on Image Processing*, 25(6):2752–2773, 2016. (cited in 9).

[ZCLX16] Chuanlei Zhai, Zhineng Chen, Jie Li, and Bo Xu. Chinese image text recognition with blstm-ctc: a segmentation-free method. In *Chinese Conference on Pattern Recognition*, pages 525–536. Springer, 2016. (cited in 30 and 37).

[ZHIA17] Oussama Zayene, Jean Hennebert, Rolf Ingold, and Najoua Essoukri Ben Amara. Icdar2017 competition on arabic text detection and recognition in multi-resolution video frames. In *2017 International Conference on Document Analysis and Recognition*, pages 1460–1465. IEEE, 2017. (cited in 114).

[ZHT+15] Oussama Zayene, Jean Hennebert, Sameh Masmoudi Touj, Rolf Ingold, and Najoua Essoukri Ben Amara. A dataset for arabic text detection, tracking and recognition in news videos-activ. In *13th International Conference on Document Analysis and Recognition (ICDAR), 2015*, pages 996–1000. IEEE, 2015. (cited in 48, 77, 79, 79, 79, 79, and 79).

[ZHT+16] Oussama Zayene, Nadia Hajjej, Sameh Masmoudi Touj, Soumaya Ben Mansour, Jean Hennebert, Rolf Ingold, and Najoua Essoukri Ben Amara. Icpr2016 contest on arabic text detection and recognition in video frames-activcomp. In *Pattern Recognition (ICPR), 2016 23rd International Conference on*, pages 187–191. IEEE, 2016. (cited in 114).

[ZL15] Yun-Zhi Zhuge and Hu-Chuan Lu. Robust video text detection with morphological filtering enhanced mser. *Journal of Computer science and Technology*, 30(2):353, 2015. (cited in 9, 10, 13, 16, 18, 20, and 21).

[ZLC+17] W Zhu, J Lou, L Chen, Q Xia, and M Ren. Scene text detection via extremal region based double threshold convolutional network classification. *PLoS ONE*, 12(8):e0182227, 2017. (cited in 17).

[ZLL10] Zhou Zhiwei, Li Linlin, and Tan Chew Lim. Edge based binarization for video text images. In *Pattern Recognition (ICPR), 2010 20th International Conference on*, pages 133–136. IEEE, 2010. (cited in 30, 32, and 34).

[ZLMZ11] Gang Zhou, Yuehu Liu, Quan Meng, and Yuanlin Zhang. Detecting multilingual text in natural scene. In *Access Spaces (ISAS), 2011 1st International Symposium on*, pages 116–120. IEEE, 2011. (cited in 21, 21, 25, 25, and 116).

[ZLY+11] Hongwei Zhang, Changsong Liu, Cheng Yang, Xiaoqing Ding, and KongQiao Wang. An improved scene text extraction method using conditional random field and optical character recognition. In *Document Analysis and Recognition (ICDAR), 2011 International Conference on*, pages 708–712. IEEE, 2011. (cited in 32).

[ZST+16] Oussama Zayene, Mathias Seuret, Sameh M Touj, Jean Hennebert, Rolf Ingold, and Najoua E Ben Amara. Text detection in arabic news video based on swt operator and convolutional auto-encoders. In *Document Analysis Systems (DAS), 2016 12th IAPR Workshop on*, pages 13–18. IEEE, 2016. (cited in 79, 79, 79, 79, 79, and 81).

[ZTH+14] Oussama Zayene, Sameh Masmoudi Touj, Jean Hennebert, Rolf Ingold, and Najoua Essoukri Ben Amara. Semi-automatic news video annotation framework for arabic text. In *2014 4th International Conference on Image Processing Theory, Tools and Applications (IPTA)*, pages 1–6. IEEE, 2014. (cited in 50, 51, and 116).

[ZTH+16] Oussama Zayene, Sameh Masmoudi Touj, Jean Hennebert, Rolf Ingold, and Najoua Essoukri Ben Amara. Data, protocol and algorithms for performance evaluation of text detection in arabic news video. In *Advanced Technologies for Signal and Image Processing (ATSIP), 2016 2nd International Conference on*, pages 258–263. IEEE, 2016. (cited in 60, 77, and 111).

[ZTH+18] Oussama Zayene, Sameh Masmoudi Touj, Jean Hennebert, Rolf Ingold, and Najoua Essoukri Ben Amara. Open datasets and tools for arabic text detection and recognition in news video frames. *Journal of Imaging*, 4(2):32, 2018. (cited in 49).

[ZW13] Zhike Zhang and Weiqiang Wang. A novel approach for binarization of overlay text. In *2013 IEEE International Conference on Systems, Man, and Cybernetics*, pages 4259–4264. IEEE, 2013. (cited in 30, 32, and 34).

[ZYB16] Yingying Zhu, Cong Yao, and Xiang Bai. Scene text detection and recognition: Recent advances and future trends. *Frontiers of Computer Science*, 10(1):19–36, 2016. (cited in 8, 44, 44, and 116).

[ZZ17] Yuanping Zhu and Kuang Zhang. Text segmentation using superpixel clustering. *IET Image Processing*, 11(7):455–464, 2017. (cited in 46).

Publication List

1. **Oussama Zayene**, Sameh Masmoudi Touj, Jean Hennebert, Rolf Ingold and Najoua Essoukri Ben Amara. MDLSTM Networks for Artificial Arabic Text Recognition in News Video. In *IET Computer Vision Journal, (12) 5, July 2018.*

2. **Oussama Zayene**, Sameh Masmoudi Touj, Jean Hennebert, Rolf Ingold and Najoua Essoukri Ben Amara. Datasets and Tools for Arabic Text Detection and Recognition in News Video Frames. In *Journal of Imaging, (4) 2, January 2018.*

3. **Oussama Zayene**, Jean Hennebert, Rolf Ingold and Najoua Essoukri Ben Amara. ICDAR2017 Competition on Arabic Text Detection and Recognition in Multi-resolution Video Frames. In *14^{th} International Conference on Document Analysis and Recognition (ICDAR), 2017.*

4. **Oussama Zayene**, Soumaya Essefi Amamou and Najoua Essoukri Ben Amara. Arabic Video Text Recognition Based on Multi-Dimensional Recurrent Neural Networks. In *14^{th} ACS/IEEE International Conference on Computer Systems and Applications (AICCSA), pages 725-729 2017.*

5. **Oussama Zayene**, Nadia Hajjej, Sameh Masmoudi Touj, Soumaya Ben Mansour, Jean Hennebert, Rolf Ingold and Najoua Essoukri Ben Amara. ICPR2016 Contest on Arabic Text Detection and Recognition in Video Frames —AcTiVComp. In *23^{rd} International Conference on Pattern Recognition (ICPR), pages 187-191, 2016.*

6. **Oussama Zayene**, Mathias Seuret, Sameh Masmoudi Touj, Jean Hennebert, Rolf Ingold and Najoua Essoukri Ben Amara. Text Detection in Arabic news Video Based on SWT Operator and Convolutional Auto-encoders. In *12^{th} IAPR Workshop on Document Analysis Systems(DAS), pages 13-18, 2016.*

7. **Oussama Zayene**, Sameh Masmoudi Touj, Jean Hennebert, Rolf Ingold and Najoua Essoukri Ben Amara. Data, protocol and algorithms for performance evaluation of text detection in Arabic news video. In *2^{nd} International Conference on Advanced Technologies for Signal and Image Processing (ATSIP), pages 258-263, 2016.*

8. **Oussama Zayene**, Jean Hennebert, Sameh Masmoudi Touj, Rolf Ingold and Najoua Essoukri Ben Amara. A dataset for Arabic text detection, tracking and recognition in news videos —AcTiV. In *13^{th} International Conference on Document Analysis and Recognition (ICDAR), pages 996-1000, 2015.*

9. **Oussama Zayene**, Sameh Masmoudi Touj, Jean Hennebert, Rolf Ingold and Najoua Essoukri Ben Amara. Semi-automatic news video annotation framework for Arabic text. In *4th International Conference on Image Processing Theory, Tools and Applications (IPTA), pages 1-6, 2014.*