Word Spotting and Recognition in Images from Heterogeneous Sources

Suman Kumar Ghosh

Word Spotting and Recognition in Images from Heterogeneous Sources

A dissertation submitted by **Suman Kumar Ghosh** at Universitat Autònoma de Barcelona to fulfil the degree of **Doctor of Philosophy**.

Bellaterra, September 14, 2018

Director:	**Dr. Ernest Valveny** Universitat Autònoma de Barcelona Dept. Ciències de la Computació & Centre de Visió per Computador
Thesis Committee	**Prof. Dr.-Ing. Gernot A. Fink** Department of Computer science TU Dortmund University, Germany **Dr. Alicia Fornes** Dept. Ciències de la Computació and Centre de Visió per Computador Universitat Autònoma de Barcelona **Dr. Jon Amazan** Naver Labs Europe Greenoble, France

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

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

ISBN

Printed by

Universitat Autònoma de Barcelona

Dept. Ciències de la Computació

DECLARATION

I declare that I am the sole author of this thesis entitled "**Word Spotting and Recognition in Images from Heterogeneous Sources**" and that the work contained therein is original, where explicitly stated otherwise in the text.

Suman Kumar Ghosh

Bellaterra (Barcelona), September 14, 2018

Universitat Autònoma de Barcelona

Dept. Ciències de la Computació

Word Spotting and Recognition in Images from Heterogeneous Sources

Thesis presented to obtain the PhD in Computer Science

Author: Suman Kumar Ghosh

PhD Advisor: Dr. Ernest Valveny

CERTIFY:
that the dissertation "**Word Spotting and Recognition in Images from Heterogeneous Sources**" presented by Suman Kumar Ghosh to obtain the PhD in Computer science, has been done under his direction at the Universitat Autònoma de Barcelona.

Dr. Ernest Valveny
Bellaterra (Barcelona), September 14, 2018

Dedicated to Prof. Hij Bij Bij.

Ackknowledgements

At the end of this thesis, I would like to take some time to thank all the people without whom this project would never have been possible. Although it is just my name on the cover, many people have contributed to the research in their own particular way and for that I want to give them special thanks. First, my supervisor Dr. Ernest Valveny, have created the invaluable space for me to do this research and develop myself as a researcher in the best possible way. I greatly appreciate the freedom you have given me to find my own path and the guidance and support you offered when needed. I truly hope that we will be given the opportunity to work together in the future. I also want to take a moment to thank my advisor Prof. Andrew D. Bagdanov, MICC (Media Integration and Communication Center) at university of Florence, Italy. Thank you for investing time and guiding my way during my visit.

I would also like to thank Jon Almazan, whose work formed the basis of this research. Special thanks to Jon for providing a nice code repository to start with.

Thanks to all my fellow researchers in CVC, specially Anguelos Nicalou and Anjan Dutta for their continuous support.

Finally, the passion for research in me is due to my friends (Dr. Anandarup Roy, Dr. Jiaul H. Paik, Dr. Bikash Shaw, Oendrila Samanta, Dr. Tapan K. Bhowmik, Dr. Manika Kar) and colleagues (Dr. Ujjwal Bhattacharya, Prof. B.B. Chaudhuri, Prof. Swapan K. Parui) in Computer Vision and Pattern Recognition Unit at Indian Statistical Institute, Kolkata, for that they deserve a big thanks.

Of course, family always give thankless support for which I shall always be indebted. I would also want to take this opportunity to give thank you for all my family members.

Abstract

Text is the most common way of information sharing from ages. With recent development of images databases of handwritten historic manuscripts the demand for algorithms to make these databases accessible for browsing and indexing are in rise. Exponential increase of publicly available image databases and personal collections of pictures, this interest now also embraces text understanding on natural images.

Enabling search or understanding large collection of manuscripts or image databases need fast and robust method. Researchers have found different ways to represent cropped word for understanding and matching, which works well in when words are already segmented. However there is no trivial way to extend these for non-segmented documents.

In this thesis we explore different methods for text retrieval and recognition from unsegmented document and scene images.

Two different ways of representation exists in literature, one uses a fixed length representation learned from cropped words and another a sequence of features of variable length. Throughout this thesis, we have studied both these representation for their suitability in segmentation free understanding of text. In the first part we are focused on segmentation free word spotting using a fixed length representation. We extended the use of one successful representation for segmentation free retrieval. In the second part of the thesis, we explore sequence based features and finally propose a unified solution where same framework can generate both kind of representations.

Resum

El text es la manera mès habitual d'intercanviar informació des de les edats. Amb el desenvolupament recent d'imatges de bases de dades de manuscrits manuscrits històrics, la demanda d'algorismes per fer accessibles aquestes bases de dades per a la navegació i la indexació estan augmentant. L'augment exponencial de les bases de dades d'imatges disponibles públicament i les colÂ·leccions personals d'imatges, aquest interès també inclou l'enteniment del text sobre imatges naturals.

Activar la cerca o comprendre una gran collecció de manuscrits o bases de dades d'imatges requereix un mètode rápid i robust. Els investigadors han trobat diferents maneres de representar paraules retallades per a la comprensió i la concordanca, cosa que funciona bé quan les paraules ja estan segmentades. No obstant això, no hi ha cap manera trivial d'estendre'ls per a documents no segmentats.

En aquesta tesi, explorem diferents mètodes per a la recuperació i el reconeixement de text a partir d'imatges sense escena de documents i escenes.

Hi ha dues formes de representació diferents a la literatura, s'utilitza una representació de longitud fixa a partir de paraules retallades i una altra amb una seqüència de característiques de longitud variable. Al llarg d'aquesta tesi, hem estudiat aquestes dues representacions per la seva idoneítat en la lliure comprensió del text. A la primera part, ens hem centrat en la segmentació de paraules lliures amb una representació de longitud fixa. Vam ampliar l'ús d'una representació exitosa per a la recuperació gratuïta de segmentacions. En la segona part de la tesi, explorem funcions basades en seqüéncies i, finalment, proposem una solució unificada on el mateix marc pot generar ambdós tipus de representacions.

Resumen

El texto es la forma màs comùn de compartir informaciùn desde edades. Con el reciente desarrollo de bases de datos de imágenes de manuscritos históricos manuscritos, la demanda de algoritmos para hacer accesibles estas bases de datos para la navegación y la indexación están en aumento. El aumento exponencial de las bases de datos de imágenes disponibles al público y las colecciones personales de imágenes, este interés ahora también abarca la comprensión de texto en imágenes naturales.

La habilitación de la búsqueda o la comprensión de una gran colección de manuscritos o bases de datos de imágenes necesita un método rápido y sólido. Los investigadores han encontrado diferentes maneras de representar la palabra recortada para la comprensión y la coincidencia, que funciona bien cuando las palabras ya están segmentadas. Sin embargo, no hay una forma trivial de ampliar estos para documentos no segmentados.

En esta tesis, exploramos diferentes métodos para la recuperación y el reconocimiento de texto a partir de documentos no segmentados e imágenes de escena.

Existen dos formas diferentes de representación en la literatura, una usa una representación de longitud fija aprendida de palabras recortadas y otra una secuencia de características de longitud variable. A lo largo de esta tesis, hemos estudiado estas dos representaciones por su idoneidad en la segmentación, la comprensión libre del texto. En la primera parte, nos centramos en la detección de palabras sin segmentación utilizando una representación de longitud fija. Extendimos el uso de una representación exitosa para la recuperación libre de segmentación.

En la segunda parte de la tesis, exploramos las características basadas en secuencia y finalmente proponemos una solución unificada donde el mismo marco puede generar ambos tipos de representaciones.

Contents

1 Introduction **1**
- 1.1 Background . . . 1
  - 1.1.1 Taxonomies . . . 2
- 1.2 Related Work . . . 3
  - 1.2.1 Word spotting in handwritten manuscripts . . . 3
  - 1.2.2 Text Recognition in handwritten manuscripts . . . 5
  - 1.2.3 Text recognition in scene text . . . 5
  - 1.2.4 Contribution and organization of the thesis . . . 7

2 Protocols and Dataset **9**
- 2.1 Datasets . . . 9
  - 2.1.1 Handwritten documents dataset . . . 9
  - 2.1.2 Scene text datsets . . . 11
- 2.2 Protocol . . . 11
  - 2.2.1 Word Spotting . . . 12
  - 2.2.2 Recognition . . . 13

3 Representing documents with attributes **15**
- 3.1 Related Works . . . 16
- 3.2 Pyramidal Histogram of Character . . . 17
  - 3.2.1 Word Representation . . . 17
  - 3.2.2 Attribute-based word image representation . . . 18
- 3.3 Segmentation Free Word Spotting with Attribute representation . . . 19
  - 3.3.1 Sliding widow - baseline . . . 20
  - 3.3.2 Integral Image based sliding window . . . 21
  - 3.3.3 Indexing based on character bi-gram presence . . . 22

PHOB representation 23
Index generation 23
Segmentation 24
Indexing by bi-grams 24
Retrieval 25
3.4 Experimental Results 26
3.4.1 QBE word spotting 26
3.4.2 QBS Word Spottig 27
3.4.3 Time complexity Analysis 27
3.5 Conclusion 28

4 Text Box Proposals **29**
4.1 Overall Architecture 30
4.2 Pre-segmentation: Generation of Text Box Broposals 31
4.2.1 Generation of Atomic Bounding Boxes 32
4.2.2 Combining Atomic Bounding Boxes into Text Box Proposals 33
Constraint by Co-linearity 33
Constraint by Spatial Ordering 34
4.2.3 Filtering Text Box Proposals 35
4.3 Efficient Computation of Word Attributes using Integral Image 37
4.4 N-gram based Representation for Indexing 39
4.5 Retrieval 40
4.5.1 Retrieval of Relevant Text Boxes using the Index 40
4.5.2 Ranking of Text Boxes 41
4.6 Experimental Results 42
4.6.1 Evaluation of Text Box Proposals 42
4.6.2 Evaluation of Word Spotting 45
QBE Word Spotting 45
QBS Word Spotting 46
4.6.3 Word Spotting with N-gram Indexing 47
4.6.4 Discussion and Comparison with Related Works 48
4.6.5 Qualitative Results 50
4.7 Conclusions 51

5 Deep Features for Segmentation Free Word Spotting **55**
5.1 Related Works 56
5.2 Methodology 57
5.3 Training 59

5.4 Experimental Results 60
5.5 Conclusions 61

6 Attention Model for Word recognition **63**
6.1 Related Work 66
6.2 Visual attention for scene text recognition 67
6.3 Inference 70
6.3.1 The basic inference procedure 70
6.3.2 Incorporating language models 70
6.3.3 Lexicon-based inference 71
6.4 Experimental Results 71
6.4.1 Baseline performance analysis 72
6.4.2 Comparison with state of the art 73
6.4.3 Discussion 75
6.5 Conclusions 77

7 SPHOC: Attention based Spatial Histogram of Characters for text recognition **79**
7.1 Methodology 81
7.1.1 Extension of the PHOC representation 82
7.1.2 Encoder 83
Length Estimator: 84
Attention map generator: 85
Histogram Generator: 86
7.1.3 Decoder 86
The basic inference procedure 87
Incorporating language models and Lexicon 88
7.2 Experiments and Results 89
7.2.1 Training and network details 89
7.2.2 Results 90
Length estimation 90
Word Spotting for Handwritten text: 91
Word Recognition for handwritten text: 91
Lexicon-free Word recognition in Scene Text 91
7.3 Qualitative Results 92
7.3.1 Attention Visualisation 92
7.3.2 Result 93
Recognition 93

7.4 Discussion 93
7.4.1 Model Size comparison 94
7.4.2 Training data requirement 95
7.5 Conclusion 95

8 Conclusion and Future works **99**

Bibliography **103**

List of Tables

2.1 Summary of different dataset used in this thesis 11

3.1 Result of our word spotting method in comparison with state-of-the-art. (1) Basic Sliding Window. (2) Based on Integral images of attributes. (3) Bi-gram Indexing. 27
3.2 Result of our word spotting method in comparison with state-of-the-art. (1) Basic Sliding Window. (2) Based on Integral images of attributes. (3) Bi-gram Indexing. 28
3.3 Result with respect to the computation time in per query basis 28

4.1 Evaluation of Text Box proposals in comparison to basic sliding window protocol for GW dataset 43
4.2 Evaluation of QBE word spotting 46
4.3 Evaluation of QBS word spotting 46
4.4 Evaluation of n-gram Indexing for QBE and QBS 47
4.5 Comparison with state of the art methods for QBE in GW20 49
4.6 Comparison with state of the art methods for QBS in GW dataset . . . 49
4.7 Comparison of computational time per query with state-of-the-art methods. (1) All Candidates + Width (2) Filtered Candidates +widh (3) Filtered Candidates+width+indexing 49

5.1 Performance of R-PHOC for QBE word spotting in segmented word images) 60
5.2 Comparison with state of the art methods for QBE in GW20 60

6.1 Impact of the different components of our framework with respect to the baseline. We compare the baseline (LSTM with no attention model) with all the variants of the proposed method, incrementally: using only the attention model (section 6.2, integrating also the explicit language model (section 6.3.2, and contraining the inference to a lexicon (section 6.3.3). 73
6.2 Scene text recognition accuracy. “50”and “Full” denote the lexicon size used for constrained text recognition as defined in [65]. Results are divided into lexicon-based and unconstrained (lexicon-free) approaches. *DICT [25] is not lexicon-free due to incorporating ground-truth labels during training. 74

7.1 Performance of the length estimator 90
7.2 Comparison with state of the art methods for word recognition in IAM dataset 91
7.3 Comparison with state of the art methods for QBE and QBS in IAM . 92
7.4 Scene text recognition accuracy. “50”and “Full” denote the lexicon size used for constrained text recognition as defined in [65]. Results are divided into lexicon-based and unconstrained (lexicon-free) approaches. *DICT [25] is not lexicon-free due to incorporating ground-truth labels during training. 93
7.5 Comparison between different approaches to compute PHOC 95

List of Figures

2.1 Example pages from GW,IAM and BCN 10
2.2 Example word images from scene text datsets. First row contains examples from SVT dataset, second and third row contains examples from IC'03 dataset and the last row contains examples of Synt'90k dataset. 12

3.1 Caption 18
3.2 Caption 19
3.3 A typical QBS Retrieval pipeline in presence of bi-gram index 25

4.1 Proposed Framework 31
4.2 *a*) Input Image and corresponding minimal bounding box shown in red and *b*) Image after each connected component is replaced by its minimal bounding box 34
4.3 *a*) Line estimation *b*) Atomic Bounding Boxes For each line 35
4.4 Combining Atomic BBs: *LEVEL1*: Atomic Bounding Boxes *LEVEL2*: Composite Bounding boxes (candidate text boxes) taking 2 atomic BBs at a time *LEVEL3*:Composite Bounding boxes (candidate text boxes) taking 3 atomic BBs at a time 36
4.5 Regular sliding windows Vs Evaluation based on combining Atomic Bounding Boxes 38
4.6 Proposed Indexing Scheme 40
4.7 Recall and Number of Candidates for different values of threshold for GW and BCN Datasets. The value in x-axis corresponds to $1 - t$, being t the classification threshold. Thus, low value in x-axis means high value for t, that is less number of candidates 44

4.8 Performance of n-gram indexing 52
4.9 Top text boxes indexed by character n-grams 53
4.10 10 top retrieval results for different queries in the three datasets evaluated in our experiments 53

5.1 The proposed pipeline for segmentation free word spotting, First the whole document image is passed along with bounding box co-ordinates of candidate words, then the feature map corresponding to these regions are pooled by ROI pooling layer and finally passed to sigmoid activation to predict the PHOC attributes for the whole set of candidates. 56
5.2 Architecture of our proposed R-PHOC network. All the layers before ROI pooling layer are same as PHOCNET [56], to facilitate transfer learning. ROI pooling is used to pool features over a candidate word region, in segmentation free word spotting 58

6.1 A typical common subspace embedding approach. Both image and text are mapped to the common space but a point in the common space can not be traced back to the text space. 64
6.2 Overall scheme of the proposed recognition framework. Given a cropped word image, a set of spatially localized features are obtained using a CNN. Then, an LSTM decoder is combined with an attention model to generate the sequence of characters. At every time step the attention model weights the set of feature vectors to make the LSTM focus on a specific part of the image. 65
6.3 The proposed Encoder-decoder framework with attention model. . . . 67
6.4 Effect of increasing size of the training set data with (green) and without the language model (blue). 75
6.5 Some results obtained with the proposed model. 76

7.1 Holistic feature based approach (on the left) vs feature section based approach (on the right) 81
7.2 Overall Architecture: First a CNN feature map is obtained then CNN feature is enriched with length and position. From this enriched feature map n different attention is computed, producing n different representation of the image, from each of these representation a histogram of character is computed by using one fully connected layer, final PHOC is obtained by concatenation of n histogram of characters 82

7.3 Attention weights at every level of PHOC is overlaid for visualization, at every level the corresponding part is given more weight. 94
7.4 Some example results on SVT dataset 95
7.5 Comparing number of parameters: on the left based on holistic representation and on the right based on feature selection mechanism . . . 96
7.6 Inception module used in our model 97
7.7 Training data requirement vs Accuracy 97

Chapter 1

Introduction

1.1 Background

Text is the most common way of sharing information and knowledge since the first writing systems appeared. Before appearance of printing systems handwriting was the default way to preserve and disseminate knowledge. Therefore huge amount of information is stored in old manuscripts about our history. In recent times due to the rise of electronic and hand held devices the way of knowledge sharing has drastically changed. Nowadays images and videos are often used to share and spread information (e.g. billboard, advertisements etc.). Due to the natural tendency of humankind often text is prevalent in these images for specific information.

Due to recent development of images databases of handwritten historic manuscripts the demand for algorithms to make these databases accessible for browsing and indexing are in rise. Exponential increase of publicly available image databases and personal collections of pictures, this interest now also embraces text understanding on natural images.

Understanding text in historical manuscripts and natural scene images poses different challenges due to their nature. In old manuscripts challenges mainly come from different manual interventions like writing styles, ageing or bleed through the document etc., whereas in natural images challenges mainly comes from capturing devices e.g.perspective distortion, blur, use of decorative fonts etc.

Though these two areas of research poses separate set of problems as both deal with understanding of text, thus also share some common properties (like sequential nature etc.). Thus in this thesis we explore various techniques for text understanding in both natural scene images and handwritten manuscripts.

As designing an algorithm to understand text like human is difficult, researchers approached this problem in different ways resulting in different set of sub-problems. In the following section we will discuss different taxonomies of text understanding before specifying the goal and contribution of the thesis.

1.1.1 Taxonomies

Early works on understanding text in mainly concerned about printed documents. Optical Character Recognition (OCR) is a technology is developed to recognize characters or words in images of printed documents. These technologies are not directly applicable to handwritten or natural scene images due to different challenges like the diversity of the handwriting style, the presence of noise and distortions in historical manuscripts or perspective distortion and blur often present in natural images.

To overcome this image or text based search in documents to find the relevant information is proposed as an alternative. Such a system is called word spotting systems as it tries to spot occurrences of a word inside a document page or database. Depending on the input (query) modality the system is called query by example (QBE) or query by string (QBS). The goal of word spotting can be defined as identifying and retrieving all those regions in a dataset of document images that contain an instance of a query word. In a multi-writer collections, where handwriting can differ significantly from document to document, this task can be quite challenging. In the literature, word spotting appears under two distinct trends wherein the fundamental difference concerns the search space which could be either a set of segmented word images (segmentation-based approaches) or the complete document image (segmentation-free approaches). In this work, we address the query by example word spotting problem in a segmentation-free multi-writer scenario.

In case of document image databases where a lot of text is present and not every text is recognizable (or required) a word spotting system can be very useful to retrieve interesting information, however in case of natural images where occurrence of text is less than that of handwriting, recognizing the text is often more useful. However as discussed state of the art OCR technologies are not adequate for this task. Thus

text in the wild or text recognition in natural scene emerged as a new research area. Sometimes a predefined lexicon is used to help the recognizer, depending on that the scene text recognition can be classified as lexicon-based and lexicon-free recognition.

1.2 Related Work

As described in the previous section, research efforts in text understanding is divided into different sub-problems. In the following sections we discuss the related works and summarize our contributions with respect to different sub-problems. We discuss most related works corresponding to every sub-problem here, however in some of the chapters, related works specific to corresponding chapter is also discussed.

1.2.1 Word spotting in handwritten manuscripts

Inspired by traditional OCR methods initial approaches for word spotting were based on segmenting the document into a sequence of characters or words and then trying to recognize the query word using some trained character or word models [9, 42]. The main drawback of these systems lied in the need of performing an expensive and error prone segmentation step to select candidate windows (words) or characters.

Some works approached the problem from a different perspective in order to avoid this error prone segmentation step, based on the extraction of local keypoints that are encoded using local descriptors, such as gradients [35, 14] or the Heat Kernel Signature [71]. Then, word spotting can be performed by locating those zones in the document images with similar interest points and, in some cases with the same spatial configuration, as the query model [14]. However they normally use a costly distance computation, which is not scalable to large datasets.

In most recent methods, word images are represented using a fixed-length description, which allows for an efficient and simple comparison of regions using cosine or Euclidean distance. For instance, Rusiñol *et al.* [50] propose a descriptor based on the well known bag of visual words (BoW) framework [11], in combination with a sliding window over the whole image to retrieve relevant regions. In addition, Latent Semantic Indexing (LSI) is used to learn a latent space where the distance between word representations is more meaningful than in the original bag of words space. Rothacker *et al.* in [49], also make use of the BoW representation to feed a HMM avoiding seg-

mentation using a patch-based framework. In [2] Almazán *et al.* proposed to use a HOG based word representation in combination with an exemplar-SVM framework to learn a better representation of the query from a single example. Compression of the descriptors by means of product quantization permits a very efficient computation over a large dataset in combination with a sliding window-based search.

All these approaches are valid only in a QBE scenario. In another work Almazán *et al.* [4] proposed to learn a fixed length word representation based on character attributes to perform both QBE and QBS using the same framework. The attribute representation encodes the presence/absence of characters in different spatial positions in the word image through a Pyramidal Histogram of Characters (PHOC). This representation was originally learned using Fisher Vector representation of the images. Once word images are represented in this attribute space, spotting is reduced to a Nearest Neighbour problem. This word attribute representation proved to be very discriminative and achieves a very high accuracy when applied to already segmented words.

Therefore, more recently some adaptations of this representation has been proposed [56, 44] that make use of CNNs learn the attribute space using more powerful features, achieving state-of-the-art results in segmentation-based word spotting. Despite being successful in achieving excellent result in segmentation based word spotting. They also have few limitations first, the proposed approach is for segmentation based and their extension to segmentation-free is not straight forward. Secondly most of these works use a fixed length embedding and thus can not be used directly for lexicon-free word recognition.

In this thesis, we propose solutions for both of these problems, first in chapter 3 we proposed a solution to extend the use of PHOC in segmentation-free word spotting. We propose to use integral images of pre-computed PHOC attributes for document pages. Further speed-up is being achieved by using character bi-gram based indexing. In chapter 4, we extend this work and proposed a method to produce text-specific bounding box proposals.

We dealt with lexicon-free word recognition problem in chapter 7, where we proposed an attention based approach to compute PHOC using convolutional features. Lexicon-free word recognition is achieved by decoding the PHOC embedding.

1.2.2 Text Recognition in handwritten manuscripts

Proposed methods in lexicon-free word recognition in handwritten documents is dominated by the use of recurrent neural network or LSTM. The most popular approach is using some feature extractor and then using LSTM with CTC loss to train end-to-end. Recently [8, 44] convolutional neural net is commonly used as feature extractor. Slightly different approach has been taken by [8], when they employed an HMM on top of convolutional feature extractor instead of CTC. Pham *et al.* [43] experimented with dropouts to improve generalization ability of recurrent neural networks.

Multi-dimensional LSTM [43] is also became very popular in recent times as they allow recurrence over both axes of an input image achieving state-of-the-art result and wining competitions in handwriting recognition [51].

Though most of these works used a set of features of variable length in [44] authors proposed to learn fixed length PHOC embedding directly from word images by using a CNN. Word recognition can be achieved by using a lexicon using nearest neighbour search, thus they have to rely on a predefined lexicon to achieve word recognition.

In chapter 7, we also learn PHOC embedding using CNN feature, but rather than learning the PHOC embedding from entire word image, we use an attention based feature selection mechanism to generate different histograms of characters per pyramidal region. In addition to this we show that how PHOC embedding can be decoded back to string, if sufficient levels are present.

1.2.3 Text recognition in scene text

Traditionally, scene text recognition systems were based on character recognizers applied in a sequential way by localizing characters using a sliding window [27, 39, 65] and then grouping responses by arranging the character windows from left to right as words. A variety of techniques have been used to classify character bounding boxes, including random ferns [65], integer programming [54] and Convolutional Neural Networks (CNNs) [27, 7]. These methods often use lexical constrains imposed by a fixed lexicon or a language model [7] while grouping the character hypotheses into words.

In contrast to sequential character recognizer models, holistic fixed-length representations were proposed in [4, 18, 46]. These works advocate the use of a joint embedding space between images and words. With the success of deep-CNN features in computer vision, holistic word representations have already been proposed using

CNNs. The first such attempt was made by Jaderberg *et al.* in [27], where a sliding window over CNN features is used for robust scene text recognition using a fixed lexicon. Later, the same authors also proposed a fixed-length representation [26] using convolutional features trained on a synthetic dataset of 9 million images [25]. However, although most of these works achieve very high accuracy, they only focus on lexicon-based recognition and cannot recognize out-of-vocabulary words.

Unconstrained text recognition requires modeling the underlying character-level language model. Jaderberg *et al.* in [23] proposed to use two separate CNNs, one modeling character unigram sequences and another n-gram language statistics. They additionally use a Conditional Random Field to model the interdependence of characters (n-grams). However, this significantly increases computational complexity.

Lexicon-free recognition systems generally relies on sequential nature of text and use some variety of recurrent neural network.

This is achieved by first encoding a text image into a sequence of features with deep neural network, then directly generating character sequence with sequence recognition techniques. In chapter 6, we proposed to use an visual attention model to extend the use of sequence-to-sequence model for scene text recognition. In contrast to other state-of-the art works in this area, we proposed to use a feature map from a lower convolutional layer to leverage spatial features. Attention is then used to generate a feature map corresponding to every character at every step.

He *et al.* [21] and Shi *et al.* [52] proposed to capture visual feature representation by using CNN or RNN, then the CTC [19] loss was combined with the neural network outputs for calculating the conditional probability between the predicted and the target sequences. They also developed an attention-based spatial transformer network (STN) for rectifying text distortion, which helps recognize curved scene texts [53].

In [10], the authors proposed to correct the focus of attention caused by attention drift as observed in attention training of speech recognition .In a recent work authors proposed to use edit probability to use a better language model.

However all of the works used recurrent network layer for lexicon free word recognition except the work by [70], where they used a sliding window over convolutional features. Though they get good results by using convolutional features but they need to use sliding window over with three different scales. In chapter 7, we also propose to use a convolutional model, however unlike other models we use an attention based feature selection mechanism to generate a sequence of features, which can be used both as a fixed length feature or a set of variable length features. This enables us to do

both word spotting and lexicon-free word spotting using the same framework.

1.2.4 Contribution and organization of the thesis

The thesis is divided in to two parts the first part (consisting of chapters 3,4 and 5) is devoted to word spotting in handwritten documents and the second part (consisting of chapters 6 and 7) is mainly concerning recognition of natural scene text). Additionally we have one chapter (Chapter 2)in beginning and one in the last (chapter 8), first one describing common experimental protocols used throughout the thesis also different datsets used in experiments and the last cone for conclusion and future directions of research. Every chapter describes, discuss and makes comparison to the state of the art methods which are most relevant to the contribution of the chapter. In the following, we elaborate the contributions of individual chapters in brief.

- In chapter 3, we propose a sliding window based segmentation free word spotting framework, which make use of Pyramidal Histogram of Character (PHOC) originally proposed [4] for segmentation based approaches. We extended the use of PHOC from segmentation based approaches to segmentation free by using an integral image framework. We also developed a bi-gram based indexing mechanism to faster evaluation of query, for this purpose we proposed to learn an attribute space based on character bi-grams. Finally we evaluated our proposals and compared with state of the art.

- In chapter 4, a bounding box proposal algorithm is proposed for text. Inspired by use of class agnostic bounding box proposal algorithms like selective search in case of object detection and other computer vision task we propose a word agnostic bounding box proposal scheme for word spotting. We also extend our indexing mechanism of chapter 3 to index these proposed bounding box proposals. Furthermore we extend the indexing mechanism to estimate presence of character n-gram instead of bi-grams.

- In chapter 5, we use a deep learning framework for segmentation free word spotting. Particularly we propose a framework which works similar to RCNN for segmentation free word spotting. Our framework takes document image and text box proposals from chapter 4 as input and produces PHOC descriptor as output for every proposal.

- In chapter 6, a lexicon-free word recognition approach is described based on visual attention. We compare the approach with fixed length approaches and discuss benefits and drawbacks of a fixed length descriptor to that of a sequence based representation.

- In chapter 7, We described a novel framework based on convolutional neural network and attention which can deal with both word spotting and lexicon-free word recognition. Thus we can generate both sequence based representation and fixed length representation.

Chapter 2

Protocols and Dataset

Quantitative analysis is very important to understand the efficacy of new methods for any problem of empirical nature. To be able to make valid comparison among methods some common protocols and datasets are needed. In this chapter we will introduce the datasets and protocols used throughout this thesis.

2.1 Datasets

The goal of this thesis is to understand text found in diverse input domains, depending on the nature of input domain the datasets can be roughly classified into two categories 1) Handwritten text 2) Scene text in uncontrolled environment.

Below we describe the datasets used for experiments in both handwriting and scene text.

2.1.1 Handwritten documents dataset

The George Washington (GW) dataset [45] has been used by most researchers in handwriting recognition and spotting and has become one of the most important datasets to benchmark the results. This dataset contains 4860 words annotated at word level. The dataset comprises 20 handwritten letters written by George Washington and

his associates in the 18th century. The writing styles present only small variations and it can be considered as a single-writer dataset. The GW dataset presents many of the challenges of old manuscripts, although it does not contain much variability in the writing style as it contains writing of a single individual.

IAM offline dataset [61] is a large dataset comprised of 1539 pages of modern handwritten English text written by 657 different writers. The document images are annotated at word and line level and contain the transcriptions of more than 13000 lines and 115000 words. This is the largest dataset available for academic research as the number of writers is 657, this gives the opportunity to evaluate the methods in multi-writer setting.

Barcelona Centuries of marriage dataset (BCN)[47] contains 50 handwritten pages from a collection of marriage licences written in 1617 from Barcelona cathedral. This database contains 13000 words and have a bigger lexicon than GW. As the manuscripts are ancient the dataset also include a considerable amount of noise caused by ageing of manuscripts. Figure 2.1 shows one example page from each of the datasets.

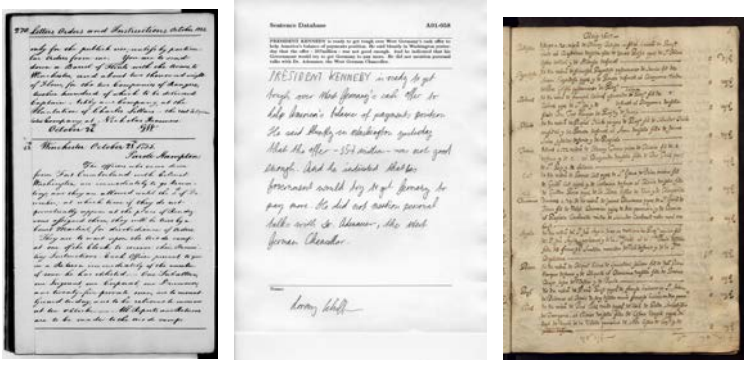


Figure 2.1: Example pages from GW,IAM and BCN

Table 2.1 summarizes the details of each dataset used in this thesis.

Name	single/multiple	# of Authors	# of Pages	# of words	# of unique words
GW	single	1	20	4860	1124
BCN	single	1/2	40	-	-
IAM	multiple	657	1539	115000	13752

Table 2.1: Summary of different dataset used in this thesis

2.1.2 Scene text datsets

Street View Text (SVT) dataset: this dataset contains 647 cropped word images downloaded from Google Street View. Many of these images are of low resolution and challenges like blurr and perspective distortion is present in this dataset.

ICDAR'03 text dataset: this dataset contains 251 full images [55] with bounding boxes around text instances. Every image is associated with a 50-word lexicon defined by Wang *et al.* [64]. A full lexicon that combines all lexicon words is also provided. we discard the images containing non-alphanumeric characters or have less than three char- acters similar to [65] for fair comparison. The resulting dataset contains 867 cropped images.
MSCOCO [62] text Dataset: This is the biggest published dataset in scene text. This dataset is challenging as none of the images are captured specifically with text recognition in mind.
Synth90k text dataset: this dataset is published by Jaderberg *et al.* in [25]. This is a huge dataset comprised of synthetically generated cropped word images. A dictionary cotaining 90K words is selected. Word images are rendered synthetically to generate these images. Background, fonts etc. are varied to generate word images with variation. We used this dataset only for training. It contains 9 million cropped word images, we have used only official training partition for training our models.

2.2 Protocol

The main experiments carried out for quantitative analysis in this thesis are word spotting and word recognition. In this section we describe the standard protocols used for these experiments throughout the thesis.

Figure 2.2: Example word images from scene text datsets. First row contains examples from SVT dataset, second and third row contains examples from IC'03 dataset and the last row contains examples of Synt'90k dataset.

2.2.1 Word Spotting

We use mean average precision and recall as standard evaluation measures. Mean average precision is used to evaluate the accuracy of retrieval. This is consistent with most of the works in literature. However some works adopted different evaluation measures and/or protocols. A candidate region from the document will be considered as relevant when it overlaps at least 50% of a word bounding box in the ground-truth and the transcription matches with that of the query word. We have used all words in the ground-truth as queries. However in BCN there are some groundtruth words where the labels contain non alphanumeric characters. We removed those words from the list of queries. In the case of the GW and BCN datasets we use 4-fold cross validation using two folds for training, one for validation and one for test. Results are the average of the 4 folds used for testing. For IAM we use the official partition of training and test. The retrieved results are sorted according to the adopted similarity measures and average precision is calculated. If the query itself is retrieved in the list then it is considered as a positive result following the same protocol as many of the published works in this area *e.g.* [50].

2.2.2 Recognition

The word recognition systems sometimes use a fixed lexicon and chose one of the word in lexicon as it's output, such a framework is referred as lexicon-based recognition framework as opposed to lexicon-free framework where no a-priori lexicon is given. The Standard evaluation protocol adopted in most of the previous works on text recognition in scene images [23, 32, 65] is word level accuracy in percentage for both lexicon-free and lexicon-based framework. For lexicon-based recognition, we used the same set of 50 words for all images in for SVT and ICDAR'03 dataset, as proposed by Wang *et al.* [65].

In case of handwriting a different protocol is commonly used based on edit distance i.e CER (character error rate) and WER (word error rate), we followed this trend for sake of valid comparison.

Finally we should comment that in addition to all these protocols, which are common to most experiments, specific experiments are also carried out to understand and analyze different contributions in different chapters of the thesis. These protocols are outlined whenever they are employed.

Chapter 3

Representing documents with attributes

In this chapter, we describe the use of fixed length attribute based representation for query by example and query by string word spotting for segmentation free word spotting. Almazan et.al. [2] proposed to learn PHOC (Pyramidal Histogram of Character) a fixed length representation based on character attributes to perform both QBE and QBS for segmented based word spotting using the same framework. The representation is learned using powerful fisher vector representation and very successfully applied for segmentation based word spotting. In this chapter, we explore different ways of using this representation for segmentation free word spotting. We explore different ways of using sliding window based framework for word spotting using this attribute based representation. We make use of the additive nature of fisher vector to store the attribute representation in a integral image format, making faster retrieval possible in a segmentation free setup. Further we have used an indexing scheme based on character bi-grams for speed up in retrieval.

In recent works PHOC embedding has been trained using deep learning based approaches but in this chapter we will limit ourselves in exploration of non-deep learning approaches. Deep learning based approaches for PHOC embedding will be dealt in this thesis later.

The rest of the chapter is organised as follows: in section 3.1 we discuss related

works most relevant to this chapter, in section 3.2 we describe the computation of the Pyramidal Histogram of Character (PHOC). In section 3.3 we explain different ways of using PHOC based attribute representation for a segmentation free word spotting.

In section 3.4, we discuss results in different experimental setups including basic sliding window based protocol, Integral image and bi-gram indexing. Finally we conclude the chapter with conclusion section outlining possible extensions and improvements.

3.1 Related Works

Word spotting methods can be categorized in to two broad sub-categories from the feature representation point of view. A word image can be either represented by a sequence of features(variable length) or a holistic representation (fixed length).

The sequence based methods uses a segmentation step before representing each segment by one feature vector followed by a sequence based comparison. The dependence on the segmentation step, which can be very sensible to handwriting distortions can be disadvantageous as its accumulates the error to the next step. Additionally sequence-based comparison methods are often computationally more expensive then fixed length representation

This motivates the researcher to explore fixed length representation without precise segmentation step.

Initial efforts [14, 35, 71] towards this goal are based on the extraction of local keypoints based descriptors. Local keypoints are aggregated to give a fixed length representation in [50] using well-known bag of visual words (BoW) framework [11]. Comparison between query and candidate word images is easy and faster with this approaches. Though all these works performs well in case of word spotting where the query is also provided as an image (i.e for QBE) but for Query By String where query is provided in the form of text none of these methods can be used directly as their is no way of comparing an image to text.

Almazan et.al. [2] proposed to learn common space fixed length representation based on character attributes to perform both QBE and QBS using the same framework. The attribute representation encodes the presence/absence of characters in different spatial positions in the word image through a Pyramidal Histogram of Characters (PHOC). This representation is learned using powerful Fisher Vector representa-

tion of the images. Once word images are represented in this attribute space, spotting is reduced to a Nearest Neighbour problem. This word attribute representation proves to be very discriminative and achieved new state of the art for segmentation based word spotting (QBE and QBS) and recognition.

3.2 Pyramidal Histogram of Character

The main idea of Pyramidal Histogram of Character (PHOC) based attribute representation as proposed by Almazán *et al.* [3] is to learn a common low dimensional representation for word images and text strings, that permits to address retrieval as a simple nearest neighbor problem.

3.2.1 Word Representation

A common representation can be learned by jointly embedding the text strings and corresponding word images in to a vector representation. To learn the embedding instead of original text it's vectorial representation is used as source. Fixed length embedding can be obtained by computing histogram from text strings.

A histogram of 36 dimensions can represent any alphanumeric text in English language, where each dimension represents whether the text string contains a particular character or not. Thus every dimension of histogram corresponds to one character in English alphabet and each dimension can be thought of as one attribute of a given text string. As this global histogram disregard the locality or order thus two completely different words with same characters will result the same embedding in this scheme. In computer vision tasks this is dealt by adding more local features by using spatial pyramid. Similarly for text strings a pyramidal version of histogram embedding can be computed. This can be achieved by computing histograms from different regions of the word and concatenating the embedding which will encode more local information. The word is segmented in regions in a pyramidal fashion i.e at level 2 the word is divided in two parts and two histograms are computed similarly at level 3, 3 histograms are computed by dividing the word into 3 parts and so on. In practice levels 2, 3, 4, and 5 are used leading to a histogram of $(2 + 3 + 4 + 5) \times 36 = 504$ dimensions. almazán *et al.* added extra attributes for 50 most common English bigrams at level 2, leading to 100 extra dimensions for a total of 604 dimensions.

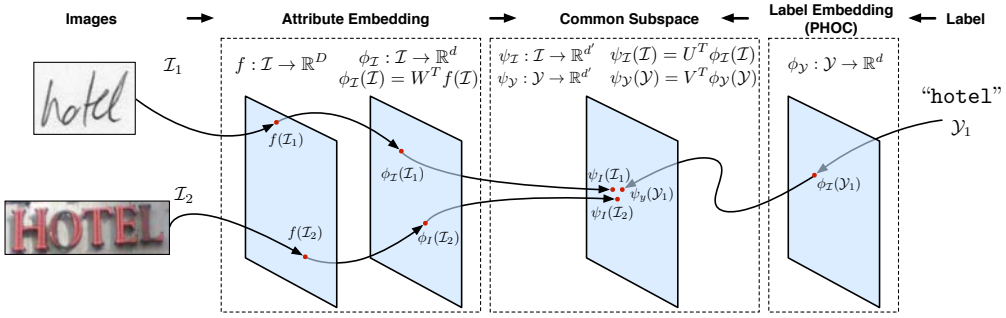


Figure 3.1: Caption

3.2.2 Attribute-based word image representation

Th PHOC embedding described above is used as a source for learning character attributes from word images. Each word image is projected into a $d-$dimensional space (same dimension as the PHOC representation) where each dimension is an attribute encoding the probability of appearance of a given character in a particular region of the image, using the same pyramidal decomposition as in the PHOC representation. Each attribute is independently learned using an SVM classifier on a Fisher Vector description of the word image, enriched with the x and y coordinates and the scale of the SIFT descriptor.

More formally, given a training image I its Fisher Vector representation [41] $f(I)$ is computed, where $f(I)$ is a function of the form $f : I \to R^D$, being D the dimension of the Fisher Vector representation. Now, to project Fisher Vector representations into the PHOC attribute space, an embedding function ϕ_I of the form $\phi_I : I \to R^d$ is learned such that

$$\phi_I(I) = \mathbb{W}^{\mathbb{T}} f(I) \tag{3.1}$$

where W is a matrix with an SVM-based classifier for each attribute learned using the PHOC labels of all the training words.

In a query by example setting both the query and word images are described with this attribute representation, which is very discriminative as each attribute is giving the probability of a certain character in a specific position within the word. Retrieval

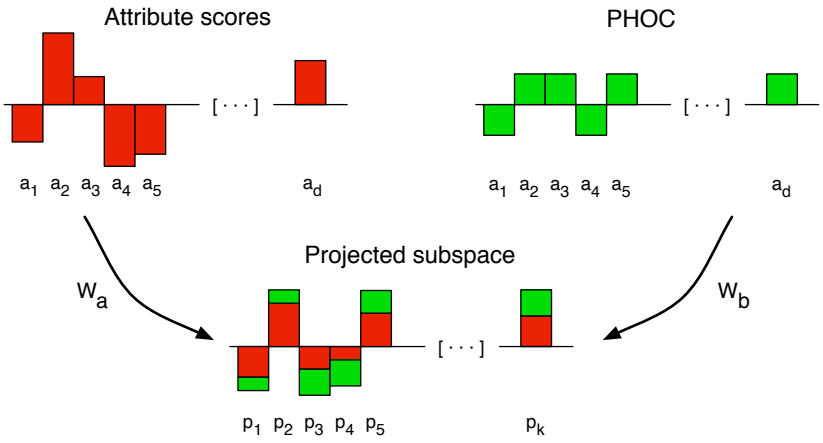


Figure 3.2: Caption

simply translates into finding the word candidates whose attribute representation is close to that of the query image.

To make direct comparison between binary PHOCs and real valued attribute representations feasible Almazán *et al.* in [3], proposed an additional step to learn a common subspace between strings and images. A final calibration step is added (3.2, using Canonical Correlation Analysis, that aims at maximizing the correlation among both representations. This final calibration and dimensionality reduction step can be represented with an additional embedding function ψ represented as $\psi_I : I \rightarrow R^{d'}$ and can be given as:

$$\psi_I(I) = U^T \phi_I(I) \tag{3.2}$$

being U the transformation matrix obtained with Canonical Correlation Analysis.

3.3 Segmentation Free Word Spotting with Attribute representation

In this section we describe the different ways to utilize the attribute representation described in above section for segmentation free word spotting. First in section 3.3.1, we

describe a basic sliding window approach as baseline. In section 3.3.2, we improve over this baseline method by using an integral image of word attributes over the document pages and finally in section 3.3.3, we show how an effective indexing scheme can be combined with this to achieve even faster retrieval.

3.3.1 Sliding widow - baseline

Given a discriminative word representation the simplest way to extend it for segmentation free word spotting is to use a sliding window protocol to generate candidate word hypothesis over the documents. In sliding window protocol a predefined window slides over the documents, at every slide the area of the image which comes under the window is inspected for possible query word. In a sliding window there are two free parameters one is the fixed size of window and the amount of displacement at each slide.

In our baseline system we use a sliding window based protocol, Given a query image we use the size (height and width) of the query as sliding window size. There exists a trade-off between number of scales and number of candidate windows. More scales can give better result with the expense of computing attributes for too many candidates on the other hand this can also increase the number of false positives. We experimented with different settings to analyze the impact of different scales in final retrieval result.

Every candidate window is represented in attribute space. To compute the attribute representation first SIFT descriptor is computed in dense scales, which is then encoded using Fisher Vector. These Fisher Vector is used to compute the attribute descriptor, to compute attribute descriptor we use the embedding matrices learned from segmented words as described in 3.2.2.

Once every candidate window and the query is represented in attribute space the retrieval is reduced to nearest neighbour problem. We compute the distance between the query and the candidates sorting them according to the distance to the query gives the final retrieval results. In this setup we have used the size of the query to define the sliding window which essentially makes the retrieval process suitable only for Query-By-Example setup. In our experiments we have extended this to Query-By-String, where the size of the window is defined as the mean width and height of the training samples for the same string.

3.3.2 Integral Image based sliding window

In sliding window protocol the query is searched exhaustively over the document image, thus it is guaranteed to return relevant regions if it exists. However this also make the number of hypothesis to inspect very large. Moreover representing these candidates in attribute space needs costly computations of SIFT descriptor and Fisher Vector making it prohibitive at run time.

In addition to that, computing attributes for every window given by a sliding window protocol needs computation of SIFT descriptors and the Fisher Vector for a large number of overlapping windows redundantly over the same image. To alleviate these problems we propose to pre-compute off-line the attribute representation for every pixel of the image and store it in an efficient integral image [63] that can be used to compute very fast the representation of any candidate window at query time.

To describe the computation of the integral image of the attribute representation, let us denote the document images of the dataset as $I^k, k = 1...n$ where n is the total number of images. For a given image I^k, we first compute the set of dense SIFT descriptors $d^k_{i,j}$ at every location (i, j). Then, we can define the embedding function into the attribute space ϕ_I for every pixel location as:

$$\phi_I(i, j) = \mathbb{W}^{\mathbb{T}} f'(d_{i,j}) \tag{3.3}$$

where $f'(d_{i,j})$ is the Point Wise Fisher Vector representation (PFV) for the (i, j) pixel of image I^k and W is a matrix encoding the attribute classifiers as in previous section. Note here that we extend the Fisher Vector embedding method [41] to derive Pointwise Fisher Vector (PFV) encoding. PFV of a local feature (corresponding to a pixel) can be understood as the contribution of that local feature to the whole Fisher Vector of the image. Before normalization the fisher vector can be understood as aggregation (summation) of first order and second order moments of each local features, thus we can compute PFV for each local descriptor (pixel). Finally this attribute representation for every pixel is projected to the lower dimensional subspace obtained through Canonical Correlation Analysis using the same transformation matrix U introduced in previous section:

$$\psi_I(i, j) = \mathbb{U}^{\mathbb{T}} \phi_I(i, j) \tag{3.4}$$

Once we have the final attribute representation for every pixel, it can be easily aggregated into an integral image $\Psi_{i,j}$:

$$\Psi_{i,j} = \sum_{i'<=i, j'<=j} \psi_{i,j} \tag{3.5}$$

So, for every pixel we can store it's contribution to attribute space using integral image format. Though the dimension of attribute space is low (around 100) but to store for every pixel for the whole document set will need lot of memory. A typical document image is of size 1000×2000, thus to store attribute representation for one document will need $1000 \times 2000 \times 100$ floating point values which is approximately 0.8 Gb of data. We reduce this memory requirements by arranging the image into $N \times N$ dimensional blocks and instead of computing and storing Fisher Vector representation for every pixel, we only compute one attribute representation for each block. This will reduce the memory requirements by a factor of N^2, where N is the size of rectangular block to compute one attribute representation. In our experiments we used block size as 16.

Finally, given a query image and the integral image representation, we first compute the attribute representation for the query and compute the attribute representation for all the candidate windows given by sliding window protocol. Given a window $w = (X_1, Y_1, X_2, Y_2)$, where (X_1, Y_1) are the co-ordinates of the top left corner and (X_2, Y_2) are the co-ordinates of the bottom right corner of the w, we can compute the attribute representation Ψ_w in a very simple way with just 4 vector additions as:

$$\Psi_w = \Psi(X_2, Y_2) + \Psi(X_1, Y_1) - \Psi(X_1, Y_2) - \Psi(X_2, Y_1) \tag{3.6}$$

Similar to the above section once candidate windows and the query is represented in attribute space retrieval is done by computing the similarity between the query and the candidates. For similarity measure cosine similarity is used. We first l2 normalize the representation and take dot product to compute similarity between query and candidate windows.

3.3.3 Indexing based on character bi-gram presence

In above approaches we rely on sliding window protocol to generate candidate word windows, which is further compared to queries. Offline computation of attributes and using integral image reduce computation at run time but still we rely on exhaustive search using a sliding window protocol. We observe that for a given query only a few areas of the document image is important, following this observation we propose to create an inverted index based on presence of character bi-grams, the hypothesis is that bi-grams can be discriminative word features that can be used as the basis for localizing areas of the document where the word is likely to appear. In an inverted index table we store the co-ordinates of regions of the documents corresponding to each bi-grams.

In run-time instead of whole documents only these regions are searched. We generate regions to be indexed by a very naive segmentation algorithm based on connected component analysis. We propose an attribute space only based on bi-grams for indexing this is similar to PHOC space proposed in section 3.2 but instead of characters we use bi-grams. At query time bi-grams present in the query word are identified and then the regions corresponding to the bi-grams are searched using a sliding window protocol for presence of the query word. In the following subsections we describe in detail each of the components of our approach: the proposed attribute representation for bi-gram based indexing, followed by use of this attribute space to generate index.

PHOB representation

To estimate the presence of these bi-grams in word images, we devised a representation based on occurrence of bi-grams called PHOB ((Pyramidal Histogram of Bigrams) henceforth in this thesis. For English alphabet number of distinct bi-grams are 26×26, which is 676, to use all of them as index will lead to a huge index. It can be observed that many of these bi-grams are rarely found in words. A nice analysis of the occurrences of character n-grams has been done in [28]

where they studied a large corpus of 183 million words. Following this study we consider 150 most popular bi-grams from this study which covers 99.21% of the total corpus. To include numeric fields the digits from 0-9 are also included in this representation. We use these bi-grams as attributes in place of characters for text representation similar to PHOC representation as described in section 3.2. Thus at the end this particular representation is obtained using 150 bi-grams at level 2 of decomposition and using 10 digits at levels 2 and 3, thus resulting in a 350 dimensional representation. Using the strategy described above (see section 3.2.2), a similar embedding function is learned to project the Fisher Vector representation of word images into the PHOB attribute space.

Index generation

The goal of the indexing scheme is to create an ordered list of regions per bi-gram over the entire document database, so that at query time only regions relevant to bi-grams in the query word have to be searched. This index file is similar to inverted index files used frequently in text retrieval.

To generate a list of regions for each bi-gram, the document is first segmented into regions. Please note that the goal of segmentation is only to identify regions of plausible occurrence of bi-grams in documents not to find an exact and accurate word segmentation.

Segmentation

For segmentation, the document image is first binarized by setting the threshold as 75% of the mean intensity of the image. Then, each document in the database is represented as a set of connected components. Connected components for which the minimum bounding boxes around the connected components are overlapping with each other are merged into a single region. However, in English handwriting some descendant of a line can sometimes overlap with ascendants of the line below. Thus merging connected components sometimes can merge components from two different lines. To avoid this a minimal bounding box for each connected component is found(by greedily finding the smallest region that contains 90% of the density of the binarized image) then this minimal bounding box is used instead of actual bounding box to find overlapping connected components. Connected components which are very close along the width of the document page are also merged together to form a single region. However this notion of closeness can vary from one document to another. To overcome this we use different thresholds thus yielding one set of regions per threshold after merging (see fig.). At the end we merge all of these regions to obtain the list of regions which are used for indexing.

Indexing by bi-grams

To generate index over the document database represented by the segmented regions, bi-grams are considered as strings and represented by the PHOB representation introduced in section 3.2.2. Each segmented region is also represented by the corresponding PHOB based attribute representation using the Fisher Vector of the region. Both representations are then converted to a low dimensional common subspace using canonical correlation analysis (CCA). Similarity between bi-grams and segmented regions can be computed as a dot product between their corresponding representations in this space. For each bi-gram, segmented regions are sorted in order of decreasing similarity and stored as inverted index files.

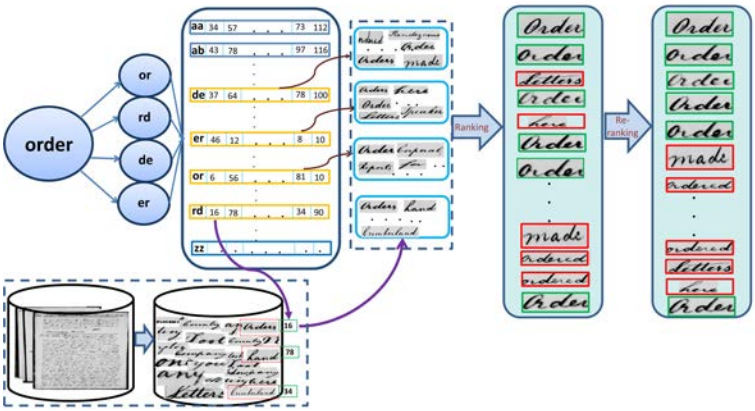


Figure 3.3: A typical QBS Retrieval pipeline in presence of bi-gram index

Retrieval

Given the indexing structure to retrieval is performed in two steps. A typical example in case of QBS is shown in figure 3.3. First a set of potential regions is identified. In second step these potential regions are searched exhaustively for retrieval. In the following, we first describe the identification of potential regions to further investigated for retrieval for both QBE and QBS and then we describe how they are further investigated for final retrieval.

QBS: Given a text query, first all distinct bi-grams of the text string are found. Then, for each character bi-gram, the inverted index is searched and the top n regions for each one are further considered for retrieval. Thus, for a query having k distinct bi-grams, $k \times n$ regions are searched. However many regions can be indexed by more than one bi-gram thus making the number of distinct regions to search much less for most cases.

QBE: Given a query image, first its PHOB attribute representation is found, and then we compute the dot product with the PHOB representation of each of the bi-grams in the index. Finally. we take the top k bi-grams based on this distance. Once bi-grams relevant to the query are identified, a similar procedure to that described for QBS can be employed, i.e the inverted indexing structure is

accessed and the top n regions for each one are further considered for retrieval.

Once potential regions are identified, the sliding window search is localized only to potential regions.

We apply the sliding window over the region as returned by the index file if the region is big enough to accommodate the query word as calculated by the mean width of all the same training words. If the region is small then we merge it with the regions on the left and on the right to make a bigger region where the sliding window can be employed. Application of sliding window is necessary as in segmentation step above we do not aim for a perfect word segmentation rather we want a rough pre-segmentation which will allow to localize the sliding window search using an indexing scheme. Using the integral image, the attribute-based representation of each candidate window explored by the sliding window can be easily obtained and compared through the cosine similarity with the PHOC representation of the query string. Finally, the candidate window list is ranked in order of decreasing similarity and non maximal suppression is performed to obtain the final relevance list.

3.4 Experimental Results

To evaluate our method and compare with other state of the art methods we use the common protocols and datasets (namely GW, IAM and BCN) as described in chapter 2.

In section 3.3 three different retrieval setups has been described, in this section we will present the results according to every retrieval setup namely (basic sliding window, integral image based and retrieval using bi-gram indexing) for both QBE and QBS.

3.4.1 QBE word spotting

In table 3.1 result of QBE word spotting is presented. It can be observed that our basic sliding window setup outperform almost every other methods in GW dataset. Almazán *et al.* while doing re-ranking with more powerful feature achieves slightly better results in terms mean average precision. It can also be observed that retrieval with indexing helps to achieve not only speed but also in precision. This gain is due

to less false positives in retrieved regions.

Table 3.1: Result of our word spotting method in comparison with state-of-the-art. (1) Basic Sliding Window. (2) Based on Integral images of attributes. (3) Bi-gram Indexing.

	GW	BCN	IAM
Almazán *et al.* [2]	51.88	84.34	-
Almazán *et al.* [2] (with RR)	57.46	84.51	-
Russiñol *et al.* [50]	30.42	42.83	-
Kovalchuk *et al.* [29]	50.1	90.7	-
Proposed (1)	**56.27**	**61.45**	**35.68**
Proposed (2)	**54.7**	**60.45**	**34.08**
Proposed (3)	**59.87**	**63.85**	**39.94**

3.4.2 QBS Word Spottig

Table 3.2 shows the result of QBS word spotting in comparison to state of the art. However comparison in this case is not straight forward as different protocols are adopted by different authors while reporting results, thus we have also provided the protocol against every result. It can be observed that only our method uses every unique word as queries. Though no conclusive inference can be drawn from these data but it can be said that our method provides competitive result in case of QBS word spotting.

3.4.3 Time complexity Analysis

We also show the average computational time to evaluate each query in Table 3.3. It can be observed that the proposed method is quite fast to be used in a real time environment. It can also be observed that use of integral image is crucial to make attribute embedding based metod feasible to use in segmentation free spotting. Further spped up is achieved by using indexing. In comparison with Almazán *et al.* [2] it is marginally slow while achieving a higher accuracy.

Table 3.2: Result of our word spotting method in comparison with state-of-the-art. (1) Basic Sliding Window. (2) Based on Integral images of attributes. (3) Bi-gram Indexing.

	segmentation	Queries	GW	BCN	IAM
Liang et al. [36]	Word-level	38 queries	67% at rank 10		
Fischer et al. [12]	Line-level	In-vocabulary words	62.08	-	47.75
Frinken et al. [14]	Line-level	In-vocabulary words	71	-	76
Proposed (1)	**-**	**All queries**	**59.62**		**39.47**
Proposed (2)	**-**	**All queries**	**59.87**		**38.78**
Proposed (3)	**-**	**All queries**	**64.7**		**39.57**

Table 3.3: Result with respect to the computation time in per query basis

	GW	BCN	IAM
Almazán *et al.* [2]	1.04s	-	0.83s
Kovalchuk *et al.* [29]	0.033	-	0.009
Proposed	3.45s	-	2.87s
Proposed with Integral Image	-	-	-
Proposed with Indexing	-	-	-

3.5 Conclusion

In this chapter a segmentation-free approach to word spotting in document images is described. We have shown an efficient way to represent PHOC based word attributes in an integral image format, which can be computed offline and used efficiently in query time. The results of our method shows significant improvements over the current state-of-the-art. In addition we are able to apply our method to the multi-write IAM dataset where we are not aware of other published results in the context of segmentation-free word spotting. An indexing scheme based on character bi-gram is also introduced to further reduce the number of potential candidate regions.

Chapter 4

Text Box Proposals

In the last chapter, we have shown segmentation free word spotting systems using Sliding Window over document page. We also make use of indexing techniques for faster query evaluation. The main disadvantage of any sliding window based method is computational complexity. For a $n \times n$ image a basic sliding window scheme needs to evaluate an order of $O(n^2)$ candidate windows.

For a typical document image of size 1024X2035 in the George Washington collection this leads to the order of 10^6 evaluations per document page. This makes it impractical in case of big databases and also prohibits the use of costly (but usually more discriminating) representations like the PHOC representation[4]. In addition, in the case of word spotting, sliding window based methods usually rely on a fixed aspect ratio of the candidate window derived from the query itself. This constraint can lead to poor results due to the large variability in the handwriting style.

In this chapter we propose to pre-segment the document images into an over-complete set of candidate bounding boxes (referred as text box proposals). Our main motivation is that exhaustive search leads to many redundant computations, which can be drastically reduced targeting the search only towards the relevant regions. These candidate bounding boxes are generated by means of connected component analysis and grouping the connected components based on different criteria. Connected component analysis is a good choice as all relevant text areas can be covered ensuring high recall. The generation of these text box proposals is similar in spirit with the bound-

ing box proposal scheme used in different computer vision tasks [60, 72]. The goal is not to obtain a perfect word segmentation, but to include all relevant text areas in the set of candidates in spite of introducing noise, overlapping or redundancy. We extend the use of integral image of pre-computed attributes introduced in previous chapter for fast computation of the PHOC representation at any location of the image. We adapt the integral image to work with the result of the pre-segmentation obtained with the text box proposals. In this way we can obtain a significant reduction of memory storage and computation time.

In previous chapter we have seen that retrieval time can be further reduced by indexing the document regions based on estimated presence of character bi-grams. In this chapter we extend this idea to proposed text box proposals. Thus we add an indexing scheme that permits to evaluate only the text boxes that are relevant to the given query according to the information stored in the index. In contrast to the previous chapter the indexing scheme now uses character n-grams as key to index rather than using only bi-grams. Thus we extend the PHOB representation introduced in chapter 3 to use character n-grams instead of only bi-grams we refer this representation as PHON (Pyramidal Histogram of N-grams) henceforth in this chapter.

The rest of the chapter will be unfolded as follows: In section 4.1 we summarize the overall architecture used for QBE and QBS word spotting. In section 4.2 we discuss how text box proposals are generated by pre-segmentation of the documents. Section 4.3 discusses the computation of integral image of attributes in case of text box proposals and how it is used for faster query evaluation. In Section 4.4, we introduce the n-gram-based indexing (PHON) of text box proposals. Section 4.5 describes the retrieval procedures adopted for both QBE and QBS. Section 4.6 presents and discusses the experiments done to validate the proposed scheme. Finally, we conclude in Section 4.7

4.1 Overall Architecture

We present in this chapter a complete system for word spotting based on the PHOC word representation that can be used in both QBE and QBS segmentation-free scenarios[1] and overcomes some of the limitation of previous approaches.

[1]We understand by segmentation-free the ability of the method to search a word in a whole non-segmented page as opposed to a segmentation-based scenario where retrieval is performed on segmented word images.

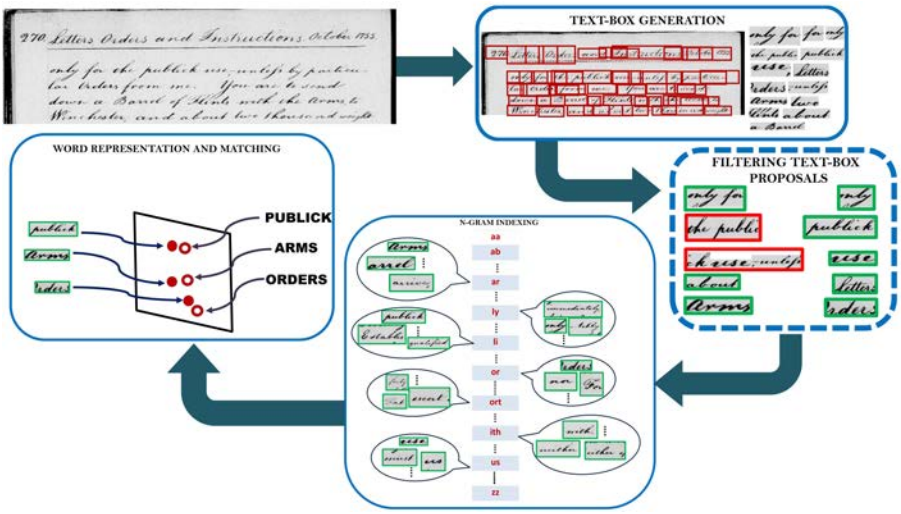


Figure 4.1: Proposed Framework

The approach proposed is illustrated in Figure 4.1. First a set of candidate windows (text box proposls) is derived offline for all documents in the database using the method described in section 4.2. All these text boxes are indexed based on the PHON representation that accounts for the N-grams contained in the candidate window. At query time the PHON representation of the query (that can be either a word image or a string) is used to select through the index the set of relevant text boxes. The final ranking is performed according to the cosine distance between the query and all selected text boxes using the complete PHOC representation. This computation can be performed very efficiently thanks to the integral image of PHOC attributes pre-computed offline.

4.2 Pre-segmentation: Generation of Text Box Broposals

Ideally candidate generation generation should produce a set of bounding boxes that perfectly segment the document at world level. However getting an ideal segmentation

at word level is very difficult due to the diversity of handwriting. Therefore, our goal in this work is not to get perfect word boundaries, but rather to propose an over-complete set of bounding boxes which covers most of the words present in the document.

To achieve this we rely on a rather simple method. First a set of *atomic bounding boxes* is extracted from the document image. Then, these atomic bounding boxes (BBs) are combined and grouped together to get the final set of text box proposals following a set of rules which guarantee a high recall in the localization of word boundaries.

4.2.1 Generation of Atomic Bounding Boxes

In search of atomic bounding boxes, our goal is to extract a set of candidates (called atomic bounding boxes) which do not contain more than a word and cover all text areas of a document. In this way, the combination of one or more of such bounding boxes can form potential candidate words, as it will be described in the next section.

We have decided to use a simple process based on connected components to extract the set of atomic bounding boxes, basically for two reasons. Firstly, connected components can easily be extracted from the document image. This process is fast and reliable and avoids using any threshold on the size of the atomic bounding boxes. Secondly, in handwriting, connected components are mostly formed by pen strokes made by writers and thus atomic in nature. In general, they will not span more than a word. Although this is not always completely true, we argue that these situations (touching adjacent words either horizontally or vertically) only occur in very specific and rare cases and do not have a real impact on the performance as we will see in the experimental results.

For connected component analysis, the document is binarized by setting the threshold as 75% of the mean intensity of the image and then connected components are extracted using 8-pixel neighbourhood. As in this stage our goal is to retain as much information as we can in the images, rather than finding a clean binarized image, we use a high threshold to enhance the overall recall. Although this can lead to a larger number of false positives, the retrieval process can later discard them.

Henceforth we will refer to the bounding boxes of the connected components as atomic bounding boxes since in our pipeline we perceive them as one unit of text image (which can contain a part of a word up to a full word), that can not be separated.

4.2.2 Combining Atomic Bounding Boxes into Text Box Proposals

In general, and by definition, atomic bounding boxes represent only some part of a word. In this section we will describe how we can combine atomic bounding boxes in order to obtain potential candidate words, referred as text box proposals. Since atomic bounding boxes are essentially the result of connected component analysis, not any combination of atomic bounding boxes is meaningful in terms of text box proposals. Therefore we will impose some spatial constraints on the total possible number of combinations to guarantee that candidate words are only composed of horizontally neighbouring atomic bounding boxes.

Constraint by Co-linearity

The first constraint that we apply is co-linearity. Clearly atomic bounding boxes from two different text lines can not form a valid candidate word bounding box. Thus, we will only generate combinations of atomic bounding boxes that could be considered collinear. For that, we avoid to use an explicit line segmentation method. Line segmentation algorithms try to find a perfect global line separation and, usually they are prone to errors and sensitive to noise. However, our goal is not to find perfect line separation but rather to infer potential collinear atomic bounding boxes.

In order to achieve that we propose a simple method to generate an over-complete set of line separation hypotheses. Then, every atomic bounding box will be assigned to all the lines with a certain degree of overlapping with the bounding box. All atomic bounding boxes assigned to the same line will be considered as collinear. Let us note that one atomic bounding box can span more than one line hypothesis and therefore, can be combined with atomic bounding boxes in different lines. This is a way of assuring a high recall of word hypothesis.

The generation of the over-complete set of line separation hypotheses is based on the local minima of the horizontal projection. Actually, horizontal projection profile has been used by many researchers [37] to estimate the exact text lines. As we are only interested on possible segmentation points we simply search for the local minima of the projection, applying first a moving average filter in order to smooth the projection profile and remove false separation points due to small artifacts near the boundaries of text lines. Ascenders and descencers can also introduce noise in the projection profile. Therefore, we pre-process atomic bounding boxes before computing the horizontal projection in order to remove this noise. For that, we replace every atomic bounding

box by a minimal bounding box obtained following a greedy approach where we first compute the pixel density of the atomic bounding box. Then, starting from the middle of the bounding box we keep growing in both vertically and horizontally directions until we reach an area which contains 90% of the pixel density of the corresponding bounding box. Figure 4.2 shows an example of the application of this simple heuristic.

Once the set of line separation points is determined, every atomic bounding box is assigned to all the lines for which the overlapping area is greater than a given threshold. Finally, all the atomic bounding boxes assigned to the same line will be considered collinear.

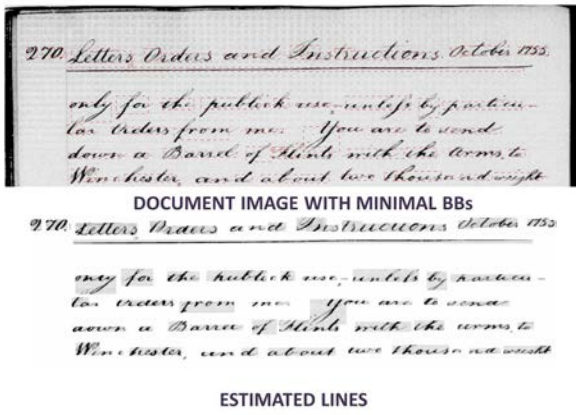


Figure 4.2: *a*) Input Image and corresponding minimal bounding box shown in red and *b*) Image after each connected component is replaced by its minimal bounding box

Constraint by Spatial Ordering

Once each atomic BB has been assigned to one or more lines, text box proposals will be generated as combinations of atomic BBs within the same line. For that we will make use of the knowledge that words are arranged left to right in English and will enforce some spatial ordering. Thus, we sort all atomic BBs from left to right according to the x position of their top left corner. Then, text box proposals will be generated as any possible combination of consecutive atomic BBs assigned to the

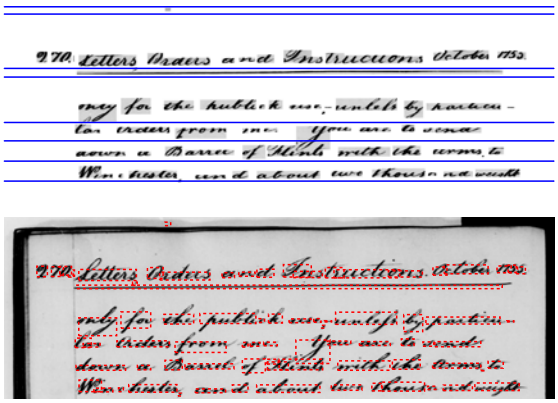


Figure 4.3: *a*) Line estimation *b*) Atomic Bounding Boxes For each line

same line. An illustration of this combination of atomic bounding boxes is shown in Figure 4.4 for combinations up to three atomic bounding boxes.

In practice this gives a manageable number of text box proposals to be evaluated at query time. Theoretically, any combination of collinear consecutive atomic bounding boxes could be used to form text box proposals. However in the experiments section we will discuss how this set can be further reduced by imposing some threshold to the size of text box proposals based on the size of the input query image. Finally, the set of all text box proposals can be defined as the union of text box proposals for all the lines in the document.

4.2.3 Filtering Text Box Proposals

In the previous section we combined all possible atomic bounding boxes within a line to get an overcomplete set of text box proposals. There, the goal was to maximize the recall by including all possible word candidates. However that will also include many false positives. To reduce the computation time in further steps, we propose to use a simple and fast binary classifier in order to classify all these text box proposals into word/non-word.

To learn this binary classifier we use simple features, which can be computed

Figure 4.4: Combining Atomic BBs: *LEVEL1*: Atomic Bounding Boxes *LEVEL2*: Composite Bounding boxes (candidate text boxes) taking 2 atomic BBs at a time *LEVEL3*:Composite Bounding boxes (candidate text boxes) taking 3 atomic BBs at a time

very fast. For each text box proposal we compute a fixed length feature vector in the following way. Every text box proposal is divided vertically into P segments and horizontally into Q segments. For each segment the pixel density is calculated. This gives a $P+Q$ dimensional feature vector. We add to this feature vector the height and the width to the text box proposal normalize with respect to the average line height and width of all the text box proposals in the training set, respectively. This leads to a final feature vector of length $P+Q+2$.

A linear support vector machine binary classifier is learned using these features. In training a text box proposal is considered as positive if the intersection over union with any of the words in the ground truth is at least 0.5. Otherwise it is considered as negative. At test time, text box proposals with a classification score lower than a given threshold t are filtered out from the subsequent processing. We tested with different values of t. Details are in the experimental section.

4.3 Efficient Computation of Word Attributes using Integral Image

In a scenario where the retrieval procedure has to rely on deciding among many (probably overlapping) candidate windows, the main bottleneck of using word attributes as basic representation is that it involves the redundant and costly computation of SIFT descriptors and Fisher Vector representation at run time.To alleviate this problem, in previous chapter, we introduced the idea of pre-computation of the attribute representation for every pixel (or over a regular grid) of the image and store it in an efficient integral image [63] format off-line. At query time this integral image of attributes can be then used to compute the representation of any candidate window using only four vector additions, and thus making actual retrieval faster.

In this chapter we extend this idea further based on the observation that text box proposals will always be the combination of a set of atomic BBs. Thus, we will only need to compute the attribute representation in rectangular areas corresponding to the boundaries of atomic BBs. Thus, word attributes can be accumulated only at every possible atomic BB boundary rather than at every pixel (or over a regular grid), reducing storage requirements.

Recall from the previous chapter that attribute representation can be computed for every point in the image and be accumulated over a integral image.

The naive approach to using this integral image of attributes consists of storing the attributes at every pixel of the image. Then at run time, given a candidate window its corresponding attribute representation can be computed by 4 vector additions. However the storage requirements of such methods is a bottleneck for their implementation in real life situations. One $n \times n$ image will need $n \times n \times 604$ floating point descriptors to be stored, which is not feasible for even a very small size database. For example a low resolution image with resolution 500×500 will require $4 \times 250000 \times 604 = 1000000$ bytes which is approximately 6 GB of data for a single image.

In previous chapter, we propose to reduce this storage requirement by arranging the pixels of an image over a regular size grid. Then, instead of computing the attributes for every pixel, we only need to compute one attribute descriptor for every grid, thus reducing the storage requirement by the factor of k^2, where k is the size of the grid.

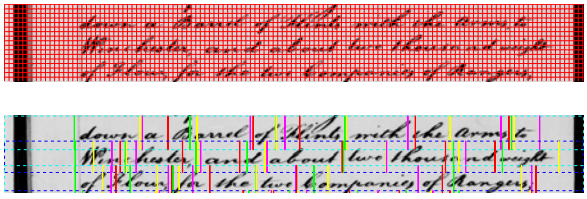


Figure 4.5: Regular sliding windows Vs Evaluation based on combining Atomic Bounding Boxes

In this chapter we improve on this scheme and instead of arranging the image pixels in a regular grid of size $k \times k$, we define the grids based on the position of atomic bounding boxes, as it is shown in figure 4.5. Apart from the corners of the atomic bounding boxes, points in the grid are generated at all intersection points between line hypothesis and vertical edges of atomic bounding boxes.

Once we have found the positions of the irregular grid for every page, we compute the word attributes for every cell of the grid and accumulate them over an integral image format for every line hypothesis. At every intersection points attributes from all atomic bounding boxes of it's left is summed and stored.

Text box proposal are stored in terms hypothesized line number and start and end position of atomic bounding boxes. It can be easily followed that given these infor-

mation exact attribute can be recovered by using simple vector additions.

4.4 N-gram based Representation for Indexing

In chapter 3 we proposed an indexing mechanism based on presence of character bi-grams, in this chapter we further extend this to estimate the presence of character n-grams for indexing text box proposals.

The representation used for indexing is similar to the PHOC framework described in Section 3.2.2. Our starting hypothesis is that character n-grams and their positions can be discriminative word features. Text regions can be effectively indexed per n-grams in an offline stage. Thus, the goal of this new representation is to identify the presence of a particular n-gram in a candidate image.

To select the n-grams that are used to generate the index we refer to the study done in [28], where they analyzed a large corpus of English words from different sources. (1) full-text articles from the New York Times(NYT) from January to March 1992(14 million words), (2) a subset of the Brown word corpus (Kucera & Francis, 1967; 1 million words), (3) a commonly used online encyclopedia (7 million words), (4) text extracted from about 100,000 randomly selected Web pages(61 million words), and (5) newsgroup text extracted from 400 different Internet discussion groups(100 million words). As in chapter 3 we consider the 150 most popular bi-grams, but we also added 50 most popular tri-grams from this study which covers 99.21% of the total corpus. To include numeric fields the digits from 0-9 are also included in our representation. Then, the N-gram attribute representation is obtained using 150 bi-grams and 50 tri-grams at level 2 of decomposition following the same framework explained in section 3.2.2 and using 10 digits at levels 2 and 3, thus resulting in a 450 dimensional representation.

Using also the strategy described in section 3.2.2 and similar to PHOB in chapter 3, an embedding function is learned to project the Fisher Vector representation of text box proposals into the PHON attribute space. Both representations are then converted to a low dimensional common subspace using canonical correlation analysis (CCA). Similarity between n-grams and text box proposals can be computed as a cosine distance (dot product) between their corresponding representations in this space. For each n-gram, text box proposals are sorted in order of decreasing similarity and stored as inverted index files (Figure 4.6. At retrieval time only top N of these proposals are explored for final retrieval.

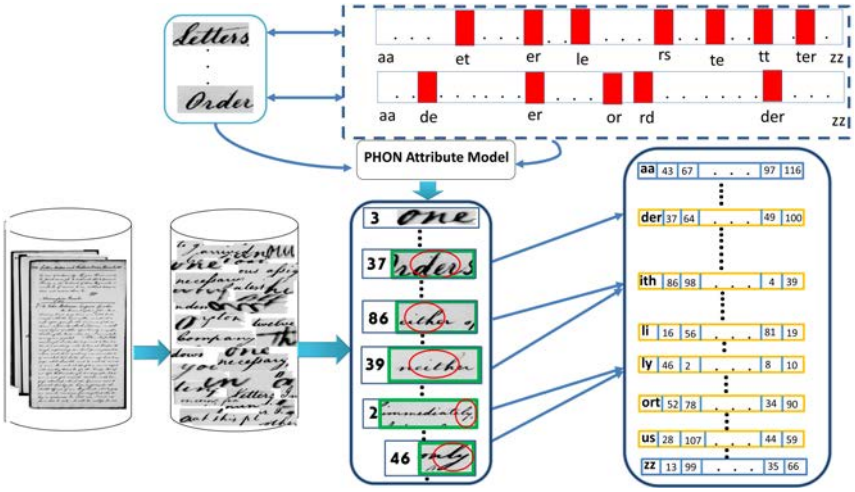


Figure 4.6: Proposed Indexing Scheme

4.5 Retrieval

Using the text box proposals generation, the word representation and the indexing scheme described in previous sections, retrieval can be performed using either an image (QBE) or a string (QBS) as a query. In both cases, the first step consists of retrieving using the index a reduced set of candidate text box proposals relevant to the query based on the N-grams of the query. In a second step, all these relevant text box proposals are ranked according to the similarity to the query based on the PHOC representation. In the following subsections we describe in detail these two steps.

4.5.1 Retrieval of Relevant Text Boxes using the Index

As a first step we use the indexing structure described in section 4.4 to generate a partial set of relevant text boxes. The use of the index is slightly different depending on whether the input query is a string (QBS) or an image (QBE):

QBS: Given a text query, first all distinct n-grams of the text string are found. Then, for each character n-gram, the inverted index is searched and the top n text box

proposals for each one are considered as relevant for retrieval. Thus, for a query having k distinct n-grams, $k \times n$ number of candidate text boxes are considered. However, it should be taken into account that the same text box proposal can be considered as relevant by different n-grams in the index. Therefore, the final number of distinct text box proposals to be evaluated will be much lower in most cases.

QBE: Given a query image, first its PHON attribute representation is found, and then we compute the dot product with the PHON representation of each of the n-grams in the index. Finally. we take the top k n-grams based on this distance. Once n-grams relevant to the query are identified, a similar procedure to that described for QBS can be employed, i.e the inverted indexing structure is accessed and the top n text box proposals for each n-gram are further considered for retrieval.

4.5.2 Ranking of Text Boxes

Finally, given a query (either an image or a string) and the list of relevant text boxes obtained through the index we have to rank this list according to the similarity of each text box proposal with the query (either a string or an image). For that, we need to obtain the attribute representation of each text box. We will rely on the integral image described in section 4.3 and thus, the attribute representation Ψ_w of a candidate window w will be computed as usual with integral images with just 4 vector additions followed by $L2$ normalization as:

$$\Psi_w = |\Psi(X_2, Y_2) + \Psi(X_1, Y_1) - \Psi(X_1, Y_2) - \Psi(X_2, Y_1)| \tag{4.1}$$

where (X_1, Y_1) are the co-ordinates of the top left corner and (X_2, Y_2) are the co-ordinates of the bottom right corner of the window.

To compute the similarity between the query and each of the candidates we use the cosine similarity by taking dot products after normalizing the output embeddings. The resulting text box candidates are sorted in decreasing order of similarity to give the output result candidates. At the end non maximal suppression is performed and candidates with more than 30% overlap are removed to present the final result.

4.6 Experimental Results

To evaluate our method and compare with other state-of-the-art methods we use the common protocols and datasets described in chapter 2. Though all the common protocols are described in chapter 2 some more analysis is needed which are specific to this chapter, thus we described the protocols for evaluating the generation of text proposals, the influence of the filtering step in section 4.6.1 before describing the result of word spotting in later sections. Results are presented in the following sections including a detailed comparison with the state-of-the-art.

4.6.1 Evaluation of Text Box Proposals

First we evaluate the quality of the text box proposals obtained with the method described in section 4.2. The desirable properties for a bounding box proposal algorithm are basically achieving a high recall with a low number of proposals and a low computation time. Thus we used the following protocols to measure the quality of text box proposals and the subsequent text box filtering.

Text Box proposals generation: The main concern of a text box proposal system is to obtain a high recall with as less number of candidates as possible. Text box proposals are independent of the query as relevance to the query is computed at a later stage. We have used the GW dataset to evaluate the performace of the generation text box proposals. As our method does not need any training, we present the result in GW dataset for all the 20 pages. We evaluate the quality of the proposals with respect to the average number of candidates proposed and the recall.

Filtering of Text Box Proposals: as explained in previous sections we have used a SVM based binary classifier to filter proposals. In the case of the GW and BCN datasets we use 4-fold cross validation using two folds for training, one for validation and one for test. Results are the average of the 4 folds used for testing. We plot recall vs number of candidate that permits to analyze the suitable value for the threshold used to classify a box as valid or not.

Obtaining a high recall is crucial as only proposed text boxes are considered in subsequent phases of retrieval. Ideally one would desire to get a high recall with the minimum number of candidate windows to be evaluated. However there will always be a trade-off between them. We should take into account that text box proposals that are not relevant to the query can still be rejected in the next steps of the retrieval

process.

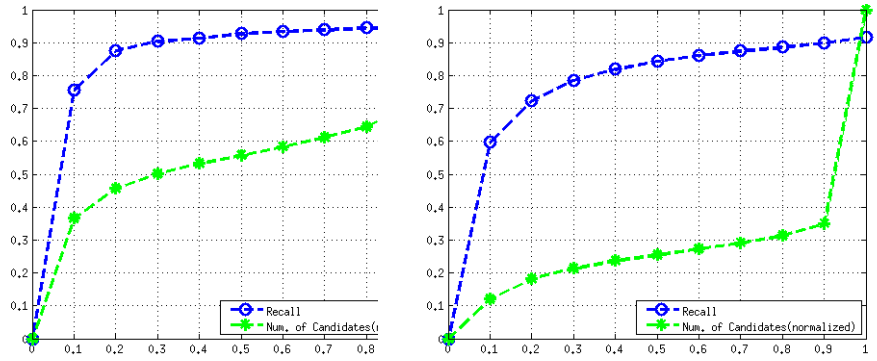
In table 4.1, we summarize the performance of text box proposals with respect to the above parameters. As there is no other bounding box proposal algorithm available for handwritten documents, we compare our scheme with a basic sliding window based protocol. Sliding window at every position will achieve a 100% recall as all positions of the document are analyzed. However, as we discussed in the beginning of this chapter, this will result in lot of redundant windows e.g. a typical page in GW dataset will generate in the order of 10^6 number of windows consequently the cost of evaluation at retrieval time can be too high. Thus, instead of computing the sliding window densely, an alternative to reduce the computation time is to evaluate the sliding window with a fixed step size was proposed in chapter 3. Second row of table shows the results using a step size of 16.

Method	Step Size	Avg. # of candidates	Recall	comp. Time Per Page
Sliding Window	1	6×10^6	100	-
Slidiing Window	16	23070	78.04	-
Text Box Proposals	-	6564.55	96.09	0.2
Text Box proposals-Filtered	-	3506.7	95.54	0.4

Table 4.1: Evaluation of Text Box proposals in comparison to basic sliding window protocol for GW dataset

In the third row of the table we can see that our method achieves a recall of 96.09 with only 6564 text boxes per page on average while a step-based sliding window achieves only 78.04 recall evaluating more than 23k number of windows. This depicts that our method is not only capable of making the whole pipeline faster, but it also makes it more efficient and captures more valid candidate text areas. One more important observation is that our method naturally deals with variable word sizes, as it is based on the combination of several atomic text boxes, given by connected component analysis. Thus to generate candidate text boxes it does not need to assume any predefined size of the query, as sliding window methods need to do. This helps in word spotting where we do not have a reference query image (for example in QBS) or in spotting in multi-writer datasets where the query from one writer and the search space from multiple writers can have different sizes. To cope with this situations in a sliding window setup we need to evaluate the sliding window at multiple scales, which leads to the generation of even more candidate windows and increases the overall computation time of the system.

Evaluation of Filtering of Text Box Proposals



(a) GW Dataset (b) BCN Dataset

Figure 4.7: Recall and Number of Candidates for different values of threshold for GW and BCN Datasets. The value in x-axis corresponds to $1 - t$, being t the classification threshold. Thus, low value in x-axis means high value for t, that is less number of candidates

To observe and report the impact of filtering of text box proposals introduced in section 4.2.3, we plot the recall and number of candidates (normalized) for different values of the threshold of the binary SVM classifier in GW and BCN datasets in Figure 4.7. The plots in green shows the number of candidates normalized in the interval [0,1] and can be interpreted as the percentage of candidates that are selected with respect to the total number of candidates obtained at maximum recall.

We can see that for both the datasets recall initially increases very rapidly and after a certain point, the increase in recall reduces significantly while the number of candidates keeps increasing. This means that a good percentage of candidates can be safely removed by using the classifier, as we can see in the last row of Table 4.1 that shows that using the classifier to filter text box proposal we can keep recall while reducing significantly the number of proposals.

To maximize the recall in all subsequent results we will set the threshold to 0.4, which gives 3506.7 candidates per image on average in the GW dataset and 4121.1 per image on average in the BCN dataset.

4.6.2 Evaluation of Word Spotting

Finally we evaluate the retrieval performance in both QBE and QBS scenarios. In this section we evaluate word spotting without using any indexing i.e we evaluate each and every text box from the entire document set. In the following section we use proposed indexing mechanism to evaluate word spotting.

For each dataset we provide results for four different settings concerning the generation of text box proposals:

1. All candidates: this is the scenario where all text box proposals are evaluated without considering any posterior filtering.

2. All Candidates + width: in word spotting filtering candidates based on the width of the query is a reasonable approach. Thus, in this setting we only consider text boxes which satisfy the following condition regarding width: $w_q/2 \leq w_t \leq 3 \times w_q/2$, where w_q and w_t are the width of the query and the candidate text box respectively.

3. Filtered Candidates: in this setting text boxes are filtered based on the binary text/nontext classifier (Section 4.2.3) so that only text boxes above a certain threshold are considered.

4. Filtered Candidates + Width: this setting is the most restrictive among all. Here we impose both the above conditions i.e filtering based on the text/non text classifier and the constraint about the width of candidate text boxes.

In the following subsections we report the results for different scenarios. In section 4.6.2, we discuss the results for QBE word spotting and in section 4.6.2 the results for QBS word spotting.

QBE Word Spotting

In table 4.2 we can see the results of the evaluation of QBE for all datasets. From the results we can observe that simply filtering text box proposals by width permits to increase precision while also reducing the computation time. If, in addition we apply the filtering of text boxs proposals with the binary text/non-text classifier we keep the same accuracy, but we can further reduce the computation time. Note that in

Dataset	Methods	mAP	Recall	Comp.Time in seconds Per Query	Comp. time in Seconds per query per page
GW	All Candidates	72.87	90.10	2.71	0.004
GW	All Candidates+width	77.05	91.36	0.716	0.0049
GW	Filtered Candidates	74.83	91.28	1.72	0.0033
GW	Filtered Candidates+width	77.15	91.81	0.75	0.0047
IAM	All Candidates	36.48	56.85	17.96	0.0033
IAM	All Candidates+width	38.73	58.60	11.04	0.0037
IAM	Filtered Candidates	36.72	57.95	13.45	0.0032
IAM	Filtered Candidates+width	38.73	58.60	10.8	0.0038
BCN	All candidates	75.3	86.06	3.25	0.0036
BCN	All candidates + width	77.06	85.72	0.84	0.0038
BCN	Filtered Candidates	75.60	83.96	1.44	0.0041
BCN	Filtered Candidates+width	76.86	83.86	0.77	0.0039

Table 4.2: Evaluation of QBE word spotting

order to make the time comparable between different datasets, we also report the time normalized by the number of pages in the test set.

QBS Word Spotting

For QBS results are presented in table 4.3 and show a similar behaviour as in the case of QBE. Filtering with width increase precision and reduces computation time while adding the filter with the text/non-text binary classifier permits to further reduce computation time without decreasing precision.

Dataset	Methods	mAP	Recall	Comp.Time in seconds Per Query	Comp. time in Seconds per query per page
GW	All Candidates	67.56	90.01	2.71	0.0048
GW	All Candidates+width	69.94	91.62	0.84	0.0049
GW	Filtered Candidates	68.30	90.03	1.73	0.0035
GW	Filtered Candidates+width	70.3	91.57	0.88	0.004
IAM	All Candidates	42.83	61.60	13.45	0.0029
IAM	All Candidates+width	43.73	63.48	10.65	0.0034
IAM	Filtered Candidates	42.53	61.60	11.87	0.0037
IAM	Filtered Candidates+width	44.73	60.60	10.94	0.0038
BCN	All Candidates	68.08	86.96	6.34	0.007
BCN	All Candidates+width	69.22	86.53	4.21	0.0045
BCN	Filtered Candidates	68.9	86.7	2.17	0.005
BCN	Filtered Candidates+width	69.80	86.26	1.85	0.0027

Table 4.3: Evaluation of QBS word spotting

4.6.3 Word Spotting with N-gram Indexing

In this section we report the results for both QBE and QBS using the n-gram based indexing scheme described in section 4.5.1. Figure 4.9, shows some examples of the top indexed regions for some n-grams. We can see that in general, the indexed images contain the corresponding n-gram.

Dataset	Mode	Setup	mAP	Recall	# Index Per N-gram	Comp. Time
GW	QBE	Efficient	69.8	85.84	300	0.30
GW	QBE	Best Trade-Off	76.05	90.01	700	0.45
GW	QBE	Best-performing	77.15	90.15	1000	0.75
GW	QBS	Efficient	64.8	78.5	700	0.40
GW	QBS	Best Trade-off	69.43	87.8	1200	0.65
GW	QBS	Best-performing	70.15	90.15	2500	0.88
BCN	QBE	Efficient	69.15	85.65	700	0.45
BCN	QBE	Best Trade-off	75.56	87.9	1500	0.55
BCN	QBE	Best-performing	76.6	86.67	2500	0.77
BCN	QBS	Efficient	65.4	89.50	800	0.45
BCN	QBS	Best Trade-off	69.5	87.9	1800	1.03
BCN	QBS	Best-performing	71.08	86.57	3000	1.85

Table 4.4: Evaluation of n-gram Indexing for QBE and QBS

In the graph of Figure 4.8 mAP, recall and the average number of candidate text boxes per query is plotted against the number of indexed regions per N-gram that are considered for retrieval. It can be observed that mAP and recall grow rapidly at first, but reach a relatively stable point after a certain value of N, where N is the number of indexed candidate regions per character N-gram. In the graph we can observe that high values of recall and precision can be achieved using the index with a relatively low number of indexed candidates per N-gram, showing that indexing removes many false positives. We can also observe that after a certain value of N adding more candidates per N-gram does not translate into big improvements in precision. There is a trade-off between computation time (that is directly related to the number of indexed region per N-grams) and the performance in terms of precision and recall.

In Table 4.4, we report QBS and QBE performances with three different choices of N, where N is the number of indexed region per N-gram. Our three different choices correspond to three different settings namely efficient, best trade-off and best-performing. For the efficient set up we chose a number of indexed regions such that a reasonable performance can be achieved with a low number of candidates. The best trade-off set up gives the best trade-off between efficiency and performance whereas in the best performing setup we report the performance for which we achieve the best

mAP, i.e we fix the size of the index (number of candidates per N-gram), so that it provides the best mAP for each dataset.

If we compare the results using the index with those reported in tables 4.2 and 4.3 obtained without using the index we can observe that we can keep the same accuracy using the index, but reducing the computation time per query as a lower number of candidates need to be evaluated. Even, in some cases, using the Best-performing setting, accuracy can be slightly improved using the index while keeping the same computation time per query. This confirms that indexing helps to remove false positives from the list of candidate text boxes.

4.6.4 Discussion and Comparison with Related Works

To come up with a comparative evaluation of state-of-the-art methods for word spotting is not straightforward due to several reasosns. The main challenges include the absence of a single evaluation protocol in the literature, the use of different datasets in the reported works and finally, the existence of different subtasks and constraints in word spotting (QBS, QBE, segmentation-free, segmentation-based, single writer, multi-writer, etc.).

In recent times many authors presented their results on GW dataset and thus, GW has become a standard dataset to evaluate and benchmark word spotting methods. However different sets of pages, query words and even evaluation measures have been used in different works making the comparison between them non-trivial. Among the two variants of word spotting (namely QBE and QBS) QBE is much more investigated than QBS. Thus, comparing results in the case of QBS word spotting is even more difficult. In a recent work Rusiñol *et al.* [50] reported an exhaustive review of many state-of-the-art methods. To compare their method with the state-of-the-art they simulated the same evaluation protocols used in each of the original works and reported the results for each of them. In this section we use the same strategy to evaluate our method in comparison to teh state-of-the-art methods. Thus, in the last column of table 4.5, we report the results of our method using the same training/test partitions, the same evaluation measures and the same protocol as described by the authors of the method compared in that row of the table. The methods proposed by Rusiñol *et al.* [50] and Almazán *et al.* [2] use the same evaluation protocol as described insection 2.2. In [34, 59], the authors use a selected set of 15 word images as queries to perform word spotting. As results for word spotting can vary depending on the word samples used as queris we report results for a set of word images with the same tran-

scription and which appear to look similar to those used in the original works. In the work proposed by Rothacker *et al.* in [49], a candidate is considered as positive if the transcription matches with the ground truth and it overlaps any of the ground truth bounding box by at least 20% of the area. Therefore, we have adapted our evaluation protocol to each of these variants and we have reported the corresponding results in each row or table 4.5. We can see that in all cases we achieve a better result than the original method, in most cases by a signifficant margin.

Method	Segmentation	Dataset	Original Result	**Proposed Method**
Slit style HOG features [59]	Line Level	20 pages, 15 queries	79.1% mAP	**86.38%mAP**
Gradient features with elastic distance[34]	NO	20 Pages, 15 Queries	60% P=R	**82.09% P=R**
BOVW+HMM [49]	No	5 pages Test and 20% overlap is considered as true positive	61.35%mAP	**83.06%mAP**
Exemplar SVM [2]	NO	All ground truth words as queries	54.5% mAP	**77.15%mAP**
bag-of-visual-words+LSI [50]	NO	All Ground Truth Words as Queries	61.35% mAP	**77.15%mAP**
Sliding Window-PHOC (chapetr3)	No	All ground truth Words as queries		
Sliding Window-PHOC with Integral Image(chapetr3)	No	All ground truth Words as queries		
Sliding Window-PHOC with bi-gram indexing (chapetr3)	No	All ground truth Words as queries		

Table 4.5: Comparison with state of the art methods for QBE in GW20

Method	Segmentation	Dataset	mAP
Text Box Proposal	**–**	**All unique words**	**69.9%**
Sliding Window(baseline)	**–**	**All unique words**	**56.4%**
Integral Image	**–**	**All unique words**	**56.4%**
Rothacker *et al.* [48]	–	15 pages Training-5 Pages Test	76.5%
Liang et al.[36]	Word-level	38 queries	67% at rank 10
Fischer et al.[12]	Line-level	In-vocabulary words	62. 08
Frinken et al. [14]	Line-level	In-vocabulary words	71

Table 4.6: Comparison with state of the art methods for QBS in GW dataset

	GW	BCN	IAM
Almazán *et al.* [2]	1.04s	-	-
Kovalchuk *et al.* [29]	0.08	-	-
Proposed(1)	**0.83s**	**0.85s**	**10.65s**
Proposed(2)	**0.84s**	**0.77s**	**10.94s**
Proposed(3)	**0.56s**	**0.45s**	**7.43s**

Table 4.7: Comparison of computational time per query with state-of-the-art methods. (1) All Candidates + Width (2) Filtered Candidates +widh (3) Filtered Candidates+width+indexing

Most of word spotting methods are designed for the QBE set up as it is more straightforward to compare candidate word images with query images. However, QBS

is much more user friendly as it allows users to interact with the database using the keyboard. Unfortunately, there are not many works reported in literature to compare with in this set up. Most of the reported works in QBS word spotting are designed for segmentation-based scenarios, where some kind of previous segmentation either at line or word level is necessary. For example, the works reported by [12] and [14] use segementation at line level. In [36] word spotting is done with already segmented words. Moreover these works used a small subset of words or in-vocabullary words, i.e only words that appeared in training can be used as queries.

Recently Rothacker *et al.* in [48] proposed a word spotting method using HMM and Bag of visual words which works in a completely segmentation-free scenario. This work is the most similar to our work but they used a slightly different evaluation scheme. They used 15 pages of the GW dataset as training and the remaining 5 pages were used for test, whereas we used all 1124 unique words as queries and candidate words are generated from all 20 pages.

Therefore, the reported results in QBS are not directly comparable to our work in terms of evaluation protocol. Nevertheless, in table 4.6 we compare the results of our method with those related works in order to put the result of our work in the context or current state-of-the art. We can observe that our method obtains better results that some of the existing methods and it is still competitive compared to methods which work in a more constrained scenario regarding either segmentation requirements [14] or the number of queries and pages in the test set [48].

Another important aspect of any word spotting task is computational time. Computation time is difficult to compare due to the differences in hardware configurations and implementation between different methods. In spite of this we report in table 4.7 the computation time of our method for QBE compared to other methods that already have reported computation time. Although these results are not completely comparable and cannot be used to draw any hard conclusion, they are useful to show that the computation time of our method is at least competitive with respect to other methods.

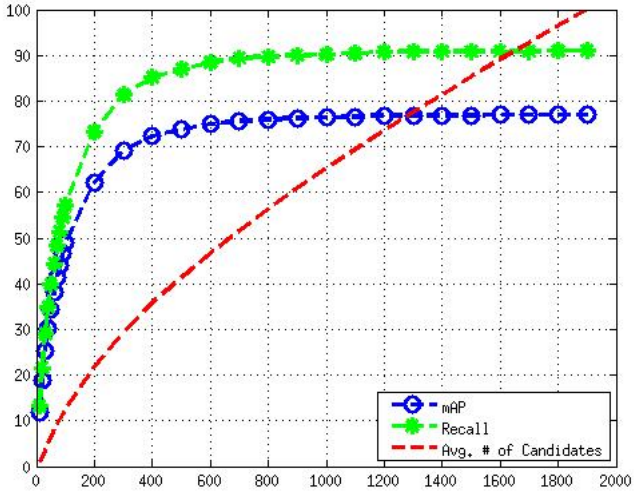
4.6.5 Qualitative Results

Finally, in figure 4.10, we show some qualitative results for both QBE and QBS tasks for different datasets. We can see how, in general, the words that are relevant to the query appear in the top positions of the retrieval list.

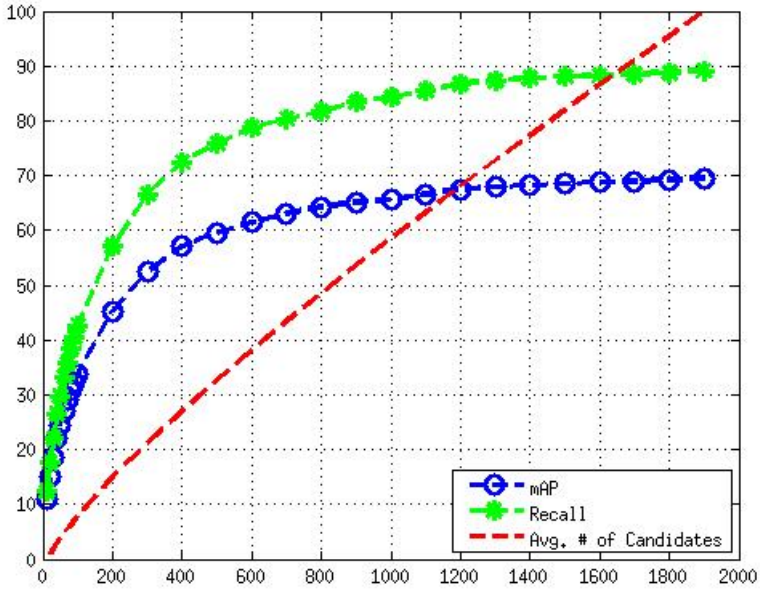
4.7 Conclusions

In this chapter we have described a complete pipeline for segmentation-free word spotting that can be used both for Query by Example and Query by String. In this pipeline word candidates are obtained using a new text box proposal algorithm. To the best of our knowledge this is the first attempt towards using bounding box proposal as intermediate step in the area of handwritten text understanding. These text box proposals are independent of any classification or retrieval scheme. Thus, any state-of-the-art word representation method can be used along with them. We have used the PHOC word attribute-based representation as it has shown state-of-the-art performance for segmentation-based word spotting. We have extended the idea of integral image framework based on word attributes proposed in last chapter, in order to adapt the framework to the variable size of text boxes. This adaptation increases the overall recall while maintaining the efficiency and reducing the storage space. We have shown that the new text box proposals permit to perform word spotting efficiently. In addition, computation time is further decreased using an indexing scheme based on n-grams. Comparison to the state-of-the-art reveals that our method does not only achieve higher accuracy among all non CNN based methods, but it is also competitive in terms of computation time.

In the following chapters we will investigate the impact of our text box proposals when used along with other state-of-the-art word representation and spotting methods and specially, trying to combine them with CNN-based methods for word representation.



(a) n-gram indexing in QBE tasks



(b) n-gram indexing in QBS tasks

Figure 4.8: Performance of n-gram indexing

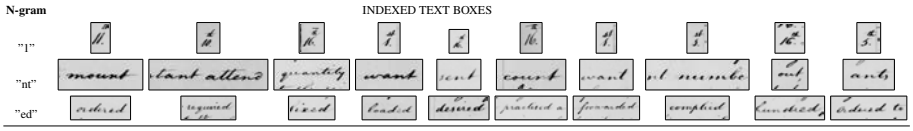


Figure 4.9: Top text boxes indexed by character n-grams

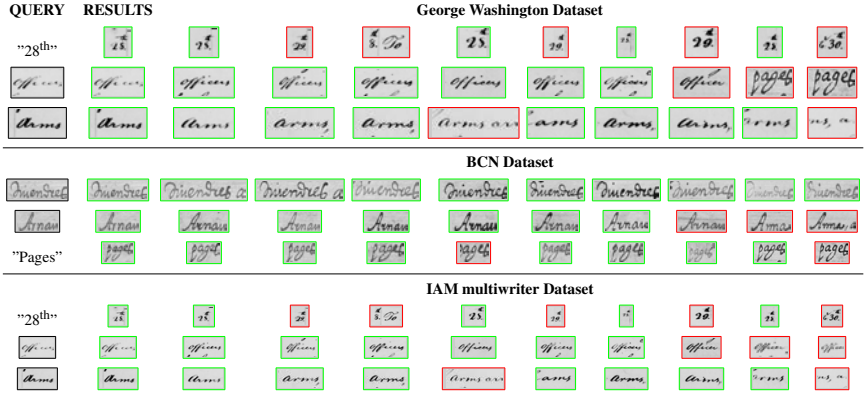


Figure 4.10: 10 top retrieval results for different queries in the three datasets evaluated in our experiments

Chapter 5

Deep Features for Segmentation Free Word Spotting

In earlier chapters we have seen that PHOC embedding gives a common word representation for both images and strings which allows direct comparison between word strings and images feasible. This property has been used in earlier chapters to extend PHOC based word spotting for segmentation free word spotting.

Originally the PHOC representation was learned using Fisher Vector as image features, recently some adaptations using CNNs to learn the attribute space have also been proposed [56, 44]. All the different variants of this representation have shown to be highly discriminant achieving state-of-the-art results in segmentation-based word spotting.

In this chapter we propose a framework to extend these CNN based PHOC representation to segmentation-free word spotting leveraging the text box proposals described in last chapter. We take advantage of recent works in object detection [16, 15] that leverage a set of blind object proposals to find all instances of objects in an image using a single forward pass in a CNN.

Thus we proposed a deep neural network (called R-PHOC) which is trained end-to-end to generate the PHOC representation of every candidate region. Given a query, word spotting can be performed by simply computing the distance between the PHOC representation of the query and the PHOC representation of all candidate regions ob-

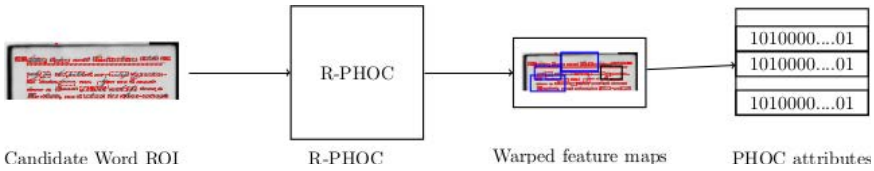


Figure 5.1: The proposed pipeline for segmentation free word spotting, First the whole document image is passed along with bounding box co-ordinates of candidate words, then the feature map corresponding to these regions are pooled by ROI pooling layer and finally passed to sigmoid activation to predict the PHOC attributes for the whole set of candidates.

tained through the R-PHOC network. As these can be computed in a single forward pass in the network, the whole procedure is very efficient in terms of computation time.

The rest of the chapter is organized as follows: In section 5.1, recent works related to our approach is described. In 5.2, we discuss the proposed methodology including basic PHOC embedding and details of training the PHOC embedding using region based CNN features. In section 5.4 we show the results of the experiments carried out to validate our approach. Finally in section 5.5 we report the conclusions of our work.

5.1 Related Works

In this chapter we investigate the use of a single deep architecture for segmentation free word spotting. In order to achieve that we make use of recent advancements in the field of 1) object detection and 2) CNN based attribute prediction. In the following we first review some of object detection pipelines and later attribute prediction by CNN.

Traditionally object detection using CNN features was performed using a sliding window protocol over the input image. However like any sliding window based approach this involves lot of redundant computation. To do this efficiently an specific architecture is needed, which can take as input the entire image and produce labels for all salient objects. The first breakthrough in this direction was made by Girschik *et al.* in [16], where they proposed the idea of region based CNN features and then SVM classifiers to classify regions into salient objects. A further modification of this approach was proposed in [15]. In Fast-RCNN [15] the concept of region of

interest(ROI) pooling was first introduced, which enables the network to aggregate features from different salient regions of the image without needing to feed all the regions separately to the network. In brief given a salient region and a CNN feature map, the feature corresponding to salient region can be computed by aggregating the features corresponding to the grids which comes under the salient region. In a nutshell a fast-RCNN framework takes a set of class independent object proposals as input and provides two set of outputs for each proposal: classification scores for each object category and an Offset for bounding box regression for each category of objects. Another object detection framework which gained attention in recent years is called YOLO []. In YOLO the feature map corresponding to an image is divided into grids and for every grid objects probabilities and bounding box is predicted for each category. This is in contrast to the fast R-CNN where given a feature map, features corresponding to object proposals is first computed by ROI pooling.

Our work in this chapter is also motivated by the recent advances in computing PHOC for cropped words using deep CNN network. First such attempt was made by *et al.* in [44]. A similar objective was served by Sabestian *et al.* in [56]. The main difference between these two works is the use of Spatial Pyramid Pooling Layer, which allows input image of different sizes as input to the network directly. In a recent work they extended this idea by introducing a temporal pooling layer and got further benefit in terms of results. On the other hand Wilkinson *et al.* in [67] used a convolutional triplet network to obtain a feature space from the handwritten words. This feature space is then further regressed to obtain PHOC embedding.

In [31] advocated the use of synthetic data to train the deep network in order to obtain PHOC embedding.

In this work we make use of PHOCNET architecture proposed by [56] and extend it for segmentation free word spotting. In order to achieve segmentation free word spotting the whole document page is used as input to the network. Feature corresponding to candidate regions are obtained by using ROI pooling similar to Fast-RCNN pipeline. Thus we replaced Spatial pooling layer with ROI pooling layer.

5.2 Methodology

An overall scheme of the framework is illustrated in figure 5.1. Document page image, a set of bounding box proposals and the query are input to the system. System consists of a convolutional neural network computes PHOC embedding for every bounding box

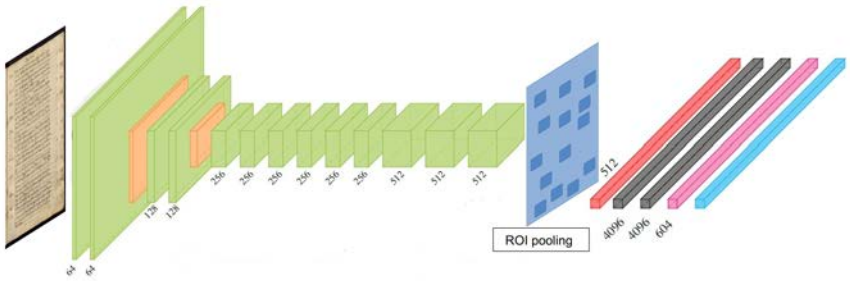


Figure 5.2: Architecture of our proposed R-PHOC network. All the layers before ROI pooling layer are same as PHOCNET [56], to facilitate transfer learning. ROI pooling is used to pool features over a candidate word region, in segmentation free word spotting

proposals. Retrieval is done by using nearest neighbour between query and bounding box proposals.

Convolutinal neural network used to embed the bounding box proposals into PHOC space is given in Figure 5.2. Architecture of the network contains the same set of convolutional and max pooling layers used [56] in order to leverage their pre-trained models. This is done purposefully to be able to use transfer learning, i.e. to reuse the learned weights from their pre-trained network, which is a common practice in other computer vision tasks like image classification etc. However for simplicity spatial pyramid layer is replaced by a ROI pooling layer. Each bounding box proposal is considered an ROI and feature obtained by ROI pooling is used to predict it's PHOC embedding. The feature representation produced by the ROI layer is fed to fully connected layers followed by sigmoid activation to produce PHOC embedding. We use the text box proposals obtained in chapter 4 as bounding boxes, however any other approach to generate bounding boxes can be used in the same way.

The ROI pooling layer uses max pooling to obtain a feature map from a ROI, by dividing the region of interest into sub-windows of fixed size. For example a ROI of (x, y, h, w) is divided into grids of approximate size h/H and w/W and pooling the features from each grid into the corresponding output grid cell. Similar to standard max pooling, feature from each channel is pooled independently.

Our goal is to train the network to predict the PHOC embeddings for each ROI (word candidate region) using the CNN features. Then, the problem is different to

most classification problems. Instead of having one true class for every training example in PHOC there can be multiple positive classes (PHOC attributes) for every training sample. For classification problems a softmax layer is the de-facto standard to obtain the final output. However for multi-label classification tasks this can not be used. This is dealt in [56] by using a sigmoid activation function in place of the softmax layer. In this work we also make use of sigmoid activation functions to obtain the final output of the network.

Sigmoid cross entropy loss (or logistic loss) is used for training the network. Thus, the loss between a predicted PHOC $\hat{\sigma}$ and the real PHOC σ of a given image is given as:

$$l(\hat{\sigma}, \sigma) = -\frac{1}{n} \sum_{i=1}^{n} \sigma_i \log(\hat{\sigma}_i) + (1 - \sigma_i)(\log(1 - \hat{\sigma}_i), \tag{5.1}$$

where n is the number of attributes in the PHOC representation σ. Given an image with a set of candidate words (ROIs), the loss is calculated as the summation of the individual loss for each ROI. Though the network could be used to regress the bounding box of every ROI to get a more precise location of every word, we leave that for future work.

5.3 Training

Training a deep network takes significant effort. Here, some of the hyper parameters are discussed in order to help the reproducibility of our method. As the loss function adopted in this work is differentiable, the network can be trained end-to-end by back propagation. We used stochastic Gradient Descent with a learning rate initially fixed at 0.0001 and updated every 1000 iterations. We trained the network for 30000 iterations. As the size of the document images in comparison to the images in Pascal dataset for object detection are much bigger, we divide the image into overlapping segments of size 600×1000 to be able to load the images in a GPU. One training epoch takes 0.524 seconds on average. We use a batch size of 128, i.e in one minibatch 128 ROIs are processed. The minibatches are sampled so as at least 60% of them contain valid text regions with an overlap of more than 0.5. From the text box proposals obtained using the procedure described in chapter 4, we only used the candidates with more than 50% overlap with ground truth words as text class and with less than 20% overlap as background. The text candidates with text overlap between 20% to 50% are filtered

as although they contain some text they do not constitute any valid word (e.g two consecutive words can be one candidate). Thus predicting PHOC for such candidates can act as a distractor.

5.4 Experimental Results

As usual we continue using the same protocol described in chapter 2 for segmentation free word spotting. For baseline analysis we performed QBE word spotting for segmented words in GW dataset. Each segmented word is assumed as a ROI and every document page is passed through the network once. The results for this experiment are shown in 5.1. Though the goal of this work is not segmentation-based word spotting, we did this study to analyse the effect of region based CNN features (particularly the efficacy of features computed by ROI pooling). As we can see that in segmentation-based scenario the R-PHOC method reaches 92.75% mAP which is comparable to FV-PHOC of [3]. R-PHOC performs slightly worse than PHOCNET, which is expected as in region based approaches features are integrated in discrete intervals. However, as described above this is very efficient in comparison to other PHOC based approaches as in one forward pass all candidate words of the entire document can be evaluated, thus achieving a massive parallelism.

Methods	**Mean Average Precision**
PHOCNET [56]	96.71
FV-PHOC [3]	93.04
R-PHOC (Region based CNN)	92.75

Table 5.1: Performance of R-PHOC for QBE word spotting in segmented word images)

Method	Segmentation	Dataset	Accuracy	Query evaluation time (in sec)
Recurrent Neural Networks [14]	Line Level	20 pages, all words of the training set appearing in all 4 folds as queries	71% mean prec.	
Character HMMs [12]	Line Level	all words of the training set ap- pearing in all 4 folds as queries	62% mean prec.	
Slit style HOG features [59]	Line Level	20 pages, 15 queries	79.1% mAP	
Gradient features with elastic distance[34]	NO	20 Pages, 15 Queries	60%	
BOVW+HMM [49]	No	5 pages Test and 20% overlap is considered as true positive	61.35% mAP	
Exemplar SVM [1]	NO	All ground truth words as queries	54.5%	
bag-of-visual-words+LSI [50]	NO	All Ground Truth Words as Queries	61.35% mAP	
Baseline PHOCNET	NO	All Ground Truth Words as Queries	87.71%mAP	371 sec.
R-PHOC (Proposed)	NO	All Ground Truth Words from test set as queries	79.83%mAP	48sec.
R-PHOC (Proposed) $>$ 5 characters	NO	All Ground Truth Words from test set as queries	86.7%mAP	48sec.

Table 5.2: Comparison with state of the art methods for QBE in GW20

QBE word spotting in segmentation-free scenario

Table 5.2 summarises the results of applying the R-PHOC network to segmentation-free QBE word spotting, compared to other methods in the state-of-the-art.

For a better comparison of the effect of using region based features we also provided the results of performing segmentation-free QBE word spotting using the original PHOCNET model. To generate these results we used the same candidates as in our approach, but in this case the PHOC representation is obtained by processing each candidate one by one. As our network is also learned to predict PHOC given a ROI of an image, the baseline result of PHOCNET is a way to provide an upper limit to our network. However this comes with a huge cost of applying the CNN for every candidate, while our network obtains the PHOC representation of all candidates in a single forward pass.

This can be verified in Table 5.2. First of all, we can see that all results using PHOC as the basis for word representation clearly outperform the rest of methods. The baseline consisting of applying PHOCNET separately to all word candidate regions achieves an accuracy of 87.71 %, but the cost of applying the CNN to all individual candidate makes this approach unfeasible in terms of computation time. When applying R-PHOC the performance decreases to 79.83 %, still outperforming all other methods by a significant margin, but reducing by a factor of almost 10 the computation time required. This result shows that the using ROI pooling permits to obtain a high accuracy while enabling the computation of the signature from all word candidates from an image by a single forward pass, which makes this approach computationally efficient. We have noticed that the down-sampling in ROI pooling has a larger effect on the small words. A similar phenomenon is observed in the case of object detection where small objects remain undetected by state-of-the-art CNN detector. Thus, we also evaluated our network on words which contain more than 5 letters and obtained an accuracy of 86.7, which is comparable to the baseline PHOCNET.

5.5 Conclusions

An efficient CNN based segmentation-free word spotting is proposed. We apply a simple pre-segmentation to generate a set of word candidates which are then passed through a convolutional neural network to predict PHOC embedding for all candidates in a single forward pass in the network. Word spotting is then performed using nearest neighbour approach. We observed that region based features obtained by integrating CNN feature maps can be used to train PHOC embeddings, thus generating an ef-

ficient scheme for segmentation-free word spotting. Though the proposed approach performs better than most of the approaches using hand crafted features, our method still performs very poorly in case of words with a low number of characters.

Chapter 6

Attention Model for Word recognition

In the previous chapters we dealt with different variants of segmentation free word spotting. This is achieved by projecting the holistic feature representation from images (queries or proposals) into a fixed dimensional attribute space. Retrieval is done as a nearest neighbour problem in that space. Normally these word embeddings are unidirectional (see Figure 6.1) in a sense that a word image can be represented as a point in these common spaces but a point in these space can not be mapped back to the string. Thus these embeddings in their current form can not be used for lexicon free word recognition.

Another plausible way to represent a word image is as a sequence of features where every component of the sequence represents a part of the image. In order to represent the image as a sequence of features one need to segment the image into its constituent parts before representing each part individually. Such a segmentation can be achieved by using supervisory signals using part annotation. However in the case of word images such annotation is extremely hard and expensive to get. This problem can be alleviated by using attention based methods, where given a set of vectors, each of them corresponding to one region of the image, different parts can be automatically attended by assigning weights to the set of feature vectors. Such models are commonly used to solve sequence-to-sequence problems like machine translation.

In this chapter we propose to investigate the use of a simple visual attention model to represent a word image as a sequence of features. Given the input image we learned

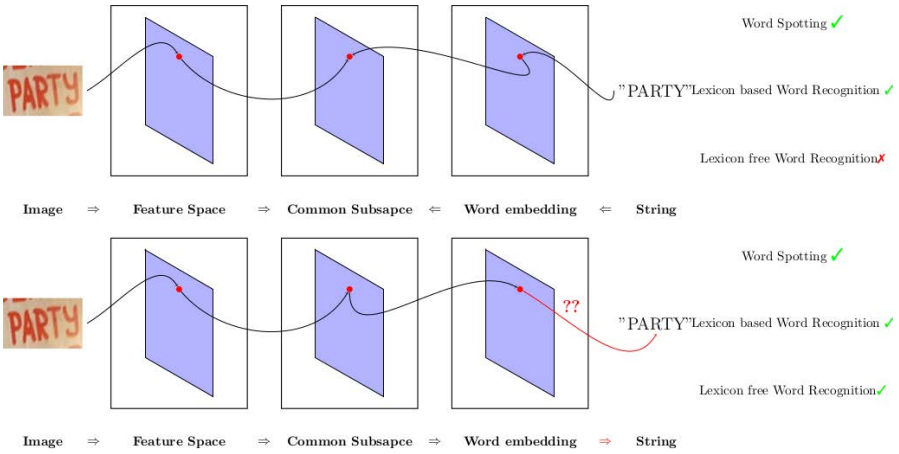


Figure 6.1: A typical common subspace embedding approach. Both image and text are mapped to the common space but a point in the common space can not be traced back to the text space.

to attend its constituent parts (characters in case of word image) and represent it by a feature vector. This feature vector can be used to infer the constituent character. Thus, given an word image our system can generate the sequence of characters contained in the word achieving lexicon-free word recognition.

An overview of the proposed recognition framework is illustrated in figure 6.2. In summary we proposed to use a LSTM-based visual attention model (based on [68]) to focus attention on relevant parts of the image at every step and infer a character present in the image. Thus, the system is able to recognize out-of-vocabulary words, although it does not need explicit character segmentation or recognition. The visual attention model can be trained using only word bounding boxes and does not need explicit character bounding boxes at training time. As the model relies on recurrent neural networks, it also learns an implicit language model from the data, which is crucial in cases such as those shown in [7]. However, the model also has the flexibility to integrate an explicit language model, which it is shown in the experiments to improve accuracy. Additionally, the output can be constrained to a fixed lexicon (when available) to work in a dictionary-based setting.

The rest of the chapter is organized as follows. In Section 6.1 we analyze the

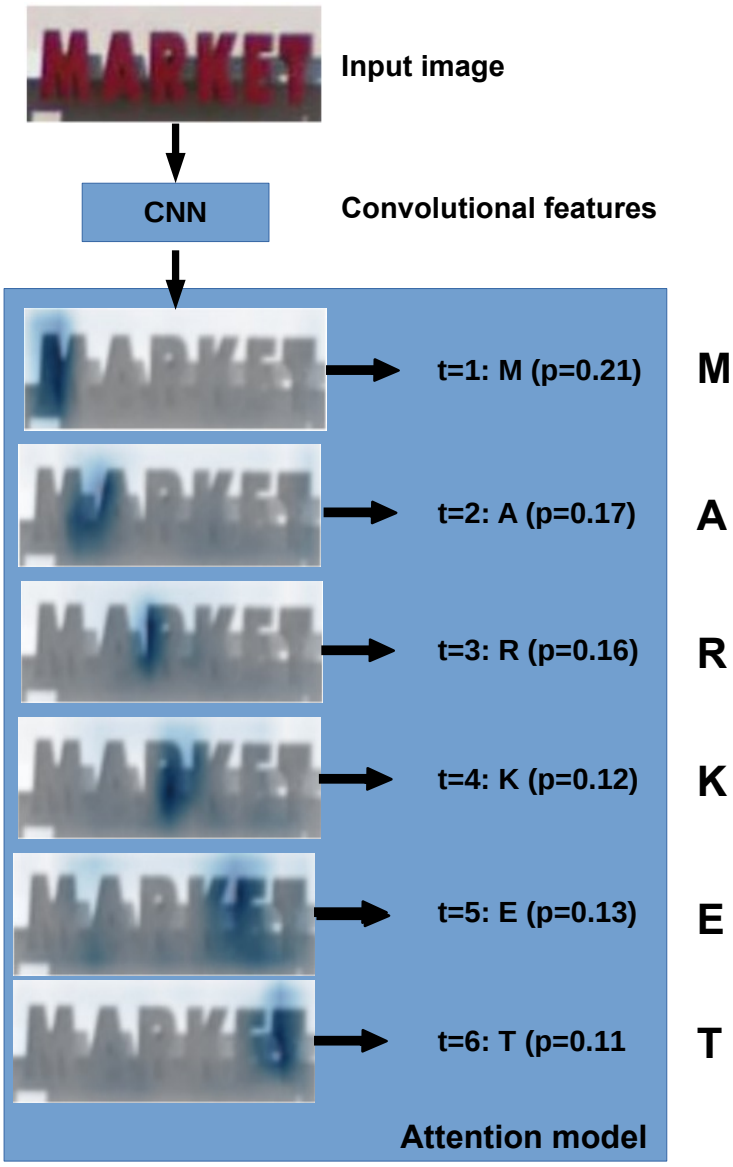


Figure 6.2: Overall scheme of the proposed recognition framework. Given a cropped word image, a set of spatially localized features are obtained using a CNN. Then, an LSTM decoder is combined with an attention model to generate the sequence of characters. At every time step the attention model weights the set of feature vectors to make the LSTM focus on a specific part of the image.

works most related to our proposed approach. Then, we present our attention-based recognition approach in Sections 6.2 and 6.3. In Section 6.4 we experimentally validate the model on a variety of standard and public benchmark datasets. We conclude in Section 6.5 with a summary of our contributions and a discussion of future research directions.

6.1 Related Work

In this section we only review work from the literature which used visual attention for robust scene text recognition and most related to our approach. A general discussion of scene text recognition has been provided in chapter 1.

Visual attention models have gained a lot of attention and have been used for machine translation [6] and image captioning [68]. In this last work the attention model is combined with an LSTM on top of CNN features. The LSTM outputs one word at every step focusing on a specific part of the image driven by the attention model. Two models of attention, hard and soft attention are proposed. In our work, we mainly follow the soft attention model, adapted to the particular case of text recognition. Attention models appear to have the potential ability to overcome some of the limitations of existing text recognition methods. They can leverage a fixed-length representation, but at the same time, they are able to guide recognition to relevant parts of the image, performing in this way a kind of implicit character segmentation.

Recently, a soft attention model has also been proposed for text recognition in the wild [32]. The main differences with our work are the following. Firstly, the success of [32] can be largely attributed to the use of Recursive Neural Network (RNN) features. They rely on the RNN features to model the dependencies between characters. Instead we use traditional CNN features and it is the visual attention model who learns to selectively attend to parts of the image and the dependencies between them. Secondly, Lee *et al.* [32] used the features from fully connected layer, while we use features from an earlier convolutional layer, thus preserving the local spatial characteristics of the image and reducing the model complexity. This also allows the model to focus on a subset of features corresponding to certain area of the image and learn the underlying inter-dependencies. Thirdly, we used LSTM instead of RNN which has been shown to learn long term dependencies better than traditional RNNs.

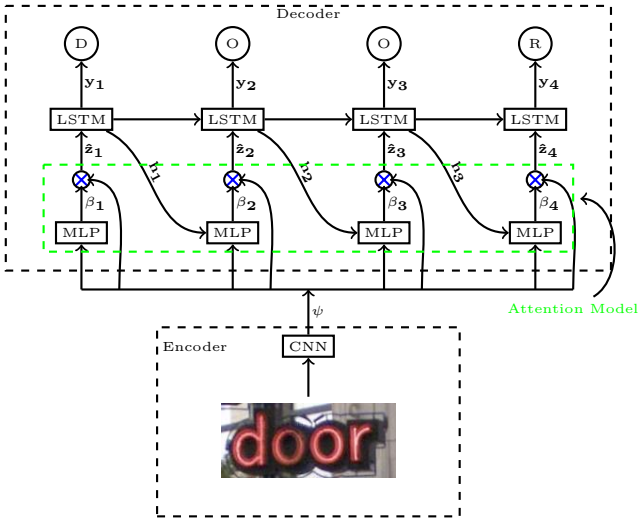


Figure 6.3: The proposed Encoder-decoder framework with attention model.

6.2 Visual attention for scene text recognition

Our recognition approach is based on an encoder-decoder framework for sequence to sequence learning. An overall scheme of the framework is illustrated in figure 6.3. The encoder takes an image of a cropped word as input and encodes this image as a sequence of convolutional features. The attention model in between the encoder and the decoder drives, at every step, the focus of attention of the decoder towards a specific part of the sequence of features. Then, an LSTM-based decoder generates a sequence of alpha-numeric symbols as output, one at every time step, terminating when a special stop symbol is output by the LSTM. Below we describe the details of each of the components of the framework.

Encoder: The encoder uses a convolutional neural network to extract a set of features from the image. Specifically, we make use of the CNN model proposed by Jaderberg et al. [26] for scene text recognition – however we do not use the fully connected layer as a fixed-length representation as it is common in previuos works. Instead, we take

the features produced by the last convolutional layer. In this way we can produce a set of feature vectors, each of them linked to a specific spatial location of the image through its corresponding receptive field. This preserves spatial information about the image and reduces model complexity. Through the attention model, the decoder is able to use this spatial information to selectively focus on the most relevant parts of the image at every step.

Thus, given an input image of a cropped word, the encoder generates a set of feature vectors:

$$\Psi = \{x_i : i = 1 \dots K\}, \tag{6.1}$$

where x_i denotes the feature vector corresponding to i^{th} part of the image. Each x_i corresponds to a spatial location in the image and contains the activations of all feature maps at that location in the last convolutional layer of the CNN.

Attention model: For the attention model, we adapt the soft attention model of [68] for image captioning, originallly introduced by [6] for neural machine translation. In [68] slightly better results are obtained using the hard version of the model that focuses, at every time step, on a single feature vector. However, we argue that, in the case of text recognition, the soft version is more appropriate since a single character will usually span more than one spatial cell of the image corresponding to each of the feature vectors. The soft version of the model can combine several feature vectors with different weights into the final representation.

As shown in figure 6.3, the attention model generates, at every time step t, a vector $\hat{z}_t$ that will be the input to the LSTM decoder. This vector $\hat{z}_t$ can be expressed as a weighted combination of the set Ψ of feature vectors x_i extracted from the image:

$$\hat{z}_t = \sum_{i=1}^{K} \beta_{t,i} x_i \tag{6.2}$$

Thus, the vector $\hat{z}_t$ encodes the relative importance of each part of the image in order to predict the next character for the underlying word. At every time step t, and for each location i a positive weight $\beta_{t,i}$ is assigned such that $\sum(\beta_i) = 1$. These weights are obtained as the softmax output of a Multi Layer Perpectron (denoted as Φ) using the set of feature vectors Ψ and the hidden state of the LSTM decoder at the

previous time step, h_{t-1}. More formally:

$$\alpha_{ti} = \Phi\left(x_i, h_{t-1}\right) \tag{6.3}$$

$$\beta_{ti} = \frac{\exp\left(\alpha_{ti}\right)}{\sum_{j=1}^{K} \exp\left(\alpha_{t,j}\right)} \tag{6.4}$$

This model is smooth and differentiable and thus it can be learned using standard back propagation.

Decoder: Our decoder is a Long Short Term Memory (LSTM) network [22] which produces one symbol from the given symbol set L, at every time step. The output of the LSTM is a vector y_t of $|L|$ character probabilities which represents the probability of emitting each of the characters in the symbol set L at time t. It depends on the output vector of the soft attention model $\hat{z}_t$, the hidden state at previous step h_{t-1} and the output of the LSTM at previous step y_{t-1}. We follow the notation introduced in [68] where the network is described by:

$$\begin{pmatrix} i_t \\ f_t \\ o_t \\ g_t \end{pmatrix} = \begin{pmatrix} \sigma \\ \sigma \\ \sigma \\ \tanh \end{pmatrix} T \begin{pmatrix} Ey_{t-1} \\ h_{t-1} \\ \hat{z}_t \end{pmatrix} \tag{6.5}$$

$$c_t = f_t \odot c_{t-1} + i_t \odot g_t \tag{6.6}$$

$$h_t = o_t \odot \tanh\left(c_t\right), \tag{6.7}$$

where T is the matrix of weights learned by the network and i_t, f_t, c_t, o_t, and h_t are the input, forget, memory, output and hidden state of the LSTM, respectively. In the above definition, $\odot$ denotes the element-wise multiplication and E is an embedding of the output character probabilities that is also learned by the network. σ and $\tanh$ denote the activation functions that are applied after the multiplication by the matrix of weights

Finally, to compute the output character probability y_t, a deep output layer is added that takes as input the character probability at the previous step, the current LSTM hidden state, and the current feature vector. The output character probability is:

$$P\left(y_t|\Psi, y_{t-1}\right) \sim \exp\left(\mathtt{L_0}\left(Ey_{t-1} + \mathtt{L_h} h_t + \mathtt{L_z}\hat{z}_t\right)\right) \tag{6.8}$$

where $\mathtt{L_0}$, $\mathtt{L_h}$ and $\mathtt{L_z}$ are the parameters of the deep output layer that are learned using back-propagation.

6.3 Inference

We use beam search over LSTM outputs to perform word inference. We first introduce the basic procedure, and then describe how we extend it to incorporate language models.

6.3.1 The basic inference procedure

Once the model is trained, we use a beam search to approximately maximize the following score function over every possible word: $\mathtt{w} = [c_1, \ldots, c_n]$:

$$S(\mathtt{w}, x) = \sum_{t=1}^{N} \log\left(P\left(c_t | c_{t-1}\right)\right), \tag{6.9}$$

where c_n is a special symbol signifying the end of a word, which immediately stops the beam search.

The beam search keeps track at every step of the top N most probable sequences of characters. For every active branch of the beam search, given the previous character of the sequence, c_{t-1}, the output character probability y_t of the LSTM is used to obtain $P\left(c_t | c_{t-1}\right)$ for all characters c_t in the symbol set L.

6.3.2 Incorporating language models

Text is a strongly contextual. There are some strict constraints imposed by the grammar of the language. For example any word in English cannot carry more than two consecutive occurrences of any alphabet letter. Leveraging such knowledge can positively impact the final recognition output. Although the LSTM implicitly learns some dependences between consecutive characters, we show that adding an explicit language model that takes into account longer dependencies gives a significant boost to recognition accuracy.

In this work we use a standard n-gram based language model during inference to leverage the language prior. The character n-gram model gives probability of a character conditioned on k previous characters, where k is a parameter of the model:

$$\Theta\left(c_k | c_{k-1}, c_{k-2} ..., c_1\right) = \frac{\#\left(c_1 c_2 ... c_{k-1}\right)}{\#\left(c_k c_k ... c_k\right)}, \tag{6.10}$$

where, $\#(c_1, \ldots\ c_n)$ is the number of occurrences of a particular substring in a training corpus.

Finally, the score function in equation 6.9 can be modified to take the n-gram language model into account as:

$$\begin{aligned} S(\mathtt{w}, x) &= \sum_{t=1}^{N} \log\left(P\left(c_t|c_{t-1}\right)\right) \\ &+ \alpha \log \Theta\left(w_t|w_{t-1}, w_{t-2}..., w_1\right) \end{aligned} \tag{6.11}$$

At every step we fix the parameter k of the language model to the number of previously generated characters in order to take into account the longest possible sequence.

6.3.3 Lexicon-based inference

Although our method is originally designed for unconstrained text recognition, it can also leverage a lexicon whenever available. The use of a lexicon D can be integrated by modifying the beam search so that all active sequences that do not correspond to any valid word are automatically removed from the beam. This can be expressed by modifying equation 6.9 as:

$$S(\mathtt{w}, x) = \begin{cases} \sum_{t=1}^{N} \log\left(P\left(c_t|c_{t-1}\right)\right) & \text{if } \mathtt{w} \in \text{D} \\ \quad + \alpha \log \Theta\left(w_t|w_{t-1}, \ldots, w_1\right) & \\ -\infty & \text{if} \mathtt{w} \notin D \end{cases} \tag{6.12}$$

This can be efficiently implemented by storing the lexicon in a trie structure and automatically removing from the beam search any alternative that do not correspond to any partial branch of the trie.

6.4 Experimental Results

In this section we report on experiments carried out to validate the proposed model for unconstrained scene text recognition. We use the common protocols as described in Chapter 2 for recognition.

Implementation details: The CNN encoder used in this work is the Dictnet model by Jaderberg *et al.* [25]. Their deep convolutional network consists of four convolutional layers and two fully connected layers. In this work we used features from the last convolutional layer. Thus, the feature map used is of size 4×13 and therefore, the LSTM takes input in the form of 52×512.

For lexicon-based recognition when we do not use the lexicon-based inference explained in section 6.3.3. Instead, we take the output of unconstrained recognition and find the closest word in the lexicon using the Levenshtein edit distance. For lexicon-based inference in unsconstrained datasets (SVT and ICDAR'03) we use the 90k-words lexicon provided by Jaderberg *et al.* in [25]. The explicit language model is also learned using this 90k word lexicon.

The parameter α (see equation 6.11) to weight the language model with respect to LSTM character probability is empirically established. In our experiments we found the best results with α between 0.25 to 0.3

6.4.1 Baseline performance analysis

In this section we analyze the impact on performance of all the components of the proposed model. We start with a baseline that consists of a simple one layer LSTM network as decoder, without any attention or explicit language model. As we are interested mainly in the impact of the attention model, we use a simple version in which CNN features from the encoder are fed to the LSTM only at the first time step. At every step the output character is determined based on the output of the previous step and the previous hidden state.

In an effort to evaluate each of our contributions, we trained the baseline system and our model with exactly the same training data. For this purpose we randomly sampled one million training samples from the Synth90k [25] dataset. For validation we used 300,000 samples randomly taken from the same synth90K dataset.

We present the results for each of the component of the framework as described above in Table 6.1. The attention model outperforms the baseline by a significant margin (around 7%). Also these results confirm the advantage of using an explicit language model in addition to the implicit conditional character probabilities learned by the LSTM model. Using the language model improves accuracy in another 7%. We also see that further constraining the inference wih a dictionary does not improve the result much, probably because the language model is learned from the same 90K

dictionary proposed by Jaderberg *et al.* in [25].

In comparison with other related works on unconstrained text recognition, it is noteworthy that with only one million training samples our complete framework can learn a better model than Jaderberg *et al.* [23] and obtain results that are close to other state-of-the-art methods that are using the whole 9 million sample training dataset (see table 7.4).

Methods	**SVT**
Baseline (LSTM-no attention)	61.7
Proposed (LSTM + attention model)	68.16
Proposed (LSTM + attention model + LM)	75.57
Proposed (LSTM + attention model+LM+dict)	76.04

Table 6.1: Impact of the different components of our framework with respect to the baseline. We compare the baseline (LSTM with no attention model) with all the variants of the proposed method, incrementally: using only the attention model (section 6.2, integrating also the explicit language model (section 6.3.2, and contraining the inference to a lexicon (section 6.3.3).

6.4.2 Comparison with state of the art

In this section we will compare our result with other related works on scene text recognition. The results of this comparison are shown in table 6.2. First, we will discuss results on unconstrained text recognition which is the main focus of our work. Then, we will analyze results for lexicon-based recognition.

Unconstrained text recogntion: apart from our method Jaderberg *et al.* [23], Lee *et al.* [32] and Bissaccco *et al.* [7] are the only methods which are capable of performing totally unconstrained recognition of scene text. Among these methods, our visual attention based model performs significantly better than Bissacco *et al.* [7] and Jaderberg *et al.* [23] in both SVT and ICDAR'03 datasets. Our model also performs as good as Lee *et al.* [32] in SVT dataset and outperforms them by 3% in ICDAR'03 dataset, which is significant given the high recognition rates.

If we further compare our model with that of Lee *et al.* [32], that also uses different variants of RNN architectures and an attention model on top of CNN features, we find that they use recursive CNN features. They report that this gives an 8% increase in

	Methods	SVT-50	SVT	ICDAR'03-50	ICDAR'03-full	ICDAR'03
Lexicon-based recognition	ABBY Baseline ABBYY [65]	35.0	-	56.0	55.0	-
	Wang *et al.* [65]	57.0	-	76.0	62.0	-
	Mishra *et al.* [38]	73.2	-	81.8	67.8	-
	Novikova *et al.* [40]	72.9	-	82.8	-	-
	Wang *et al.* [66]	70.0	-	90.0	84.0	-
	Goel *et al.* [17]	77.3	-	89.7	-	-
	Alsharif and Pineau [5]	74.3	-	93.1	88.6	-
	Almazan *et al.* [4]	89.2	-	-	-	-
	Lee *et al.* [33]	80.0	-	88.0	76.0	-
	Yao *et al.* [69]	75.9	-	88.5	80.3	-
	Rodriguez-Serrano *et al.* [46]	70.0	-	-	-	-
	Jaderberg *et al.* [26]	86.1	-	96.2	91.5	-
	Su and Lu*et al.* []	83.0	-	92.0	82.0	-
	Gordo *et al.* [18]	90.7	-	-	-	-
	*DICT Jaderberg et al. [25]	95.4	80.7	98.7	98.6	93.1
Unconstrained	Bissacco *et al.* [7]	90.4	78.0	-	-	-
	Jaderberget al. [23]	93.2	71.7	97.8	97.0	89.6
	Lee *et al.* [32]	96.3	80.7	97.9	97.0	88.7
	Proposed (LSTM + attention model)	91.7	75.1	93.4	91.0	89.3
	Proposed (LSTM + attention model + LM)	95.2	80.4	95.7	94.1	92.6
	Proposed (LSTM + attention model+LM+dict)	95.4	-	96.2	95.7	-

Table 6.2: Scene text recognition accuracy. "50"and "Full" denote the lexicon size used for constrained text recognition as defined in [65]. Results are divided into lexicon-based and unconstrained (lexicon-free) approaches. *DICT [25] is not lexicon-free due to incorporating ground-truth labels during training.

accuracy over the baseline. This success is due to the recurrent nature of the CNN feature which implicitly model the conditional probability of character sequences.using recursive CNN performs better than the traditional convolutional feature. However, the RNN architecture they use improves only 4% over the baseline. In contrast our method rely on traditional CNN features (which can possibly encodes the presence of individual characters as shown in [23] from lower convolutional layer preserving local spatial characteristics, which reduces the complexity of the model. In addition, as reported in table 6.1, our combination of LSTM and soft attention model achieves a much larger margin, 14%, over the baseline. Theses results show that a combination of local convolutional features using the context based attention attention performs better or comparable to the previous state-of-the- art results.

Lexicon-based recognition For SVT-50 we can observe that our method obtain a similar result than the best of the methods [25] specifically designed to work in a lexicon-based scenario. Comparing with methods for unsconstrained text recognition, only the method of Lee*et al.* [32] outperforms our best setting. But as we have already discussed, part of this better performance can be explained by the use of the more complex recursive CNN features.

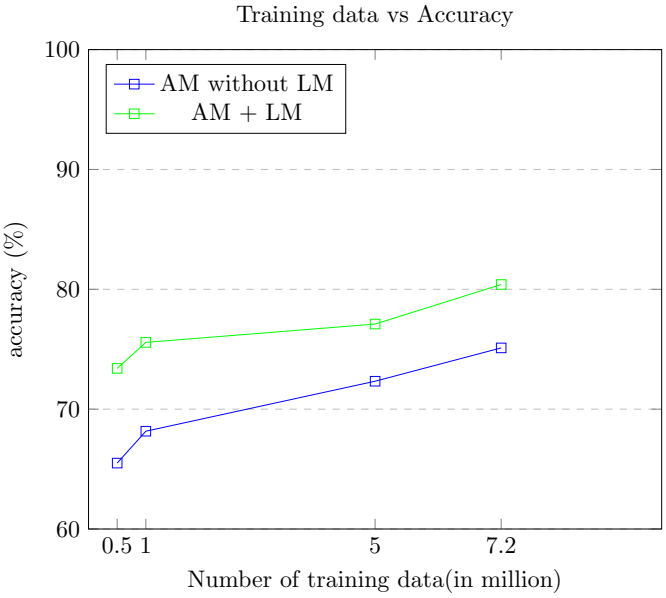


Figure 6.4: Effect of increasing size of the training set data with (green) and without the language model (blue).

Concerning ICDAR'03-50 and ICDAR'03-full, our results, although do not beat current state of the art are very competitive and comparable to the best performing methods.

6.4.3 Discussion

Effect of language modeling: The LSTM with attention model implicitly learns some character sequence model as the output at every step is conditioned by the previous character and also the attention is conditioned by the previous hidden state. However, the baseline analysis in 6.1 shows that apart from this implicit character sequence model, adding an explicit language model helps to improve the performance. This finding is inline with the works of Bissacco *et al.* [7], who also uses a static n-gram language model learned by a huge in-house corpus. In our case we used the 90k dictionary as corpus to learn the language probability. To efficiently calculate the lan-

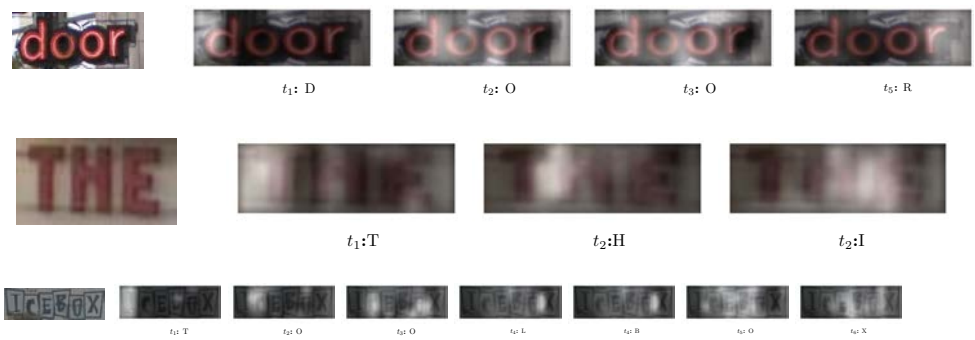


Figure 6.5: Some results obtained with the proposed model.

guage probability we make use of a trie [13] data structure. In Bissacco *et al.* , authors used a n-gram model with $n = 8$ and additionally they also used word probabilities to re-rank the words, for which they learn additional word level language model. The authors analyzed the impact of the language model and they report a reduction of the word error rate by approximately 39.2%. In contrast, our language model is small and improves the recognition accuracy around 4%, a result which is very consistent across the different datasets – see Table 7.4.

Effect of Training Data As the famous quote of Google Research Director Peter Norvig "We do not have better algorithms. We just have more data". The effectiveness of data can not be denied in today's machine learning systems. As the real datasets are too small for learning deep networks, Jaderberg *et al.* [25] proposed to learn deep models using only a synthetic dataset. This dataset has 9 million cropped word images. In order to analyze the behavior of our attention models with increasing data, we plot in Figure 6.4 the recognition accuracy in SVT dataset with respect to the number of training samples. As expected, accuracy increased with the training size.

We also analyzed the effect of language model with an increasing number of training samples. Initially with 0.5 million training data the improvement when adding the language model is around 8%. However as we increase the training data size this reliance on language model reduces to around 4%. One reason for this can be that with more data the implicit language model learned by the LSTM is more powerful. The use of an LSTM capable of modeling character sequences can also be an additional reason that explains why the improvement obtained by Bissacco *et al.* using the

language model is greater.

Qualitative Results As the features used to encode the image correspond to different parts of the image, we can visualize the attention at every time step of the network. In Figure 6.5 we can see those visualizations. We can observe that every time step a part of the image is attended, in a way that roughly mimics the natural reading order. We can also notice that the attention model is performing some kind of implicit character segmentation.

6.5 Conclusions

In this paper we proposed an LSTM-based visual attention model for scene text recognition. The model uses convolutional features from a standard CNN as input to an LSTM network that selectively attends to parts of the image at each time step in order to recognize words without resorting to a fixed lexicon. We also propose a modified beam search strategy that is able to incorporate weak language models (n-grams) to improve recognition accuracy. Experimental results demonstrate that our approach outperforms or performs comparably to state-of-the-art approaches that use lexicons to constrain inferred output words. Experimental results shows that context plays a important part in case of real data, thus using a explicit language model always helps to improve the result.

Chapter 7

SPHOC: Attention based Spatial Histogram of Characters for text recognition

Till now in this thesis two different types of representation have been investigated. First in chapters 3,4 and 5 a fixed length representation to project word images into a attribute space is introduced. Then in chapter 6 a representation based on sequence of features is investigated. It can be observed that a fixed length representation is more suitable for retrieval setup while a sequence based variable length representation works better in case of lexicon-free word recognition by using sequence-to-sequence models.

In this chapter our goal is to reduce this gap between different problem domains and come up with a unified representation based on the PHOC embedding, which can be used for both word spotting and lexicon-free recognition.

Thus, we introduce some modifications to PHOC descriptor introduced in chapter 3, that will allow a much more easy and accurate decoding of a word image.

In order to approach this problem we make the observation that PHOC can be decoded uniquely back to the string if the exact number of characters l in the word is known. In such a case, level l will contain l histograms of characters, each of them corresponding to one single character in the word. Following this observation we

propose two modifications to the PHOC representation and network architecture.

First of all, we add a length estimator module to the network architecture that infers the length of the word (as a number of characters) and can be trained jointly with the PHOC representation of the word.

Secondly, we extend the number of levels in the original PHOC representation (5 levels) up to 10 levels, since most words have more than 5 characters. This leads to an increase in the output size which is computationally demanding as current networks realizes the output of PHOC as a single fully connected layer with sigmoid activation. Thus, the layer size is in the order of ($\hat{n}$), $\hat{n} = \sum_i^n i$, being n is the number of levels. In this way, we observe that current methods compute PHOC as a holistic embedding by estimating the full embedding from features computed from the whole word image. Though such an approach is simple and does not need any feature selection mechanism (to select features from different regions) it needs a huge number of parameters. To alleviate this we propose to extend the attention based feature selection idea proposed in chapter 6. Instead of computing the whole output embedding from a holistic feature representation from the entire word image, we propose to attend different regions corresponding to every level and compute an independent histogram of characters for every region using only features from that region in the image. Thus, the final PHOC representation can be obtained by concatenating the individual histograms of characters. This approach makes the number of parameters much lower as the output embedding is just one histogram of characters per region. To realize this, a feature selection mechanism is needed to generate features corresponding to regions. For this purpose we propose to use a novel multi-head attention mechanism to generate different feature representations for different pyramidal regions of the word image. With the attention module we are able to generate different feature representations for each level of the pyramidal representation. In this way, we can use one fully connected layer of only 36 output units for every particular histogram of characters. The final PHOC representation is obtained by concatenating the output of each independent fully connected layer for every level (see figure 7.1).

This new architecture for computing an extended PHOC representation permits also to propose a decoding procedure that takes as input the estimated word length and the PHOC image representation and returns the word string corresponding to the image, which permits to use PHOC both for recognition and spotting.

The rest of the chapter is organized as follows: In section 7.1, we describe different components of our methodology. Section 7.2, explains different experiments performed for experimental validation of the approach including qualitative and quan-

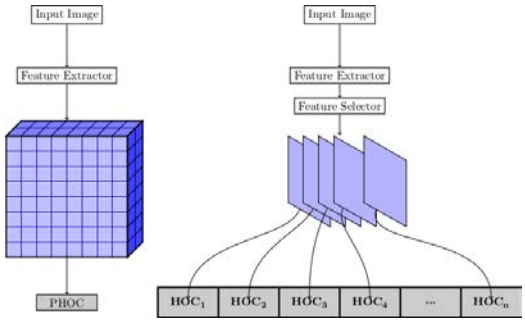


Figure 7.1: Holistic feature based approach (on the left) vs feature section based approach (on the right)

titative results followed by one section each for discussion and conclusion.

7.1 Methodology

In this section we describe the three main components of our work. First, in section 7.1.1 we explain the extension of the PHOC representation adding more levels in order to make it more discriminant for recognition. Then, in section 7.1.2 we introduce the architecture of the encoder that permits to obtain the new PHOC representation in an efficient way. Finally, in section 7.1.3 we describe how the PHOC representations can be decoded into the corresponding string.

Figure 7.2 shows the overall architecture that we propose for the encoder in oder to obtain the extended PHOC representation. In a nutshell, our word spotting and recognition pipeline works as follows: first a convolutional feature map is extracted from the input image. This feature map is used to estimate the length (as the number of characters) of the word. Additionally, we also create a new feature map encoding the position of every pixel in the original feature map. The concatenation of the original feature map, the estimation of the length and the encoding of the position is used as input to the attention mechanism that predicts a set of weights that permit to select relevant features from the feature map for every region at every level of the pyramid. Then, every different feature map obtained as output of the attention module is used to estimate each of the histograms of characters at the corresponding level

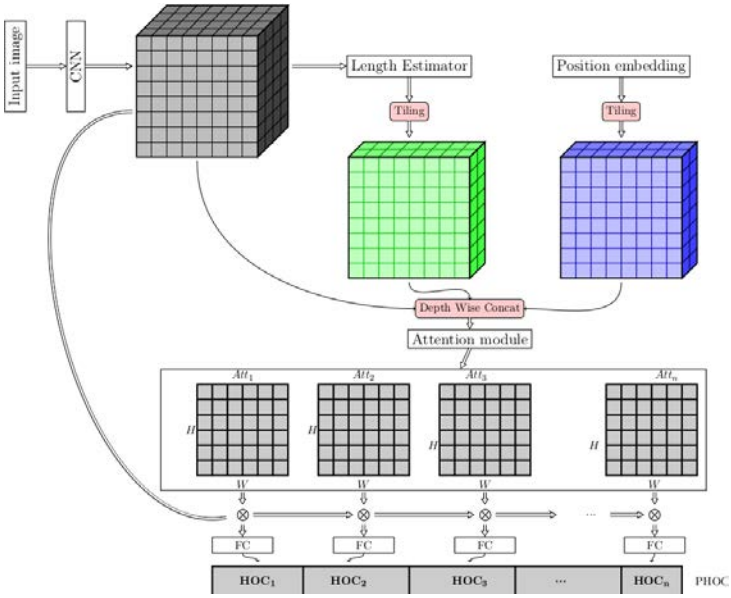


Figure 7.2: Overall Architecture: First a CNN feature map is obtained then CNN feature is enriched with length and position. From this enriched feature map n different attention is computed, producing n different representation of the image, from each of these representation a histogram of character is computed by using one fully connected layer, final PHOC is obtained by concatenation of n histogram of characters

of the pyramid. The concatenation of all individual histograms gives the final PHOC representation.

Further, the PHOC representation and the estimated length will be used by the decoder to transform the estimated PHOC into its corresponding string.

7.1.1 Extension of the PHOC representation

Though PHOC gives state-of-the-art result in the case of word spotting it can not be used for unconstrained word recognition (where a vocabulary is not available) due to the fact that the representation is unidirectional (see figure 6.1). PHOC can be directly

computed given a string or can be estimated by a learned model given a word image, but given a PHOC vector (obtained either from a string or from a word image) it can not be decoded back to its corresponding string.

However one interesting observation about PHOC embedding is that, for a particular level k, the representation is obtained by concatenation of k histograms of characters. These histograms corresponds to k different sub-regions of the image (or sub-strings of the word). Thus the histogram at level k can be formally described as $H_k = U\{x_j\}$, where x_j, is the histogram for j^th sub-region (string). Now consider the case when $k = l$, where l is the number of characters in the word. This makes every sub-string (or region) containing only one character, and therefore, every histogram will have a single peak corresponding to that particular character. In this case PHOC can be decoded to its corresponding string by just concatenation of the characters having the maximum value in the histogram at every sub-region. Thus, considering that we can estimate the length of a word (see next section), we could easily decode its PHOC given that we have enough number of levels in the PHOC representation. This implies that the original PHOC representation that includes only 5 levels might not be enough in the general case as there are many words longer than 5 characters that could not be decoded with the previous assumption.

Thus, we propose to extend the original PHOC representation adding extra levels. The final number of levels should keep some compromise between the ability of decoding any word and the increased computational complexity of longer representations. In order to arrive at a reasonable choice we have computed the cumulative percentage of unique words for each dataset. We can see that number of words with more than 10 characters is quite rare (less than 2%) in all the datsets. Thus, we have fixed the total number of levels to 10 that permits to decode most words with a reasonable computational cost.

7.1.2 Encoder

The encoder for our model has two requirements. The primary goal is to embed the image into the PHOC space, However, we also need to estimate the number of characters in the word in order to be able to decode the PHOC to the string as discussed in the previous section.

In our encoder we use a deep convolutional network which takes the whole word image as input and produces as output the PHOC representation as well as an es-

timation of the word length. There have been several attempts to train the PHOC embedding both in the case of handwriting and lexicon based scene text recognition in []. All these networks followed a common pipeline to obtain the PHOC embedding. They used a series of convolutional plus max pooling layers to obtain a feature map, followed by several fully connected layers. The size of the last fully connected layer is equal to the size of the PHOC embedding size and is followed by a sigmoid activation to produce multiple labels for an image. However, this type of network structure has an important drawback in our case. For every increased level in the representation, the number of output units needs to be increased by $k \times voc$, where k is the level number and voc is the alphabet size. This can lead to very big models if the number of levels in PHOC needs to be significantly increased.

To alleviate this we propose a novel architecture by leveraging the fact that PHOC is a concatenation of histograms of different regions of the word image. Thus, instead of generating the whole PHOC representation in a single step, we propose to generate one independent histogram per region of the word. Later, all these histograms are concatenated to obtain the complete PHOC representation. For that, we propose to use an attention mechanism to implicitly segmenting the image into regions. In a nutshell we learn attention maps over the image which are used as weights to generate each of the histograms. The weights of a particular region of the image can be interpreted as the relevance of a that region in order to predict a given histogram.

Thus, our encoder has three different components: the length estimator, the attention map generator and the histogram generator. We discuss each of these components below. The overall architecture of the model is given in figure 7.2. The whole network is trained end-to-end by using multi-task loss one for histogram each and the length.

Length Estimator:

Estimating the length is important to correctly decode the PHOC to its corresponding string. Since we want to compute the length and the PHOC embedding in a single end-to-end step, we use the same convolutional feature map for both tasks. We model the length estimation as a regression problem and we explore different design choices to obtain the length as output.

The most straightforward approach to represent the length information as output would be to use one output unit giving the absolute length. Such network could be learned by minimizing the absolute distance or the euclidean distance between the predicted length and the actual length. However, in this way, the prediction could be

very noisy.

Thus, we follow a different approach and represent the length as a n dimensional vector V_l, where n is the maximum allowed length of any word (10 in practice, as the number of levels of the PHOC representation). For a given word W of length k it is defined as:

$$V_l(W) = \langle l_1, l_2 ... l_i ... l_n \rangle, \text{ where } l_i = 1 \text{ if } i <= k, 0 \text{ otherwise}$$

Given this vector representation the obvious choice is to minimize the sigmoid cross entropy loss between the estimated vector $\hat{V}_l$ and V_l. We tested with different configurations for the layers used to predict the length given the feature map generated by the CNN. In particular we explored the following settings: 1) Average pooling followed by FC layer 2) Enriching the feature map with with positional information (see below for more details) followed by FC layers. In our experiments we have observed negligible difference in the accuracy of the estimated length among the three configurations. However enriching the CNN features with positional information gives a slightly better accuracy and thus, this is the option we have taken.

Attention map generator:

The goal is to select relevant features from the convolutional feature map for each PHOC level that permit to estimate the histogram of characters of that level independently of the other levels. This is realized by generating a weight matrix $\mathbb{W}$ over each location of the image, i.e each point in the feature map. Since our goal is to select the features corresponding to every level of pyramid the weights must be dependent on the position and the length of the word. Thus we define our feature selection function as follows:

$$\mathbb{W} = \Psi(C_f, C_l, C_{pos}) \tag{7.1}$$

where C_f is the convolutional feature map, C_l is the estimated length information and C_{pos} is an embedding of each position of the feature map. The length estimation C_l is obtained by tiling the vector V_l described above along the feature map. In addition to the estimated length, we also enrich the feature vector with positional information C_{pos} encoding the position of every point in the feature map. We embed the position of every point as a binary vector of dimension $H + W$, where H and W are respectively height and width of input feature map, given by:

$$
\begin{aligned}
C_r &= \langle r_i \rangle, \text{where } p_i = 1, \text{ if } i \text{ equal to row position of the grid} \\
C_c &= \langle c_i \rangle, \text{where } c_i = 1, \text{ if } i \text{ equal to column position of the grid} \\
C_{pos} &= \langle r_i, c_i \rangle
\end{aligned} \tag{7.2}
$$

The final input to the attention module is the element-wise concatenation of C_f, C_l and C_pos. Ψ is computed by a 1×1 convolutional layer with n channels, where n is the number of pyramidal levels for which the weight distribution is needed. This attention layer is added on top of the convolutional layers.

Histogram Generator:

Finally n feature vectors are computed by doing the weighted average of the convolutional feature map using the n attention map. This n feature vectors are used to generate n histograms of characters representing each level of the PHOC pyramid. The concatenation of all these individual histograms gives the final PHOC embedding for the image. The sigmoid cross entropy loss between the groundtruth PHOC computed from the string and the estimated PHOC is minimized using stochastic gradient descent to train the network.

The above network has two objective functions, one for the length estimation and another for the PHOC estimation. The network is trained end-to-end by using a multi-task loss by weighted average of the two individual losses.

7.1.3 Decoder

The encoder as mentioned in earlier section produces n histograms (also called sub-PHOC in this article) and length encoding l_k for an input word image. The goal of the decoder is to predict the string given the word embedding and the length. Given the length information one can find the level at which sub-phocs contain exactly one character each. Thus the simplest way to predict the word is to use these sub-PHOCs according to the length of the string and generate one character per sub-phoc(region) and concatenate to get the recognized word. This approach to combine the estimated length and the PHOC assumes that estimated PHOC and length is always in sync. However this can not be guaranteed as scene text and handwriting produces challenging and diverse types of data. Moreover this naive decoding acts independently on

every level of PHOC and therefore, it does not use any language statistics. Nevertheless we compute results using this naive approach that can serve as a baseline and help in understanding the effect of language model and presence of lexicon in our approach.

To overcome the limitation of naive decoding we propose a variant of beam search to decode the embedding which we call beam search based decoding. In the following section first beam search based decoding is explained which is further extended to use language model and predefined lexicon.

The basic inference procedure

We use a beam search to approximately maximize the following score function over every possible word: $\mathtt{w} = [c_1, \ldots, c_n]$:

$$S(\mathtt{w}, x) = \sum_{t=1}^{N} \log\left(P\left(c_t\right)\right), \tag{7.3}$$

where N is the length of the string, as given by the encoder.$P\left(c_t\right)$ gives the score of t^{th} character. Given a level of pyramid N (determined by the estimated length), corresponding N sub-phocs can be found. Every sub-phoc is interpreted as score function $P\left(c\right)$), where c is a symbol in symbol set L.

The beam search keeps track at every step of the top K most probable sequences of characters.

Varying estimated length The above procedure starts with one level of sub-phocs equal to the estimated length of the word image. Thus, the prediction of this method will highly depend on the success of the estimated length. In experiments we have seen that in around 10 % cases the estimated length varies by one character (see experimets section). Hence exploration of other levels is necessary for successful recognition. We extended the basic beam search procedure to incorporate information from multiple level by varying the estimated length or level given by N by +/- 1. Thus instead of starting with K top scoring root nodes from one level to explore further in the beam search tree we start with K top scoring nodes from 3 adjacent levels and then explore further. The exploration for a given branch ends once all sub-phocs are explored for a given level.

Incorporating language models and Lexicon

Text is a strongly contextual. There are some strict constraints imposed by the grammar of the language. For example any word in English cannot carry more than two consecutive occurrences of any alphabet letter. Leveraging such knowledge can positively impact the final recognition output. Unlike the LSTM based model where some language model is implicitly learned by models our model is only relying on the shape of characters thus to include some dependencies between consecutive characters, we introduced an explicit language model. In results, we will show that this gives a significant boost to recognition accuracy.

In this work we use a standard n-gram based language model during inference to leverage the language prior. The character n-gram model gives probability of a character conditioned on k previous characters, where k is a parameter of the model:

$$\Theta\left(c_k|c_{k-1}, c_{k-2}..., c_1\right) = \frac{\#\left(c_1 c_2 ... c_{k-1}\right)}{\#\left(c_k c_k ... c_k\right)}, \tag{7.4}$$

where, $\#(c_1, \dots\ c_n)$ is the number of occurrences of a particular substring in a training corpus.

Finally, the score function in equation 7.3 can be modified to take the n-gram language model into account as:

$$\begin{aligned} S\left(\mathtt{w}, x\right) &= \sum_{t=1}^{N} \log\left(P\left(c_t\right)\right) \\ &+ \alpha \log \Theta\left(w_t|w_{t-1}, w_{t-2}..., w_1\right) \end{aligned} \tag{7.5}$$

At every step we fix the parameter k of the language model to the number of previously generated characters in order to take into account the longest possible sequence.

Although our method is originally designed for unconstrained text recognition, it can also leverage a lexicon whenever available. The use of a lexicon D can be integrated by modifying the beam search so that all active sequences that do not correspond to any valid word are automatically removed from the beam.

This can be efficiently implemented by storing the lexicon in a trie structure and automatically removing from the beam search any alternative that do not correspond to any partial branch of the trie.

7.2 Experiments and Results

In this section we present our experimental validation of the proposed approach.We evaluate the performance of the proposed method using standard datasets and protocols as described in chapter 2.

Before going to the details of results we first describe in section 7.2.1 the architectural details and design considerations which will help in reproducing the results. We have also discussed the training steps used to arrive at the output models. In below section we discuss about these issues before explaining the results.

7.2.1 Training and network details

We use inception style network to extract features from cropped word images. We use fixed size images at input. For scene text we use 32×80 and for handwritten images we use 70×150 sized images. Our feature extractor used 3 inception blocks and a pooling architecture similar to [58].

The feature map is fed as input to two branches: one is used for length estimation and other is used for PHOC computation. For length estimator we experimented with different alternatives as mentioned in 7.1.2.

Feature selection module is realized by 1×1 convolutional layer with k filters, where k is the number of feature maps required. Thus k is equal to the number of histogram of characters in the PHOC embedding. In our experiments we used PHOC till level 10 thus we need $1 + 2 + 3 + 4 + 5 + 6 + 7 + 8 + 9 + 10 = 55$ filters for the feature selection layer.

Finally, average pooling is done after feature selection for each weighted feature maps. These weighted feature maps are then used to produce one histogram of characters each.

We use synthetic data released by Jaderberg *et al.* in [32] for training in the case of scene text recognition. We also observed that our model need less amount of data than many deep learning models. In section 7.4.2, we have plotted the amount of data requirements against accuracy for further insight.

In case of handwriting we use official partition of IAM dataset for training. The same model is used for both recognition and spotting.

7.2.2 Results

In this section we present our experimental validation of the proposed approach. One of the crucial factors upon which our approach rely on is how accurate the length estimator performs. Thus in the following subsection we first analyze the performance of our length estimator for different datasets. Then, we show results for handwritten word spotting and for word recognition in two different contexts, handwritten text and scene text. Finally we analyze qualitatively the impact of the attention module in our framework.

Length estimation

Although this problem has never been addressed before, computation of accuracy for length estimation is straightforward. For every word we compare the number of characters in the ground-truth transcription with the length estimated by the network and compute the accuracy as the percentage of correct matches. As in decoding we use the histograms of characters not only at the level of the estimated length, but also at adjacent levels we also compute the accuracy when the estimated length differs from the ground-truth length by one. This is marked as "accuracy at 1" in table 7.1.

We show results in table 7.1 for two different datasets, representative of the two domains: the IAM dataset for handwritten documents and the SVT dataset for scene text. It can be observed that the estimation of the length is pretty accurate for both handwritten and scene text. When we allow for a difference of only one character between the estimation and the ground-truth the accuracy is very close to 100%. Therefore we can conclude that the estimation of the length is accurate enough to be used in the decoding of PHOC into word transcriptions.

Dataset	Accuracy	Accuracy at 1
IAM	96.43	99.45
SVT	95.4	98.6

Table 7.1: Performance of the length estimator

Method	CER	WER
Voigtlaender *et al.*	3.5	9.2
Doetsch *et al.*	4.7	12.2
Pham *et al.*	5.1	13.6
SPHOC(naive decoding)	8.4	13.6
SPHOC(Beam search decoding)	4.5	11.6
SPHOC(Beam search decoding + LM)	2.1	9.5

Table 7.2: Comparison with state of the art methods for word recognition in IAM dataset

Word Spotting for Handwritten text:

Though our focus in this work is not word spotting, as the efficacy of PHOC has already been shown in case of word spotting. Still for the sake of completion and to propose a unified embedding for all subtasks in this area we report results of word spotting in IAM and compared it with state of the art results.

Word Recognition for handwritten text:

Once we confirm that our length estimators can estimate the word length reliably, we use the length information for word recognition. We modified the beam search algorithm to start with three hypothesis based on three different length i.e the estimated length, estimated length + 1 and estimated length -1. We used basic n-gram language model as described in 7.1.3 and computes K most probable word, where$K$ is the size of the beam. We compute the character error rate (CER) and Word Error Rate (WER) for comparison with state of the art works.

Lexicon-free Word recognition in Scene Text

In case of scene text word recognition is more studied problem, there is regular competition organized within the framework of robust reading competition. Though there are multiple datasets proposed, still SVT is used to compare different algorithms. Most successful algorithms used LSTM for lexicon free word recognition. There are only few works which do not uses LSTM for end-to-end recognition. Among them first is dictnet proposed by Jaderberg *et al.* in [25] where they used a classification ap-

Method	QBE	QBS
TPP-PHOCNET	84.80	92.97
PHOCNET	85.50	92.38
Deep Feature Embedding	84.24	91.58
Attribute SVM	55.73	73.72
Triplet CNN	81.58	89.48
BLSTM	-	78.00
SPHOC(ours)	84.1	91.2

Table 7.3: Comparison with state of the art methods for QBE and QBS in IAM

proach to classify word image among 90K output classes, thus the output is restricted to 90K dictionary. Second approach which used CNN for word recognition is also used PHOC as the final output, however they didn't provide any means to decode the PHOC to word. Consequently this approach can not provide lexicon free recognition. Finally recently ... *et al.* proposed a CNN based approach and used CTC algorithm to decode convolutional sequence into a sequence of characters. Though they achieved state of the art in few datasets but the biggest limitation of this algorithm the sequence depends on predefined size. They have shown that the result depends on this size.

7.3 Qualitative Results

In this section we visualize some of the aspects of our results first in section 7.3.1 we visualized the effects of feature selection mechanism and later in section 7.3.2, we give some qualitative results for different scenarios.

7.3.1 Attention Visualisation

In this section we go into the details of feature selection mechanism by visualizing the weight maps. In figure 7.3 we overlaid the attention maps over the input image. This can be observed that the weight distribution matches with level of pyramid in PHOC, thus the histogram generator gets the input features corresponding to the concerned regions.

	Methods	SVT-50	SVT	ICDAR'03-50	ICDAR'03-full	ICDAR'03
Lexicon-based	ABBY Baseline ABBYY [65]	35.0	-	56.0	55.0	-
	Wang *et al.* [65]	57.0	-	76.0	62.0	-
	Mishra *et al.* [38]	73.2	-	81.8	67.8	-
	Novikova *et al.* [40]	72.9	-	82.8	-	-
	Wang *et al.* [66]	70.0	-	90.0	84.0	-
	Goel *et al.* [17]	77.3	-	89.7	-	-
	Alsharif and Pineau [5]	74.3	-	93.1	88.6	-
	Almazan *et al.* [4]	89.2	-	-	-	-
	Lee *et al.* [33]	80.0	-	88.0	76.0	-
	Yao *et al.* [69]	75.9	-	88.5	80.3	-
	Rodriguez-Serrano *et al.* [46]	70.0	-	-	-	-
	Jaderberg *et al.* [24]	86.1	-	96.2	91.5	-
	Su and Lu*et al.* []	83.0	-	92.0	82.0	-
	Gordo *et al.* [18]	90.7	-	-	-	-
	*DICT Jaderberg et al. [25]	95.4	80.7	98.7	98.6	93.1
Unconstrained	Bissacco *et al.* [7]	90.4	78.0	-	-	-
	Jaderberget al. [23]	93.2	71.7	97.8	97.0	89.6
	Lee *et al.* [32]	96.3	80.7	97.9	97.0	88.7
	Shi *et al.* []	96.4	80.8	98.7	97.6	89.4
	Shi *et al.* []	95.5	81.9	98.3	96.2	90.1
	Cheng *et al.* []	97.1	85.9	99.2	97.3	94.2
	Cheng *et al.* (baseline) []	97.1	81.5	99.2	97.3	94.2
	Shi *et al.* (baseline) []	97.1	82.2	99.2	97.3	94.2
	Proposed(SPHOC + Naive decoding)	**84.7**	**75.2**	-	-	-
	Proposed (SPHOC+ Beam Search decoding)	**95.7**	**85.2**	-	-	-

Table 7.4: Scene text recognition accuracy. "50"and "Full" denote the lexicon size used for constrained text recognition as defined in [65]. Results are divided into lexicon-based and unconstrained (lexicon-free) approaches. *DICT [25] is not lexicon-free due to incorporating ground-truth labels during training.

7.3.2 Result

Recognition

7.4 Discussion

In this section we want to discuss some of the advantages and limitations of our proposed model.

Figure 7.3: Attention weights at every level of PHOC is overlaid for visualization, at every level the corresponding part is given more weight.

7.4.1 Model Size comparison

One of the main contributions of this work is to generalize the PHOC embedding for lexicon free word recognition, to be able to recognize the word we increased the number of levels but instead of increasing the size of final feature vector we decreased it as our model computes one histogram of characters as PHOC can be interpreted as a concatenation of histogram of characters at different level. This hugely reduces the number of parameters as explained in figure 7.5. Additionally we make use of inception architecture proposed in [58], which also allows our model to go deeper than other models while keeping the less number of parameters, in particular our feature extractor uses sequence of inception modules as shown in 7.6. We tested with different number of inception modules finally the best performance achieved by using 4 inception modules sequentially.

Above two design choices yields in a much smaller model size (number of parameters). in the following table we reported a comparison among the reported deep networks for computation of PHOC. For PHOCNET we computed the model available in []. For [], we used the resnet

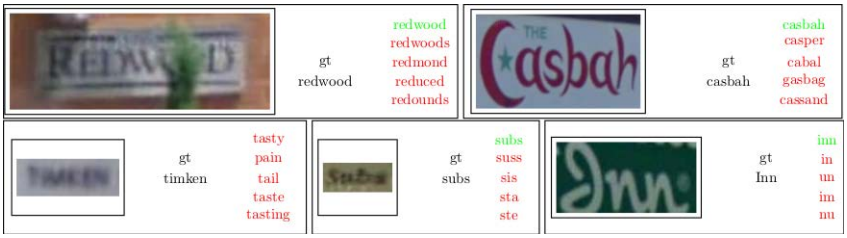


Figure 7.4: Some example results on SVT dataset

Method	Model Size(in MB)	Number of Parameters (in millions)
PHOCNET [57]	277	138
Deep Feature Embedding [30]		
Triplet CNN [67]		
Ours (SPHOC)	72	34

Table 7.5: Comparison between different approaches to compute PHOC

7.4.2 Training data requirement

As shown in the previous section that we used lot less number of parameters to compute PHOC, this also facilitates in the training process. As the number of free parameters are less and the goal is to generate a histogram of characters the amount of data required is less. In the 7.7, we plotted the recognition accuracy v/s the amount of training data. We can see that our model take only a fraction of data in comparison with PHOCNET.

7.5 Conclusion

In this paper we proposed a one stop solution for all problems related to text understanding in images. A popular embedding is first learned by a novel attention based architecture is used which facilitates to compute a large embedding vector with very less number of parameters. Attention allows to select features from the concerned area and computing embedding for that part combining which we can generate the global embedding. Thus we achieved a fully convolutional architecture for text un-

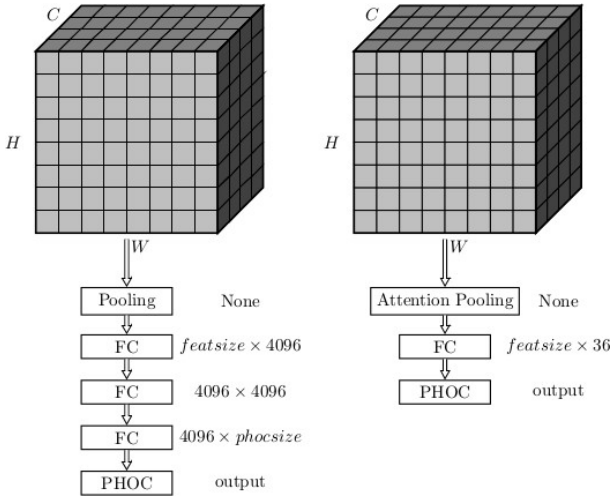


Figure 7.5: Comparing number of parameters: on the left based on holistic representation and on the right based on feature selection mechanism

derstanding. As our model does not need recurrent computation it is better suited for parallelism.

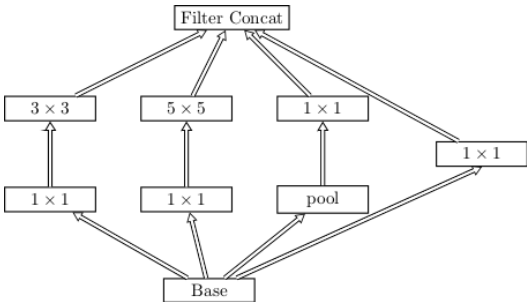


Figure 7.6: Inception module used in our model

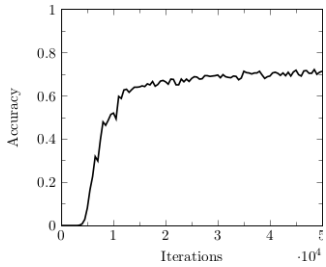


Figure 7.7: Training data requirement vs Accuracy

Chapter 8

Conclusion and Future works

Understanding text in diverse environment is a challenging problem. In last decades a lot of different techniques were proposed by different researchers. One of the common and crucial aspect of all these methods is how to represent a word image. Broadly there are two different ways to represent a word image either using sequence of feature vectors or using a fixed length. In this thesis we have studied both sequence based feature vector and fixed length feature vector for their suitability in segmentation free word spotting and lexicon-free word recognition.

In the first part of the thesis we extended the use of a fixed length features representation for segmentation free word spotting. In chapter 3, we proposed a segmentation-free word spotting method based on sliding window framework. we also propose to use pre-computed integral image of attributes for faster query evaluation. Further computation speed-up is achieved by indexing different regions of documents by character bi-grams.

Sliding window schemes is based on exhaustive search in entire document database, thus often proved to be expensive in terms of computation. Thus to overcome this in chapter 4, we proposed a bounding box proposal algorithm. We extended our indexing scheme proposed in chapter 3 to index using character n-gram instead of bi-gram. We have shown that using our proposed bounding box proposal algorithm can improve the performances dramatically. In chapter 4 we have also adapted the integral image of attributes to the variable sized text boxes. This adaptation is proved to be beneficial

in increasing the recall and decreasing the false positives.

In recent times deep learning is being successfully applied in most computer vision problems with great results. We explored the possibility of segmentation-free word spotting using deep learning features. In chapter 5, We proposed a deep learning framework where we can input a document image and a set of pre-computed bounding boxes to generate PHOC descriptor for each of the bounding boxes. Analysis of this framework suggests that a comparable performance can be achieved by using CNN feature warping. We have also observed that our framework performs poorly in case of queries with less number of characters. We think quantization used by ROI pooling is the reason behind this limitation. We believe this can be rectified by using feature map at different depth, also ROI-align layer [20] can be used to reduce the effect of quantization.

In chapter 6 and 7 our focus of research is word recognition using attention based mechanisms. In chapter 6, we first use a standard sequence to sequence approach to recognize word images. We use LSTM based visual attention models, where at every step we attend to different parts of the word image to generate one character. In chapter 7 our goal is to proposed a unified solution for all the sub-problems in this area namely word spotting and lexicon-free word recognition. We observed that lexicon-free word recognition can be obtained using PHOC if sufficient level is used.

Publications

The following publications are a consequence of the research carried out during this thesis and give an idea of the progression that has been achieved

Journals

- Suman Ghosh and Ernest Valveny. Text Box Proposal for handwritten word spotting from documents, International Journal of Document Analysis and Research, June 2018, Volume 21, Issue 1–2, 91–108.

Conferences

- Suman Ghosh and Ernest Valveny. A sliding window framework for word spotting based on word attributes, in Iberian Conference on Pattern Recognition and Image Analysis, 2015, 652-661.
- Suman Ghosh and Ernest Valveny. Query by string word spotting based on character bi-gram indexing, International Conference of Document Analysis and Recognition, 2015, 881-885.
- Suman Ghosh, Lluis Gomez, Dimosthenis Karatzas and Ernest Valveny. Efficient indexing for Query By String text retrieval. International Conference of Document Analysis and Recognition, 2015, 1236-1240
- Suman Ghosh and Ernest Valveny. R-PHOC: Segmentation-Free Word Spotting using CNN, in International Conference of Document Analysis and Recognition, 2017, 881-885.
- Suman Ghosh, Ernest Valveny and Andrew D Bagdanov. Visual attention models for scene text recognition, International Conference of Document Analysis and Recognition, 2017,943-948.

Bibliography

[1] J. Almazán, A. Gordo, A. Fornés, and E. Valveny. Efficient exemplar word spotting. In *BMVC*, 2012.

[2] J. Almazán, A. Gordo, A. Fornés, and E. Valveny. Segmentation-free word spotting with exemplar svms. *Pattern Recognition*, 2014.

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

[4] 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.

[5] Ouais Alsharif and Joelle Pineau. End-to-end text recognition with hybrid hmm maxout models. *CoRR*, abs/1310.1811, 2013.

[6] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *CoRR*, abs/1409.0473, 2014.

[7] A. Bissacco, M. Cummins, Y. Netzer, and H. Neven. Photoocr: Reading text in uncontrolled conditions. In *2013 IEEE International Conference on Computer Vision*, pages 785–792, Dec 2013.

[8] T. Bluche, H. Ney, and C. Kermorvant. Tandem hmm with convolutional neural network for handwritten word recognition. In *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 2390–2394, May 2013.

[9] H. Bunke, S. Bengio, and A. Vinciarelli. Offline recognition of unconstrained handwritten texts using hmms and statistical language models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(6):709–720, June 2004.

[10] Zhanzhan Cheng, Fan Bai, Yunlu Xu, Gang Zheng, Shiliang Pu, and Shuigeng Zhou. Focusing attention: Towards accurate text recognition in natural images. *CoRR*, abs/1709.02054, 2017.

[11] Gabriella Csurka, Christopher R. Dance, Lixin Fan, Jutta Willamowski, and Cedric Bray. Visual categorization with bags of keypoints. In *In Workshop on Statistical Learning in Computer Vision, ECCV*, pages 1–22, 2004.

[12] Andreas Fischer, Andreas Keller, Volkmar Frinken, and Horst Bunke. Lexicon-free handwritten word spotting using character hmms. *Pattern Recognition Letters*, 33(7):934 – 942, 2012. Special Issue on Awards from ICPR 2010.

[13] Edward Fredkin. Trie memory. *Commun. ACM*, 3(9):490–499, September 1960.

[14] V. Frinken, A. Fischer, R. Manmatha, and H. Bunke. A novel word spotting method based on recurrent neural networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(2):211–224, Feb 2012.

[15] Ross Girshick. Fast r-cnn. In *International Conference on Computer Vision (ICCV)*, 2015.

[16] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Computer Vision and Pattern Recognition*, 2014.

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

[18] Albert Gordo. Supervised mid-level features for word image representation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.

[19] 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.

[20] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross B. Girshick. Mask R-CNN. *CoRR*, abs/1703.06870, 2017.

[21] Pan He, Weilin Huang, Yu Qiao, Chen Change Loy, and Xiaoou Tang. Reading scene text in deep convolutional sequences. 2016.

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

[23] Max Jaderberg, Karen Simonyan, Andrea Vedaldi, and A Zisserman. Deep structured output learning for unconstrained text recognition. *ICLR 2015*, 2014.

[24] Max Jaderberg, Karen Simonyan, Andrea Vedaldi, and A. Zisserman. Reading text in the wild with convolutional neural networks. *CORR/abs/1412.1842*, 2014.

[25] Max Jaderberg, Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Synthetic data and artificial neural networks for natural scene text recognition. In *NIPS Deep Learning Workshop 2014*, 2014.

[26] 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.

[27] Max Jaderberg, Andrea Vedaldi, and Andrew Zisserman. Deep features for text spotting. In *ECCV 2014*. https://bitbucket.org/jaderberg/eccv2014_textspotting, 2014.

[28] Michael N Jones and Douglas JK Mewhort. Case-sensitive letter and bigram frequency counts from large-scale english corpora. *Behavior research methods, instruments, & computers*, 36(3):388–396, 2004.

[29] A. Kovalchuk, L. Wolf, and N. Dershowitz. A simple and fast word spotting method. In *2014 14th International Conference on Frontiers in Handwriting Recognition*, pages 3–8, Sept 2014.

[30] Praveen Krishnan, Kartik Dutta, and CV Jawahar. Deep feature embedding for accurate recognition and retrieval of handwritten text. In *Frontiers in Handwriting Recognition (ICFHR), 2016 15th International Conference on*, pages 289–294. IEEE, 2016.

[31] Praveen Krishnan and CV Jawahar. Matching handwritten document images. In *European Conference on Computer Vision*, pages 766–782. Springer, 2016.

[32] Chen-Yu Lee and Simon Osindero. Recursive recurrent nets with attention modeling for OCR in the wild. *CoRR*, abs/1603.03101, 2016.

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

[34] Yann Leydier, Frank Lebourgeois, and Hubert Emptoz. Text search for medieval manuscript images. *Pattern Recognition*, 40(12):3552 – 3567, 2007.

[35] Yann Leydier, Asma Ouji, Frank LeBourgeois, and Hubert Emptoz. Towards an omnilingual word retrieval system for ancient manuscripts. *Pattern Recognition*, 42(9):2089 – 2105, 2009.

[36] Y. Liang, M.C. Fairhurst, and R.M. Guest. A synthesised word approach to word retrieval in handwritten documents. *Pattern Recognition*, 45(12):4225 – 4236, 2012.

[37] Laurence Likforman-Sulem, Abderrazak Zahour, and Bruno Taconet. Text line segmentation of historical documents: a survey. *International Journal of Document Analysis and Recognition (IJDAR)*, 9(2):123–138, Apr 2007.

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

[39] Lukáš Neumann and Jiří Matas. Real-time scene text localization and recognition. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 3538–3545. IEEE, 2012.

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

[41] Florent Perronnin, Jorge Sánchez, and Thomas Mensink. Improving the fisher kernel for large-scale image classification. In Kostas Daniilidis, Petros Maragos, and Nikos Paragios, editors, *Computer Vision – ECCV 2010*, pages 143–156, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg.

[42] Rodriguez-serrano Jose Perronnin, Florent. Local gradient histogram features for word spotting in unconstrained handwritten documents. *IEEE Conference on Frontiers of Handwritten Word Recognition*.

[43] Vu Pham, Christopher Kermorvant, and Jérôme Louradour. Dropout improves recurrent neural networks for handwriting recognition. *CoRR*, abs/1312.4569, 2013.

[44] Arik Poznanski and Lior Wolf. Cnn-n-gram for handwriting word recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2305–2314, 2016.

[45] Tony M. Rath and R. Manmatha. Word spotting for historical documents. *International Journal of Document Analysis and Recognition (IJDAR)*, 9(2):139–152, Apr 2007.

[46] Jose A. Rodriguez-Serrano, Albert Gordo, and Florent Perronnin. Label embedding: A frugal baseline for text recognition. *International Journal of Computer Vision*, 113(3):193–207, 2015.

[47] VeróNica Romero, Alicia FornéS, NicoláS Serrano, Joan Andreu SáNchez, Alejandro H Toselli, Volkmar Frinken, Enrique Vidal, and Josep LladóS. The esposalles database: An ancient marriage license corpus for off-line handwriting recognition. *Pattern Recognition*, 46(6):1658–1669, 2013.

[48] L. Rothacker and G. A. Fink. Segmentation-free query-by-string word spotting with bag-of-features hmms. In *2015 13th International Conference on Document Analysis and Recognition (ICDAR)*, pages 661–665, Aug 2015.

[49] Leonard Rothacker, Marcal Rusinol, and Gernot A Fink. Bag-of-features hmms for segmentation-free word spotting in handwritten documents. In *Document Analysis and Recognition (ICDAR), 2013 12th International Conference on*, pages 1305–1309. IEEE, 2013.

[50] Marcal Rusinol, David Aldavert, Ricardo Toledo, and Josep Llados. Efficient segmentation-free keyword spotting in historical document collections. *Pattern Recognition*, 48(2):545 – 555, 2015.

[51] J. A. Sánchez, V. Romero, A. H. Toselli, M. Villegas, and E. Vidal. Icdar2017 competition on handwritten text recognition on the read dataset. In *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, volume 01, pages 1383–1388, Nov 2017.

[52] 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*, 39:2298–2304, 2017.

[53] Baoguang Shi, Xinggang Wang, Pengyuan Lyu, Cong Yao, and Xiang Bai. Robust scene text recognition with automatic rectification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4168–4176, 2016.

[54] David L Smith, Jacqueline Field, and Erik Learned-Miller. Enforcing similarity constraints with integer programming for better scene text recognition. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 73–80. IEEE, 2011.

[55] Lucas Panaretos Sosa, S. M. Lucas, A. Panaretos, L. Sosa, A. Tang, S. Wong, and R. Young. Icdar 2003 robust reading competitions. In *In Proceedings of the Seventh International Conference on Document Analysis and Recognition*, pages 682–687. IEEE Press, 2003.

[56] Sebastian Sudholt and Gernot A. Fink. PHOCNet : A Deep Convolutional Neural Network for Word Spotting in Handwritten Documents. In *Proc. Int. Conf. on Frontiers in Handwriting Recognition*, 2016.

[57] Sebastian Sudholt and Gernot A. Fink. PHOCNet: A Deep Convolutional Neural Network for Word Spotting in Handwritten Documents. In *Proc. Int. Conf. on Frontiers in Handwriting Recognition*, Shenzhen, China, 2016. Winner of the best paper award.

[58] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. *CoRR*, abs/1512.00567, 2015.

[59] K. Terasawa and Y. Tanaka. Slit style hog feature for document image word spotting. In *2009 10th International Conference on Document Analysis and Recognition*, pages 116–120, July 2009.

[60] Jasper RR Uijlings, Koen EA Van De Sande, Theo Gevers, and Arnold WM Smeulders. Selective search for object recognition. *International journal of computer vision*, 104(2):154–171, 2013.

[61] U. v. Marti and H. Bunke. A full english sentence database for off-line handwriting recognition. In *In Proc. Int. Conf. on Document Analysis and Recognition*, pages 705–708, 1999.

[62] Andreas Veit, Tomas Matera, Lukas Neumann, Jiri Matas, and Serge Belongie. Coco-text: Dataset and benchmark for text detection and recognition in natural images. In *arXiv preprint arXiv:1601.07140*, 2016.

[63] Paul Viola and Michael Jones. Rapid object detection using a boosted cascade of simple features. In *Computer Vision and Pattern Recognition, 2001. CVPR*

2001. Proceedings of the 2001 IEEE Computer Society Conference on, volume 1, pages I–I. IEEE, 2001.

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

[65] Kai Wang, Boris Babenko, and Serge Belongie. End-to-end scene text recognition. In *2011 International Conference on Computer Vision*, pages 1457–1464. IEEE, 2011.

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

[67] Tomas Wilkinson and Anders Brun. Semantic and verbatim word spotting using deep neural networks. In *Frontiers in Handwriting Recognition (ICFHR), 2016 15th International Conference on*, pages 307–312. IEEE, 2016.

[68] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron C. Courville, Ruslan Salakhutdinov, Richard S. Zemel, and Yoshua Bengio. Show, attend and tell: Neural image caption generation with visual attention. *CoRR*, abs/1502.03044, 2015.

[69] Cong Yao, Xiang Bai, Baoguang Shi, and Wenyu Liu. Strokelets: A learned multi-scale representation for scene text recognition. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2014.

[70] Fei Yin, Yi-Chao Wu, Xu-Yao Zhang, and Cheng-Lin Liu. Scene text recognition with sliding convolutional character models. *CoRR*, abs/1709.01727, 2017.

[71] X. Zhang and C. L. Tan. Segmentation-free keyword spotting for handwritten documents based on heat kernel signature. In *2013 12th International Conference on Document Analysis and Recognition*, pages 827–831, Aug 2013.

[72] C Lawrence Zitnick and Piotr Dollár. Edge boxes: Locating object proposals from edges. In *European conference on computer vision*, pages 391–405. Springer, 2014.